

Karl Aberer et al. (Eds.)

LNCS 4825

# The Semantic Web

6th International Semantic Web Conference  
2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007  
Busan, Korea, November 2007, Proceedings

 Springer

*Commenced Publication in 1973*

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

*Lancaster University, UK*

Takeo Kanade

*Carnegie Mellon University, Pittsburgh, PA, USA*

Josef Kittler

*University of Surrey, Guildford, UK*

Jon M. Kleinberg

*Cornell University, Ithaca, NY, USA*

Friedemann Mattern

*ETH Zurich, Switzerland*

John C. Mitchell

*Stanford University, CA, USA*

Moni Naor

*Weizmann Institute of Science, Rehovot, Israel*

Oscar Nierstrasz

*University of Bern, Switzerland*

C. Pandu Rangan

*Indian Institute of Technology, Madras, India*

Bernhard Steffen

*University of Dortmund, Germany*

Madhu Sudan

*Massachusetts Institute of Technology, MA, USA*

Demetri Terzopoulos

*University of California, Los Angeles, CA, USA*

Doug Tygar

*University of California, Berkeley, CA, USA*

Moshe Y. Vardi

*Rice University, Houston, TX, USA*

Gerhard Weikum

*Max-Planck Institute of Computer Science, Saarbruecken, Germany*



Karl Aberer, Key-Sun Choi, Natasha Noy  
Dean Allemang, Kyung-Il Lee, Lyndon Nixon  
Jennifer Golbeck, Peter Mika, Diana Maynard  
Riichiro Mizoguchi, Guus Schreiber  
Philippe Cudré-Mauroux (Eds.)

# The Semantic Web

6th International Semantic Web Conference  
2nd Asian Semantic Web Conference  
ISWC 2007 + ASWC 2007  
Busan, Korea, November 11-15, 2007  
Proceedings

Volume Editors

Karl Aberer and Philippe Cudré-Mauroux, EPFL, Switzerland  
E-mail: {karl.aberer;philippe.cudre-mauroux}@epfl.ch

Key-Sun Choi, Korea Advanced Institute of Science and Technology, Korea  
E-mail: kschoi@cs.kaist.ac.kr

Natasha Noy, Stanford University, USA, E-mail: noy@stanford.edu

Dean Allemang, TopQuadrant, USA, E-mail: dallemang@topquadrant.com

Kyung-Il Lee, Saltlux Inc., Korea, E-mail: tony@saltlux.com

Lyndon Nixon, Free University Berlin, Germany  
E-mail: nixon@inf.fu-berlin.de

Jennifer Golbeck, University of Maryland, USA, E-mail: jgolbeck@umd.edu

Peter Mika, Yahoo! Research Barcelona, Spain, E-mail: pmika@yahoo-inc.com

Diana Maynard, University of Sheffield, Sheffield, S1 4DP, UK  
E-mail: diana@dcs.shef.ac.uk

Riichiro Mizoguchi, Osaka University, Japan  
E-mail: miz@ei.sanken.osaka-u.ac.jp

Guus Schreiber, Free University Amsterdam, The Netherlands  
E-mail: schreiber@cs.vu.nl

Library of Congress Control Number: 2007938332

CR Subject Classification (1998): H.4, H.3, C.2, H.5, F.3, I.2, K.4

LNCS Sublibrary: SL 3 – Information Systems and Application  
incl. Internet/Web and HCI

ISSN 0302-9743

ISBN-10 3-540-76297-3 Springer Berlin Heidelberg New York

ISBN-13 978-3-540-76297-3 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media  
springer.com

© Springer-Verlag Berlin Heidelberg 2007  
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India  
Printed on acid-free paper SPIN: 12181520 06/3180 5 4 3 2 1 0

# Preface

The World-Wide Web continues to grow and new technologies, modes of interactions, and applications are being developed. Building on this growth, Semantic Web technologies aim at providing a shared semantic information space, changing qualitatively our experiences on the Web. As Semantic Web technologies mature and permeate more and more application areas, new research challenges are coming to the fore and some unsolved ones are becoming more acute. These issues include creating and managing Semantic Web content, making Semantic Web applications robust and scalable, organizing and integrating information from different sources for novel uses, making semantics explicit in order to improve our overall experience with information technologies, and thus enabling us to use the wealth of information that is currently available in digital form for addressing our everyday tasks.

These and other challenges brought several hundred of researchers, developers, government workers, venture capitalists, students, and users to Busan, Korea for the Sixth International Semantic Web Conference which was held jointly with the 2nd Asian Semantic Web Conference (ISWC 2007 + ASWC 2007).

We are excited to offer this volume with the main proceedings of ISWC 2007 + ASWC 2007 to the growing community of researchers and practitioners of the Semantic Web. The tremendous response to our Call for Papers from a truly international community of researchers and practitioners from 30 countries, the careful nature of the review process, and the breadth and scope of the papers finally selected for inclusion in this volume all speak to the quality of the conference and to the contributions made by the papers in these proceedings. This year, all papers will again be freely available online to all interested parties through an unprecedented agreement with Springer. In addition, several online applications will provide access to semantically annotated information about the papers and the conference itself.

The Research Track attracted 257 submissions, an 18% increase over ISWC 2006, which shows a robust growth of interest in the field. The review process was conducted by a large international Program Committee and headed by 20 Vice-Chairs, along with Program Chairs. Each Vice-Chair was responsible for papers in a particular research area. The review process consisted of three distinct phases. First, each paper was evaluated by at least three members of the Program Committee. Second, Vice-Chairs led a discussion among the reviewers on each of the papers and generated a meta-review and a final recommendation, soliciting additional reviews in some cases. Finally, the Program Chairs made

the definitive decisions on the papers. This year, we also took an unusual step of accepting a small number of papers conditionally: these were papers with very strong technical content, but problems in presentation. The revision process for these papers was supervised by a Vice-Chair, who then made the final decision on the acceptance. Such a structured process ensured a high-quality review, but of course required a great deal of effort from the members of the Program Committee and the Vice-Chairs. In total, 50 papers were accepted out of 257 submissions, a 19% acceptance rate.

The Semantic Web In-Use track accepted 12 papers this year, out of 29 submissions, which were reviewed by three referees each. The submissions came from industry primarily and demonstrate the continuing interest in Semantic Web technologies from companies. Submissions came both from applications in industrial sectors, such as healthcare and the public sector, and from research work applicable to industry, such as supporting data integration and migration from legacy systems. Hence, two critical aspects of the Semantic Web research in industry were covered: real-world experiences from using semantics in business scenarios and bringing fundamental research to maturity for application in industry. Bringing these two aspects together, as promoted by the Semantic Web In-Use track, is an important contribution to establishing a strong basis for Semantic technology use in the business of the future.

A unique aspect of the International Semantic Web Conference is the Semantic Web Challenge. This is an international competition in which participants from both academia and industry are encouraged to show how Semantic Web techniques can provide useful or interesting applications to end-users. Over the last five years, the Semantic Web Challenge has seen more than 50 integrated applications built around distributed data sources using semantic descriptions to handle data. The challenge received 23 submissions this year, a large increase over the previous year, making the selection process harder. The topics ranged from expertise location to cultural heritage, and from television to climate change. Keeping with the broad international appeal, there were submissions from Europe, North America, and Asia, from both industry and academia. The winner of the challenge was announced at the ISWC Conference and received 1,000 Euro in travel support and a 250-Euro voucher to purchase books from Springer.

This year ISWC 2007 + ASWC 2007 hosted, for the third year running, a doctoral consortium for PhD students within the Semantic Web community, giving them the opportunity to discuss in detail their research topics and plans and to get extensive feedback from leading scientists in the field. A most interesting invited talk was also given by Fabio Ciravegna from the University of Sheffield, who discussed the challenges and requirements for Semantic Web technologies when confronted with the task of knowledge acquisition and sharing in large,

complex distributed environments. Submissions were reviewed by a panel of experienced researchers. The acceptances comprised Six papers and Six posters, which were presented in a full-day session. Each student was also assigned a mentor who led the discussions following the presentation of the work, and provided more detailed feedback and comments, focusing on the PhD proposal itself and presentation style, as well as on the actual work presented. The mentors were drawn from the set of reviewers and comprised some leading researchers in the field – both from academia and from industry.

Keynote talks from eminent scientists and practitioners punctuated the conference. The founder of the Internet Archive, Brewster Kahle, discussed the use of Semantic Web technologies in providing universal access to human knowledge. Ronald M. Kaplan, Chief Technology and Science Officer at Powerset Inc., examined how Natural Language techniques can complement Semantic Web technologies, both from data consumer and from service supplier perspectives. Christopher Welty, from the IBM T.J. Watson Research Center, discussed scalable information management techniques based on ontologies and the Semantic Web.

Workshops and tutorials are a vital part of the conference program. This year, we received a record number of workshop proposals, ranging in topics from ontology matching, the lexicon/ontology interface, or service matchmaking, to applications of Semantic Web technologies in the domain of cultural heritage. In addition to a strong selection of tutorials chosen from the submitted tutorial proposals, the conference included an invited tutorial, in which key researchers in the area of the Semantic Web provided a comprehensive introduction to the field. We are grateful to Mike Dean and Young-Tack Park (Poster and Demo Co-chairs), John Domingue and David Martin (Tutorial Co-chairs), and Harith Alani and Geert-Jan Houben (Workshop Co-chair) for ensuring the success of these events.

We offer many thanks to Tom Heath and Knud Möller, Metadata Co-chairs, for their expert coordination of the production of the semantic mark-up associated with each contribution to the conference and the applications and exhibits that use that mark-up.

The meeting would not have been possible without the tireless work of the Local Chair, Sung-Kook Han. We thank him and his team for providing excellent local arrangements and for making everyone feel welcome in Busan. The superb work of Masahiro Hori (Publicity Chair), York Sure and Young-Sik Jeong (Sponsor Co-chairs), Noboru Shimizu and Myung-Hwan Koo (Exhibition Co-chair) was critical to the conference success.

Finally, we would like to acknowledge the Semantic Web Science Association for providing the organizational oversight for ISWC 2007 + ASWC 2007 and the conference sponsors for their generous support.

In conclusion, ISWC 2007 + ASWC 2007 was an exciting event, reflecting the high level of energy, creativity, and productivity in the Semantic Web community.

November 2007

Karl Aberer  
Key-Sun Choi  
Natasha Noy  
Dean Allemang  
Kyung-Il Lee  
Lyndon Nixon  
Jennifer Golbeck  
Peter Mika  
Diana Maynard  
Riichiro Mizoguchi  
Guus Schreiber  
Philippe Cudré-Mauroux

# Organization

General Co-chairs	Riichiro Mizoguchi, Osaka University, Japan Guus Schreiber, Free University Amsterdam, The Netherlands
Local Organization Chair Research Track Co-chairs	Sung-Kook Han, Wonkwang University, Korea Karl Aberer, EPFL, Switzerland Key-Sun Choi, Korea Advanced Institute of Science and Technology, Korea
Workshop Co-chairs	Natasha Noy, Stanford University, USA Harith Alani, University of Southampton, UK Geert-Jan Houben, Vrije Universiteit Brussel, Belgium
Tutorial Co-chairs	John Domingue, Knowledge Media Institute, Open University, UK David Martin, SRI, USA
Semantic Web In-Use Track Co-chairs	Dean Allemang, TopQuadrant, USA Kyung-Il Lee, Saltlux Inc., Korea Lyndon Nixon, Free University Berlin, Germany
Semantic Web Challenge Co-chairs	Jennifer Golbeck, University of Maryland, USA Peter Mika, Yahoo! Research Barcelona, Spain
Poster and Demo Co-chairs	Young-Tack, Park Sonngsil University, Korea Mike Dean, BBN, USA
Doctoral Consortium Chair Sponsor Co-chairs	Diana Maynard, University of Sheffield, UK Young-Sik Jeong, Wonkwang University, Korea York Sure, University of Karlsruhe, Germany
Exhibition Co-chairs	Myung-Hwan Koo, Korea Telecom, Korea Noboru Shimizu, Keio Research Institute, Japan
Publicity Chair Proceedings Chair Metadata Co-chairs	Masahiro Hori, Kansai University, Japan Philippe Cudré-Mauroux, EPFL, Switzerland Tom Heath, Knowledge Media Institute, Open University, UK Knud Möller, DERI, National University of Ireland, Galway, Ireland

## Local Organizing Committee

Jeong-Jin Yang, Catholic University, Korea  
Jung-Min Choi, Hanyang University, Korea  
Ki-Sun Choi, KIAST, Korea

Kyung-Il Lee, Saltlux, Korea  
Myung-Gil Jang, ETRI, Korea  
Sung-Kook Han, Wonkwang University, Korea  
Se-Young Park, Kyungpook National University, Korea  
Young-Sik Jeong, Wonkwang University, Korea  
Young-Tak Park, Sonngsil University, Korea  
Jeon-Yong Byun, Dongguk University, Korea  
Joong-min Choi, Hanyang University, Korea  
Se-Yeong Park, Kyungpook National University, Korea  
Young-Tack Park, Sonngsil University, Korea  
Suk-Doo Choi, Hansung University, Korea  
Young-Man Ko, Sungkyungwan University, Korea  
Min-Koo Kim, Ajou University, Korea  
SungHyuk Kim, Sookmyung Women's University, Korea  
In-Cheol Kim, Kyonggi University, Korea  
Se-Jin Nam, OntoWeb, Korea  
Nam-Young Joon, Chung-Ang University, Korea  
Min-Duck KI, Konkuk University, Korea  
Sang-Kyu Park, ETRI, Korea  
Chun-Myoung Park, Chungju National University, Korea  
JongHyun Baek, Daewoo Information System, Korea  
Won-Kyung Sung, KISTI, Korea  
Moo-Jung An, LG C&S, Korea  
Sam-Gyun Oh, Sungkyungwan University, Korea  
Kyu-Chol Lee, Chungnam National University, Korea  
Sang-Ku Lee, Seoul National University, Korea  
Jae-Ho Lee, Seoul University, Korea  
Jungwoo Lee, Yonsei University, Korea  
Hae-Yun Lee, Hankuk University of Foreign Studies, Korea  
GunSik Jo, Inha University, Korea  
Hee-Rahk Chae, Hankuk University of Foreign Studies, Korea  
Byung-il Choi, Seoul National University, Korea  
Hung-Joung Choi, INEK Inc, Korea  
Kwang-Rok Han, Hoseo University, Korea  
Song-Soon Hyun, Dawin Inc, Korea

## **Program Committee**

### **Area Vice-Chairs**

Richard Benjamins, Intelligent Software Components(iSOCO), Spain

Abraham Bernstein, University of Zurich, Switzerland



Dan Brickley, Asemanatics, UK

Vinay Chaudhri, SRI, USA

Hamish Cunningham, University of Sheffield, UK

Giuseppe De Giacomo, University of Rome La Sapienza, Italy

Jérôme Euzenat, INRIA Rhône-Alpes, France

Tim Finin, UMBC, USA

Carole Goble, University of Manchester, UK

Ian Horrocks, University of Manchester, UK

David Karger, MIT, USA

Hong-Gee Kim, Seoul National University, Korea

Matthias Klusch, DFKI, Germany

Manolis Koubarakis, National and Kapodistrian University of Athens, Greece

Enrico Motta, Open University, UK

Hideaki Takeda, University of Tokyo, Japan

Val Tannen, University of Pennsylvania, USA

Frank van Harmelen, Vrije Universiteit Amsterdam, The Netherlands

Takahira Yamaguchi, Keio University, Japan

Steffen Staab, University of Koblenz-Landau, Germany

## **Program Committee, Research Track**

Mark Ackerman, University of Michigan, USA

Yuan An, University of Toronto, Canada

Chandrasekhar Anantaram, Tata Consultancy Services Limited, India

Ion Androutsopoulos, Athens University of Economics and Business, Greece

Galia Angelova, Bulgarian Academy of Sciences, Bulgaria

Lora Aroyo, Free University Amsterdam, The Netherlands

Budak Arpinar, University of Georgia, USA

Yannis Avrithis, NTUA, Greece

Danny Ayers, Italy

Christopher Baker, Institute for Infocomm Research, Singapore

Sean Bechhofer, University of Manchester, UK

Sonia Bergamaschi, Università di Modena, Italy

Leopoldo Bertossi, Carleton University, Canada

Kalina Bontcheva, University of Sheffield, UK

Alex Borgida, Rutgers University, USA

Paolo Bouquet, University of Trento, Italy

John Breslin, DERI Galway, Ireland

Paul Buitelaar, DFKI, Germany

Mark Burstein, BBN, USA

Diego Calvanese, Free University of Bozen-Bolzano, Italy

Jeremy Carroll, HP, UK

Harry Chen, Image Matters, USA

Adam Cheyer, SRI International, USA

Jae Hwa Choi, Dankook University, Korea

Jan Chomicki, University at Buffalo, USA

Vassilis Christophides, FORTH-ICS, Greece

Philipp Cimiano, University of Karlsruhe, Germany

Fabio Ciravegna, University of Sheffield, UK

Nigel Collier, NII, Japan

Oscar Corcho, University of Illinois at Chicago, UK

Isabel Cruz, University of Manchester, USA

Bernardo Cuenca Grau, University of Manchester, UK

Mathieu D'Aquin, KMi, The Open University, UK

Jos de Bruijn, University of Bolzano, Italy

Mike Dean, BBN Technologies, USA  
Stefan Decker, DERI Galway, Ireland  
Thierry Declerck, DFKI, Germany  
Alin Deutsch, University of California, San Diego, USA  
Ian Dickinson, HP, UK  
Li Ding, Stanford University, USA  
Martin Dzbor, KMi, Open University, UK, UK  
Michael Erdmann, Ontoprise, Germany  
Boi Faltings, EPFL, Switzerland  
Wenfei Fan, University of Edinburgh, UK  
Dieter Fensel, University of Innsbruck, Austria  
Achille Fokoue, IBM Research, USA  
Richard Fritzson, USA  
Ariel Fuxman, Microsoft Search Labs, USA  
Aldo Gangemi, CNR-ISTC, Italy  
Fausto Giunchiglia, University of Trento, Italy  
Birte Glimm, University of Manchester, UK  
Richard Goodwin, IBM Research, USA  
Mark Greaves, Vulcan, USA  
Gianluigi Greco, University of Calabria, Italy  
Marko Grobelnik, J. Stefan Institute, Slovenia  
Tom Gruber, RealTravel, USA  
Ramanathan Guha, Google, USA  
Volker Haarslev, Concordia, Canada  
Peter Haase, Karlsruhe University, Germany  
Siegfried Handschuh, National University of Ireland in Galway, Ireland  
Manfred Hauswirth, DERI Galway, Ireland  
Sandro Hawke, W3C, USA  
Patrick Hayes, University of West Florida, USA  
Jeff Heflin, Lehigh University, USA  
Masahiro Hori, Kansai University, Japan  
Matthew Horridge, University of Manchester, UK  
Andreas Hotho, University of Kassel, Germany  
Eero Hyvnen, Helsinki University of Technology, Finland  
Ryutaro Ichise, National Institute of Informatics, Japan  
Arantza Illarramendi, Universidad del Pais Vasco, Spain  
Noriaki Izumi, National Institute of Advanced Industrial Science and  
Technology (AIST), Japan  
Anupam Joshi, UMBC, USA  
Jason J. Jung, Inha University, Korea  
Lalana Kagal, MIT, USA  
Yannis Kalfoglou, University of Southampton, UK  
Bo Young Kang, Seoul National University, Korea  
Vangelis Karkaletsis, Institute of Informatics and Telecommunications, Greece  
Vipul Kashyap, Partners Healthcare System, USA

Takahiro Kawamura, Toshiba, Japan  
Yevgeny Kazakov, University of Manchester, UK  
Anastasios Kementsietsidis, University of Edinburgh, UK  
Brian Kettler, Lockheed Martin Advanced Technology Labs, USA  
Deepali Khushraj, Nokia Research Center, USA  
Hak Lae Kim, DERI Galway, Ireland  
Ja Hee Kim, Seoul National University of Technology, Korea  
Yasuhiko Kitamura, Kwansai Gakuin University, Japan  
Yoshinobu Kitamura, Osaka University, Japan  
Michel Klein, Vrije Universiteit Amsterdam, The Netherlands  
In Young Ko, Information and Communications University (ICU), Korea  
Paul Kogut, Lockheed-Martin, USA  
Seiji Koide, Ishikawajima-Harima Heavy Industries Co., Ltd, Japan  
Yiannis Kompatsiaris, ITI, Greece  
Kouji Kozaki, Osaka University, Japan  
Georg Lausen, University of Freiburg, Germany  
Jae Ho Lee, University of Seoul, Korea  
Domenico Lembo, Sapienza University of Rome, Italy  
Joshua Levy, SRI International, USA  
Juanzi Li, Tsinghua University, China  
Yaoyong Li, University of Sheffield, UK  
Alexander Löser, SAP, Germany, Germany  
Shiyong Lu, Wayne State University, USA  
Carsten Lutz, TU Dresden, Germany  
Zlatina Marinova, Sirma Group Corp., Bulgaria  
Scott Marshall, The Netherlands  
Yutaka Matsuo, National Institute of Advanced Industrial Science and  
Technology, Japan  
Cynthia Matuszek, Cycorp, USA  
Luke McDowell, United States Naval Academy, USA  
Prasenjit Mitra, The Pennsylvania State University, USA  
Boris Motik, University of Manchester, UK  
Yohei Murakami, National Institute of Information and Communication  
Technology, Japan  
Mark Musen, Stanford University, USA  
Wolfgang Nejdl, L3S and University of Hannover, Germany  
Tim Oates, UMBC, USA  
Sascha Ossowski, University Rey Juan Carlos, Spain  
Jeff Pan, University of Aberdeen, UK  
Jack Park, SRI International, USA  
Terry Payne, University of Southampton, UK  
Adam Pease, Articulate Software, USA  
Filip Perich, Shared Spectrum, USA  
Paulo Pinheiro da Silva, The University of Texas at El Paso, USA  
Dimitri Plexousakis, FORTH, Greece

Axel Polleres, Universidad Rey Juan Carlos, Spain  
Alexandra Poulouvassilis, University of London, UK  
Geri Reif, University of Zurich, Switzerland  
Ulrich Reimer, University of Applied Sciences St. Gallen, Switzerland  
Riccardo Rosati, University of Rome La Sapienza, Italy  
Marie-Christine Rousset, LIG (Laboratoire d'Informatique de Grenoble), France  
Lloyd Rutledge, Telematica Instituut and CWI, The Netherlands  
Alan Ruttenberg, UK  
Vassilis Samoladas, Technical University of Crete, Greece  
Uli Sattler, University of Manchester, UK  
Kent Seamons, BYU, USA  
Luciano Serafini, IRS Trento, Italy  
Paul Shabajee, HP Labs, UK  
Amit Sheth, Kno.e.sis Center, Wright State University, USA  
Pavel Shvaiko, University of Trento, Italy  
Wolf Siberski, L3S, Germany, Germany  
Kiril Simov, Bulgarian Academy of Science, Bulgaria  
Michael Sintek, DFKI, Germany, Germany  
Sergej Sizov, University of Koblenz, Germany, Germany  
Spiros Skiadopoulos, University of the Peloponnese, Greece  
Kavitha Srinivas, IBM Research, USA  
Giorgos Stamou, National Technical University of Athens, Greece  
Robert Stevens, University of Manchester, UK  
Michael Strube, EML Research, Germany  
Heiner Stuckenschmidt, University of Mannheim, Germany  
Rudi Studer, University of Karlsruhe, Germany  
Gerd Stumme, University of Kassel, Germany  
Valentina Tamma, University of Liverpool, UK  
Sergio Tessaris, University of Bolzano, Italy  
Bhavani Thuraisingham, University of Texas at Dallas, USA  
Francesca Toni, Imperial College, UK  
Raphael Troncy, CWI, The Netherlands  
Walt Trzuskowski, NASA, USA  
Christos Tryfonopoulos, Max-Planck-Institut für Informatik, Germany  
Victoria Uren, KMi The Open University, UK  
Mike Uschold, Boeing, USA  
Jacco van Ossenbruggen, CWI, Amsterdam, The Netherlands  
Ubbo Visser, University of Bremen, Germany  
Max Voelkel, AIFB, Universität Karlsruhe, Germany  
Christopher Welty, IBM Research, USA  
Max Wilson, University of Southampton, UK  
Michael Witbrock, Cycorp, USA  
Guizhen Yang, SRI International, UK  
Milena Yankova, Sirma AI Ltd., Bulgaria  
Peter Yim, CIM Engineering, USA  
Lina Zhou, UMBC, USA

## **Program Committee, Semantic Web In-Use Track**

Alec Morgan, GM R&D center, USA  
Andre Valente, Alelo Inc., USA  
Andy Crapo, GE Global Research, USA  
Chris Bizer, Free University of Berlin, Germany  
Chris Preist, Hewlett Packard Labs, UK  
Chris van Aart Sogeti, Netherlands  
Dave Reynolds, Hewlett-Packard Labs, UK  
David Karger, MIT, USA  
Deepali Khushraj, Nokia Research, USA  
Enrico Coiera, University of New South Wales, Australia  
Gary Edwards, ISX Corporation, USA  
Holger Knublauch, Top Quadrant, USA  
Ivan Herman, World Wide Web Consortium, USA  
John Davies, British Telecom, UK  
John Domingue, Open University, UK  
Juan Prieto, XimetriX, Spain  
Juergen Angele, Ontoprise, Germany  
Lars Garshol, Ontopia, Norway  
Leo Sauermann DFKI, Karlsruhe University, Germany  
Luca Gilardoni, Quinary, Italy  
Martin Hepp, University of Innsbruck, Austria  
Mike Dean, BBN Technologies, USA  
Mike Denny, Consultant, USA  
Ralph Traphoener, empolis GmbH, Denmark  
Rama Akkiraju, IBM, USA  
Shoji Kurakake, NTT DoCoMo, Japan  
Siegfried Handschuh, National University of Ireland Galway, Ireland  
Steve Cayzer, Hewlett-Packard Labs, UK  
Vinay Chaudhri, SRI, USA

## **Program Committee, Doctoral Consortium Track**

Lora Aroyo, Free University Amsterdam, The Netherlands  
Paul Buitelaar, DFKI, Germany  
Joerg Diederich, L3S Research Center, Germany  
Martin Dzbor, Open University, UK  
Adam Funk, University of Sheffield, UK  
Aldo Gangemi, CNR-ISTC, Italy  
Marko Grobelnik, J. Stefan Institute, Slovenia  
Tom Gruber, RealTravel, USA  
Peter Haase, Karlsruhe U, Germany  
Siegfried Handschuh, National University of Ireland in Galway, Ireland  
Alain Léger, Wanadoo, France

Enrico Motta, Open University, UK  
 Jeff Pan, University of Aberdeen, UK  
 Elena Simperl, DERI Galway, Ireland  
 Holger Wache, Fachhochschule Nordwestschweiz, Switzerland

## Additional Reviewers

Sudhir Agarwal	Rigel Gjomemo
Luis Rodrigo Aguado	Alyssa Glass
José Luis Ambite	Alfio Gliozzo
Alia Amin	Antoon Goderis
Wee Tiong Ang	Karthik Gomadam
Alessandro Artale	Bernardo Cuenca Grau
Wouter van Atteveldt	Stephan Grimm
Robert Battle	Gunnar Aastrand Grimnes
Jamal Bentahar	Tudor Groza
Antonis Bikakis	Slawomir Grzonkowski
Holger Billhardt	Alan Halverson
Ryan Blace	Hans-Jörg Happel
Joel Booth	Yoshihiko Hayashi
David Buján Carballal	John Hebler
David Bujn	Michiel Hildebrand
Francesco Carbone	Jörg Hoffmann
Charalambos Charalambous	Aidan Hogan
Peter Clark	June S. Hong
Vasa Curcin	Carlos Hurtado
Alfredo Cuzzocrea	Giovambattista Ianni
Danica Damljanovic	Antoine Isaac
Brian Davis	Anuj R. Jaiswal
Renaud Delbru	William Jarrold
Klaas Dellschaft	Rajaraman Kanagasabai
Xi Deng	Bo-Young Kang
Zhongli Ding	Zoi Kaoudi
Pavlin Dobrev	Gjergji Kasneci
Ji Donghong	Esther Kaufmann
Ludger van Elst	Ai Kawazoe
Matthew Fisher	Yevgeny Kazakov
Giorgos Flouris	Uwe Keller
Blaz Fortuna	Christoph Kiefer
Thomas Franz	Malte Kiesel
Naoki Fukuta	Ja-Hee Kim
Chiara Ghidini	Woo Joo Kim
Daniel Giacomuzzi	Dave Kolas
Luca Gilardoni	Stasinou Konstantopoulos
Andrew Gibson	Roman Korf

XVIII Organization

Ioannis Kotsiopoulos

Beate Krause

Kiriakos Kritikos

Hagen Langer

Ora Lassila

Sungin Lee

Jiwen Li

José-Manuel López-Cobo

Thomas Lukasiewicz

Rodrigue Carlos Nana Mbinkeu

Edgar Meij

Christian Meilicke

Eduardo Mena

Iris Miliaraki

Paolo Missier

Knud Möller

Fergal Monaghan

Junichiro Mori

Meenakshi Nagarajan

Amedeo Napoli

Pantelis Nasikas

Andriy Nikolov

Tommaso Di Noia

Vit Novacek

Damyan Ognyanov

Mirko Orsini

Renato Pajarola

Peter F. Patel-Schneider

Panagiwtis Padiaditis

Rafael Penaloza

Andrew Perez-Lopez

Laura Po

Antonella Poggi

Steven Poltrock

Livia Predoiu

Valentina Presutti

Guilin Qi

Paraskevi Raftopoulou

Cartic Ramakrishnan

Tim Redmond

Doug Reid

Christoph Ringelstein

Eric Ringger

Luis Rodrigo

Marco Rospocher

Daniel Rubin

Marco Ruzzi

Carsten Saathoff

Antonio Sala

Brahmananda Sapkota

Leo Sauermann

Simon Scerri

François Scharffe

Simon Schenk

Anne Schlicht

Christoph Schmitz

Alex Schutz

Arash Shaban-Nejad

Pavel Shvaiko

Heiko Stoermer

William Sunna

Boontawee Suntisrivaraporn

Vlad Tanasescu

Matthew Taylor

Giorgio Terracina

Yannis Theoharis

Christopher Thomas

Alessandra Toninelli

Panayiotis Tritakis

Tania Tudorache

Matteo Vasirani

Tony Veale

Luis Vilches

Raphael Volz

Yimin Wang

Katy Wolstencroft

Huiyong Xiao

Wei Xing

Baoshi Yan

Jung-Jin Yang

Valentin Zacharias

Xiao Zhang

Jun Zhao

Antoine Zimmermann



## Sponsors

### Emerald Sponsor

Saltlux Inc.

### Gold Sponsors

Google Inc.

u-MTec Co.

Korea Advanced Institute of Science and Technology KAIST

VULCAN Inc.

Korea e-business association Koeb

### Silver Sponsors

IBM Corp.

NeOn Project

EASTWEB Project

AsiaLink Program

Electronics and Telecommunications Research Institute ETRI

Franz Inc.



# Table of Contents

## Research Track

Enabling Advanced and Context-Dependent Access Control in RDF Stores .....	1
Automatically Composing Data Workflows with Relational Descriptions and Shim Services .....	15
COMM: Designing a Well-Founded Multimedia Ontology for the Web .....	30
How Service Choreography Statistics Reduce the Ontology Mapping Problem .....	44
Kernel Methods for Mining Instance Data in Ontologies .....	58
An Ontology Design Pattern for Representing Relevance in OWL .....	72
Logical Foundations of (e)RDF(S): Complexity and Reasoning .....	86
Scalable Cleanup of Information Extraction Data Using Ontologies .....	100
A Cognitive Support Framework for Ontology Mapping .....	114
Making More Wikipedians: Facilitating Semantics Reuse for Wikipedia Authoring .....	128
CLOnE: Controlled Language for Ontology Editing .....	142

Creating a Dead Poets Society: Extracting a Social Network of Historical Persons from the Web .....	156
.....	
OBO and OWL: Leveraging Semantic Web Technologies for the Life Sciences .....	169
.....	
History Matters: Incremental Ontology Reasoning Using Modules .....	183
.....	
SALT: Weaving the Claim Web .....	197
.....	
YARS2: A Federated Repository for Querying Graph Structured Data from the Web .....	211
.....	
Discovering Simple Mappings Between Relational Database Schemas and Ontologies .....	225
.....	
Potluck: Data Mash-Up Tool for Casual Users .....	239
.....	
An Empirical Study of Instance-Based Ontology Matching .....	253
.....	
Finding All Justifications of OWL DL Entailments .....	267
.....	
How Useful Are Natural Language Interfaces to the Semantic Web for Casual End-Users? .....	281
.....	
The Fundamentals of iSPARQL: A Virtual Triple Approach for Similarity-Based Semantic Web Tasks .....	295
.....	
Conjunctive Queries for a Tractable Fragment of OWL 1.1 .....	310
.....	
Continuous RDF Query Processing over DHTs .....	324
.....	

Bringing Semantic Annotations to Web Services: OWL-S from the SAWSDL Perspective.....	340
Ontology-Based Controlled Natural Language Editor Using CFG with Lexical Dependency .....	353
Web Search Personalization Via Social Bookmarking and Tagging .....	367
Measuring Incoherence in Description Logic-Based Ontologies .....	381
Alternating-Offers Protocol for Multi-issue Bilateral Negotiation in Semantic-Enabled Marketplaces.....	395
An Event-Based Approach for Semantic Metadata Interoperability .....	409
Evaluating the Semantic Web: A Task-Based Approach .....	423
Using Tableau to Decide Expressive Description Logics with Role Negation .....	438
Instance Migration in Heterogeneous Ontology Environments .....	452
ALLRIGHT: Automatic Ontology Instantiation from Tabular Web Documents .....	466
A Caching Mechanism for Semantic Web Service Discovery .....	480
A Method for Recommending Ontology Alignment Strategies .....	494
Lifecycle-Support in Architectures for Ontology-Based Information Systems .....	508

Ontology-Based Interpretation of Keywords for Semantic Search .....	523
<i>Ontology-Based Interpretation of Keywords for Semantic Search</i> .....	
RDFSync: Efficient Remote Synchronization of RDF Models .....	537
<i>RDFSync: Efficient Remote Synchronization of RDF Models</i> .....	
Sindice.com: Weaving the Open Linked Data .....	552
<i>Sindice.com: Weaving the Open Linked Data</i> .....	
SPARQL++: An Ontology Combination Algorithm .....	566
<i>SPARQL++: An Ontology Combination Algorithm</i> .....	
PORE: Positive-Only Relation Extraction from Wikipedia Text .....	580
<i>PORE: Positive-Only Relation Extraction from Wikipedia Text</i> .....	
Ontology Performance Profiling and Model Examination: First Steps ...	595
<i>Ontology Performance Profiling and Model Examination: First Steps</i> ...	
The Semantic Web and Human Inference: A Lesson from Cognitive Science .....	609
<i>The Semantic Web and Human Inference: A Lesson from Cognitive     Science</i> .....	
From Web Directories to Ontologies: Natural Language Processing Challenges .....	623
<i>From Web Directories to Ontologies: Natural Language Processing     Challenges</i> .....	
On the Foundations of Computing Deltas Between RDF Models .....	637
<i>On the Foundations of Computing Deltas Between RDF Models</i> .....	
Semplore: An IR Approach to Scalable Hybrid Query of Semantic Web Data .....	652
<i>Semplore: An IR Approach to Scalable Hybrid Query of Semantic Web     Data</i> .....	
Semantic Cooperation and Knowledge Reuse by Using Autonomous Ontologies .....	666
<i>Semantic Cooperation and Knowledge Reuse by Using Autonomous     Ontologies</i> .....	
An Unsupervised Model for Exploring Hierarchical Semantics from Social Annotations .....	680
<i>An Unsupervised Model for Exploring Hierarchical Semantics from     Social Annotations</i> .....	
SPARK: Adapting Keyword Query to Semantic Search .....	694
<i>SPARK: Adapting Keyword Query to Semantic Search</i> .....	

## Semantic Web In-Use Track

Unlocking the Potential of Public Sector Information with Semantic Web Technology .....	708
<i>Yves Fassin, Jean-François Fassin, and Jean-François Fassin</i>	
DBpedia: A Nucleus for a Web of Open Data .....	722
<i>Markus Lischke, Daniel Gligor, and Daniel Gligor</i>	
A Semantic Case-Based Reasoning Framework for Text Categorization .....	736
<i>Christoph Bock, Christoph Bock, and Christoph Bock</i>	
Purpose-Aware Reasoning About Interoperability of Heterogeneous Training Systems .....	750
<i>Christoph Bock, Christoph Bock, and Christoph Bock</i>	
A Collaborative Semantic Web Layer to Enhance Legacy Systems .....	764
<i>Christoph Bock, Christoph Bock, and Christoph Bock</i>	
HealthFinland—Finnish Health Information on the Semantic Web .....	778
<i>Christoph Bock, Christoph Bock, and Christoph Bock</i>	
Spatially-Augmented Knowledgebase .....	792
<i>Christoph Bock, Christoph Bock, and Christoph Bock</i>	
Recipes for Semantic Web Dog Food—The ESWC and ISWC Metadata Projects .....	802
<i>Christoph Bock, Christoph Bock, and Christoph Bock</i>	
Matching Patient Records to Clinical Trials Using Ontologies .....	816
<i>Christoph Bock, Christoph Bock, and Christoph Bock</i>	
Application of Ontology Translation .....	830
<i>Christoph Bock, Christoph Bock, and Christoph Bock</i>	
Ontology-Based Information Extraction for Business Intelligence .....	843
<i>Christoph Bock, Christoph Bock, and Christoph Bock</i>	

EIAW: Towards a Business-Friendly Data Warehouse Using Semantic Web Technologies ..... 857

**Semantic Web Challenge**

GroupMe! - Where Semantic Web Meets Web 2.0 ..... 871

CHIP Demonstrator: Semantics-Driven Recommendations and Museum Tour Generation ..... 879

Semantics-Based Framework for Personalized Access to TV Content: The iFanzly Use Case ..... 887

Revyu.com: A Reviewing and Rating Site for the Web of Data ..... 895

Potluck: Semi-Ontology Alignment for Casual Users ..... 903

**Doctoral Consortium**

Semi-automatic Ontology Engineering Using Patterns ..... 911

From Texts to Structured Documents: The Case of Health Practice Guidelines ..... 916

Knowledge Enhanced Searching on the Web ..... 921

Ontology-Driven Management of Semantic Spaces ..... 926

Ontology Mapping: An Information Retrieval and Interactive Activation Network Based Approach ..... 931

Probabilistic Information Integration and Retrieval in the Semantic Web ..... 936

OWL-DL as a Power Tool to Model Negotiation Mechanisms with Incomplete Information .....	941
Cultural Adaptivity for the Semantic Web .....	946
Lightweight Community-Driven Ontology Evolution .....	951
Exploiting Patterns in Ontology Mapping .....	956
Integrating Uncertainty into Ontology Mapping .....	961
Mutual Contextualization in Tripartite Graphs of Folksonomies .....	966
<b>Author Index</b> .....	971



# Enabling Advanced and Context-Dependent Access Control in RDF Stores

Fabian Abel<sup>1</sup>, Juri Luca De Coi<sup>2</sup>, Nicola Henze<sup>1</sup>,  
Arne Wolf Koesling<sup>2</sup>, Daniel Krause<sup>1</sup>, and Daniel Olmedilla<sup>2</sup>

<sup>1</sup> Distributed Systems Institute (KBS), University of Hannover, Hannover, Germany  
{abel,henze,krause}@kbs.uni-hannover.de

<sup>2</sup> L3S Research Center and University of Hannover, Hannover, Germany  
{decoi,koesling,olmedilla}@L3S.de

**Abstract.** Semantic Web databases allow efficient storage and access to RDF statements. Applications are able to use expressive query languages in order to retrieve relevant metadata to perform different tasks. However, access to metadata may not be public to just any application or service. Instead, powerful and flexible mechanisms for protecting sets of RDF statements are required for many Semantic Web applications. Unfortunately, current RDF stores do not provide fine-grained protection. This paper fills this gap and presents a mechanism by which complex and expressive policies can be specified in order to protect access to metadata in multi-service environments.

## 1 Introduction

The Semantic Web vision requires that existing data is provided with machine-understandable annotations. These annotations (commonly referred to as *metadata*) are meant to facilitate tasks such as data sharing and integration. However, it is often the case that information cannot be shared unconditionally: many Semantic Web applications require to control when, what and to whom information is disclosed. Nevertheless, existing metadata stores do not support access control, or their support is minimal (e.g., protection may apply only to the database as a whole and not to the data it contains). On the one hand, access control could be embedded within the metadata store: in this case the access control mechanism would be repository-dependent and not portable across different platforms. On the other hand, a more general solution would be adding a new component on top of the metadata store in charge of access control-related issues. Still this second approach requires to face problems which are not trivial, since the obvious solution of filtering out private triples from the results is not possible. The reason is that those triples may not be known in advance, as it happens when the result of the query consists of triples not previously available in the metadata store.

Furthermore, the Semantic Web envisions that interactions can be performed between any two entities, even if they did not carry out common transactions in the past, making unsuitable traditional identity-based access control mechanisms. Therefore, more advanced mechanisms (based e.g., on properties of the

requester) are required. Semantic policy languages lately emerged in order to address these requirements: they provide the ability to specify complex conditions such as time constraints and may even provide an interface to query external packages such as other repositories. However, evaluating such constraints for each triple to be potentially returned by the metadata store is not affordable, since it is too expensive in terms of time.

In this paper we present an architecture that integrates advanced access control mechanisms based on Semantic Web policies with different kinds of RDF metadata stores. Given an RDF query, our framework partially evaluates all applicable policies and constraints the query according to the result of such evaluation. The modified query is then sent to the RDF store which executes it like an usual RDF query. Our framework enforces fine-grained access control at triple level, i.e., all triples returned as a response to the query can be disclosed to the requester according to the policies in force.

The rest of the paper is organized as follows: §2 presents an scenario in order to motivate the need for flexible access control mechanisms over semantic data. Different approaches and related work are described in §3. In §4 we introduce how a policy engine can be integrated on top of an RDF store in order to restrict access at the RDF triple level. §5 presents our current implementation which is used in order to perform a set of experiments, which estimate the impact of this approach and are included in this section too. Finally, §6 concludes the paper and outlines our future work.

## 2 Access Control in RDF Stores

The Personal Reader [1] is a distributed multi-service and multi-user environment. In the Personal Reader Framework there exist many modular applications divided into (i) Personalization Services for accessing and personalizing Semantic Web data sources (ii) Syndication Services for aggregating and processing data provided by other services and (iii) User Interfaces which can be accessed by different users. A remarkable feature of the Personal Reader Framework is its plug-and-play nature, i.e., new Personalization and Syndication Services can be plugged into the system and immediately used by currently available services and users.

Enabling behavior and content adaptation in different applications requires the use of a shared user profile. Such a user profile is in charge of storing semantic data from different services, application domains, and users. RDF databases have been chosen to store these metadata, since they provide efficient access and high flexibility: arbitrary RDF data referring to various ontologies can be stored within the RDF database.

Different services may store in or require sensitive data from the user profile in the RDF repository. It is crucial for the user to be able to specify which (kinds of) services are allowed to access and retrieve which part of the data stored in the user profile. For example, Alice must be able to allow a recommendation service to access information about her friends but not her private information

(e.g., address or telephone number). Similarly, a means needs to be provided to allow Bob to restrict access to his health-related data only to his medical service. Health-related data may be defined as instances of a class `Health` in some ontology, and the medical service may need to identify itself by providing some credential.

The most part of current RDF databases provide none or very simple security mechanisms. For example, one of today's most widespread RDF database management systems, `OpenRDF` [2], allows to define access rights only for a whole database. Therefore, a more fine-grained solution is required, in which access to smaller fragments of data (e.g., RDF triples) can be checked at query time. Furthermore, the result of the check may depend on conditions unrelated to the data to be accessed (e.g., time, location, conditions), such as properties of the requester (possibly to be certified by credentials) or environmental factors (e.g., time of the request). Ideally, a good solution should be expressive and flexible while at the same time not excessively affecting the response time.

### 3 Related Work

The problem we focus on in this paper is how to restrict access to RDF data. One way to address this problem is defining subsets of an RDF database which can be accessed by some requester. For example, Named Graphs [3] can be used to evaluate SPARQL queries [4] based on allowed RDF graphs [5] or in combination with TriQLP [6] which allows the formulation of trust-policies, in order to answer graph-based queries. Those queries describe conditions under which suiting data should be considered trustworthy. Access control based on identity could be performed if all requesters and their allowed graphs were known in advance. However, this is not the case in our scenario presented above and since access to data may be (not) allowed depending on contextual conditions, these approaches do not apply: on the one hand named graphs cannot be statically precomputed for each possible combination of environmental factors, since their number would be too big; on the other hand named graphs cannot be created at runtime, since the creation process would excessively slow down the response time. Furthermore, the plug-and-play nature of the Personal Reader Framework as well as the possibility that services dynamically change the RDF database itself by adding or removing data from the user profile would significantly complicate managing such named graphs.

[7] defines simple rule-based policies over the RDF database: such policies describe subgraphs on which actions like `read` and `write` can be executed: subgraphs are identified by specifying graph patterns. Some approaches also respect RDF Schema entailments [8]. However, all these approaches require to instantiate the graph patterns, i.e., to generate one graph for each policy and execute the given query on each graph, hence leading to longer response times. Furthermore, these approaches cannot be applied to contextual queries either.

Finally, many policy languages (e.g., KAOS [9], Rei [10], PeerTrust [11] or Protune [12]) allow to express access authorizations on the Semantic Web by

means of policies. However, none of them describes how to integrate policies in RDF databases.

## 4 Policy-Based Query Expansion

As shown in §3, existing work on RDF data protection does not suit the requirements of dynamic Semantic Web environments such as the Personal Reader Framework presented in §2. Available solutions do not handle contextual information in a proper way, as they either require a large amount of memory or unacceptably increase the response time. Filtering returned results is not an adequate solution, either: current RDF query languages allow to arbitrarily structure the results, as shown in the following example<sup>1</sup>.

```
CONSTRUCT {CC} news:isOwnedBy {User}
  FROM {User} ex:hasCreditCard {CC}; foaf:name {Name}
  WHERE Name = 'Alice'
```

Post-filtering the results of a query is hence not straightforward whenever their structure is not known in advance. It could be possible to break constructs queries into a select query and the generation of the returned graph (construct), therefore avoiding this problem. However, the query response time may be considerably too large since this approach cannot make use of repository optimizations and policies are enforced after all data (allowed and not allowed) has been retrieved. As an example, suppose an unauthorized requester submits a query asking for all available triples in the store. A post-filtering approach would retrieve all triples first and then filter them all out.

To address these problems we decided to enforce access control as a layer on top of RDF stores (also making our solution store independent). Our strategy is to pre-evaluate the contextual conditions of the policies, which do not depend on the content of the RDF store. Then, we expand the queries before they are sent to the database, therefore integrating the enforcement of the rest of (metadata) conditions with the query processing, thereby restricting the queries in such a way that they only utilize allowed RDF statements. This way, policies can hold a greater expressiveness and support both metadata and contextual conditions, while pushing part of its enforcement to the highly optimized query evaluation of the RDF store. This approach allows to include more complex conditions without dramatically increasing the overhead produced by policy evaluation, and while relying on the underlying RDF store to evaluate RDF Schema capabilities (as discussed in §8).

---

<sup>1</sup> Our examples use SeRQL [13] syntax (and for simplicity we do not include the namespace definitions), since SeRQL is the language we exploit in our implementation. However the ideas behind our solution are language-independent and can be applied to other RDF query languages.

## 4.1 RDF Queries

We assume disjoint, infinite sets  $I$ ,  $B$ , and  $L$ , which denote IRIs, blank nodes and literals. In addition, let  $Pred$ ,  $Const$  and  $Var$  be mutually disjoint sets of predicates, constants and variables such that  $Const = I \cup B \cup L$ . Then (using similar notation as in [14]) an RDF graph is a finite set of triples  $I \cup B \times I \times Const$ .

In the following we assume a query language with queries having the following structure (§6.19 in [15], [2], [3]):

SELECT/CONSTRUCT  $RF$  FROM  $PE$  WHERE  $BE$

where

- $RF$  is the result form, either a set of variables (projection in select queries) or a set of triples (construct clause in construct queries).
- $PE$  is a path expression as defined below.
- $BE$  is a boolean expression, that is, a string<sup>4</sup> representing a set of constraints in the form of (in)equality binary predicates and numerical operators such as greater than or lower than, connected by boolean connectives (AND and OR).

and a query will be denoted as  $q = (RF, PE, BE)$ . As today’s established RDF query languages like SerQL [13] or SPARQL [4] do not support  $\dots$  or  $\dots$  operations yet, we focus on common  $\dots$  operations. An example query is provided in Figure 1. Without access control enforcement, this query would return an RDF graph containing all RDF triples matching the graph pattern defined in the FROM block, i.e., the query answer would include identifier and name of a person, her phone number(s) and the document(s) she is interested in.

We define a path expression as a triple  $(s, p, o)$  such that  $s \in I \cup B \cup Var$ ,  $p \in I \cup Var$  and  $o \in Const \cup Var$ . Hereafter we will use  $(s, p, o)$  and  $triple(s, p, o)$  ( $triple \in Pred$ ) as synonyms. In addition, given an expression  $E$  (result form, path or boolean expression), we will denote by  $vars(E)$  the set of all unbound variables occurring in  $E$ .

**Definition 1.**  $disunify(e, \theta) = \{(s', p, o') \mid (s, p, o) \in e, (s, p, o) \in \theta, (s', p, o') \in BE\}$

- $$- \begin{cases} s' = v_s \quad be_s = (v_s = s) & \text{if } s \in I \cup B \\ s' = v_s \quad be_s = (v_s = Value) & \text{if } s \in Var, [s = Value] \in \theta \\ s' = s \quad be_s = \varepsilon & \text{if } s \in Var \end{cases}$$

<sup>2</sup> Although our examples will use the syntax of the SerQL query language, the results of this paper apply also to other languages with similar structure (e.g., SPARQL [4]).

<sup>3</sup> Extending our algorithm to support UNION and INTERSECTION operators is straightforward. The union (resp. intersection) of two queries would be expanded into the union (resp. intersection) of the two expanded queries.

<sup>4</sup> In the rest of the paper we also use  $BE$  to represent a set of boolean expressions. The exact meaning will be clear from the context.



**Table 1.** Example of high-level policies controlling access to RDF statements

No.	Policy
<i>pol<sub>1</sub></i>	deny access to triples (X, foaf:phone, Z) IF (X, foaf:currentProject, l3s:reverse) AND Requester = 'RecommenderService'.
<i>pol<sub>2</sub></i>	allow access to triples (X, foaf:phone, Z) IF Requester = Service AND Service is a trusted service AND (l3s:alice, foaf:knows, X).
<i>pol<sub>3</sub></i>	allow access to triples (l3s:alice, foaf:phone, Z).
<i>pol<sub>4</sub></i>	allow access to triples (X, Y, Z) IF Time is the current time AND 09:00 < Time AND Time < 17:00 AND Y = foaf:name AND X != l3s:tom.
<i>pol<sub>5</sub></i>	allow access to triples (l3s:alice, foaf:interest, Z) IF (Z, rdf:type, foaf:Document) AND (X, foaf:currentProject, P) AND (Z, foaf:topic, T) AND (P, foaf:topic, T).

3. anyone can receive Alice's phone number.
4. RDF statements containing  $\dots$  of entities different from Alice's boss Tom can be accessed during work time
5. the last policy controls access to Alice's interests. Only interests related to her current project(s) can be accessed.

Many algorithms could be exploited in order to evaluate policies and to handle conflicts which arise whenever two different policies allow and deny access to the same resource. However such algorithms are out of the scope of this paper. Therefore, in the following we assume a simple policy evaluation algorithm like the following one:

```

if a  $\dots$  policy is applicable then access to the triple is denied
else if an  $\dots$  policy is applicable then access to the triple is allowed
else access to the triple is denied ( $\dots$ )

```

More advanced algorithms exploiting priorities or default precedences [10] among policies could be used as well.

### 4.3 Policy Evaluation and Query Expansion

Given an RDF query, each RDF statement matching a pattern specified in the FROM block is accessed and, if the policies in force allow it, returned. Our approach consists of analyzing the set of RDF statements to be accessed and restricting it according to the policies in force. Contextual conditions (e.g., time constraints and conditions on properties of the requester) are evaluated by some policy engine, whereas other constraints are added to the given query and enforced during query processing.

**Definition 2 (Policy applicability).**

$$\begin{aligned}
 & P \text{ is applicable to } (e, \Sigma, \text{ff}) \text{ if } \exists \text{ff} \text{ such that } \text{ff} \text{ satisfies:} \\
 & \text{ff} = \widehat{\text{pol}(e)} \\
 & - \sigma' = \text{mgu}(e, H^T(\text{pol})) \\
 & - \exists \sigma, \sigma'' : \sigma = \sigma' \sigma'' \wedge \forall cp_i \in B(\text{pol}), P \cup \Sigma \models \sigma cp_i \\
 & - \exists be_i \in B(\text{pol}) : \forall pe_i \in B(\text{pol}), \text{vars}(\sigma be_i) \cap (\text{vars}(\sigma pe_i) \cup \text{vars}(\sigma e)) = \emptyset \Rightarrow \\
 & P \cup \Sigma \models \sigma be_i
 \end{aligned}$$

$$\text{ff} = (e, \text{pol}) \xrightarrow{P, \Sigma} (PE, BE) \text{ where } PE = \{pe_i \mid pe_i \in B(\text{pol}), pe_i \neq \text{ff}\}$$

$$\text{ff} = \text{disunify}(pe_i, \theta) = (pe'_i, BE')$$

- $PE = \{pe'_i \mid pe_i \in B(\text{pol}), pe'_i \neq pe_i\}$
- $BE = \{\sigma be_i \mid be_i \in B(\text{pol}) \wedge \exists pe_i : \text{vars}(\sigma be_i) \cap (\text{vars}(\sigma pe_i) \cup \text{vars}(\sigma e)) \neq \emptyset\}$
- $BE = BE' \cup \{\sigma_i \mid \sigma_i = [X = Y] \wedge (X \in \text{Const} \vee Y \in \text{Const})\}$

Intuitively, a policy  $pol$  is applicable to  $e$  if the triple the policy is protecting unifies with the path expression and all the contextual predicates and bound boolean expressions (or those not dependent of metadata expressions in the body of the policy) are satisfied. The return value is a set with the path expressions found in the body of the policy and all extracted boolean expressions which have not been evaluated and relate to the path expressions found.

Following our example, assuming contextual predicates are satisfied, then  $pol_4$  is applicable to  $(Person, foaf : name, Name)$  and returns  $(\emptyset, \{[Person = l3s : tom]\})$ . In addition,  $pol_1$ ,  $pol_2$  and  $pol_3$  are applicable to  $(Person, foaf : phone, Phone)$  and returns  $(\{Var8, foaf : currentProject, Var9\}, \{[Var8 = Person], [Var9 = l3s : reverse]\})$ ,  $(\{(Var1, foaf : knows, Var2)\}, \{[Var1 = l3s : alice], [Var2 = Person]\})$  and  $(\emptyset, [Person = l3s : alice])$  respectively.

Before we describe the query expansion algorithm, and for sake of clarity, we describe the conditions under a query does not need to be evaluated since the result is empty. Intuitively, a query fails if there does not exist any allowed triple to be returned according to both the query and the applicable policies, that is if there exist a path expression for which no allowed triples exist (disallow by default) or if there exist a path expression for which a policy (which does not depend on path expressions) specifies that no triple is allowed (explicit disallow).

**Definition 3.**  $q = (RF, PE, BE)$  is applicable to  $(P, \Sigma)$  if:

- $\exists e \in PE : \nexists pol \in P, H(\text{pol}) = \text{allow}(T) \wedge \widehat{\text{pol}(e)}$
- $\exists e \in PE : \exists pol \in P, H(\text{pol}) = \text{disallow}(T) \wedge \text{pol}(e) \wedge e, \text{pol} \xrightarrow{P, \Sigma} (\emptyset, \emptyset)$

Let's denote by  $\text{append}(BE, Conn)$  (resp.  $\text{prefix}(BE, Conn)$ ) a function that given a set of boolean expressions  $BE$  and a connective (e.g., AND or OR) returns a new boolean expression in which all the elements of  $BE$  are enclosed by brackets and connected (resp. prefixed) by  $Conn$ . The pre-filtering algorithm is defined as follows:



Input:

a query  $q = (RF, PE, BE)$ , a set of policy rules  $P$  and a state  $\Sigma$

Output:

$PE_{new}^+ \equiv$  new optional path expressions (from allow policies)  
 $PE_{new}^- \equiv$  new optional path expressions (from disallow policies)  
 $BE_{new}^+ \equiv$  conjunction of boolean expressions (from allow policies)  
 $BE_{new}^- \equiv$  conjunction of boolean expressions (from disallow policies)

*policy\_prefiltering*( $q, P$ ):

$BE_{or}^+ \equiv$  disjunction of boolean expressions (from allow policies)  
 $BE_{or}^- \equiv$  disjunction of boolean expressions (from disallow policies)  
 $P_{app} \equiv$  a set of applicable policies

$PE_{new}^+ = PE_{new}^- = \emptyset$

$\forall e \in PE$

$BE_{or}^+ = BE_{or}^- = \emptyset$

// check allow policies

$P_{app} = \{pol \mid pol \in P \wedge H(pol) = allow(T) \wedge \widehat{pol}(e)\}$

if  $P_{app} = \emptyset$

return query failure // no triples matching  $e$  are allowed

if  $\exists pol \in P_{app} : e, pol \xrightarrow{P, \Sigma} (\emptyset, \emptyset)$

// all triples matching  $e$  are allowed without restrictions

else

$\forall pol \in P_{app}$

$e, pol \xrightarrow{P, \Sigma} (PE', BE')$

if  $PE' = \emptyset$

$BE_{or}^+ \cup = append(BE', 'AND')$

else if  $\exists \theta, \widetilde{PE} \in PE_{new}^+ : \theta = mgu(\widetilde{PE}, PE')$

$BE_{or}^+ \cup = append(\theta BE', 'AND')$

else

$PE_{new}^+ \cup = PE'$

$BE_{or}^+ \cup = append(BE', 'AND')$

$BE_{new}^+ \cup = append(BE_{or}^+, 'OR')$

// check disallow policies

$P_{app} = \{pol \mid pol \in P \wedge H(pol) = disallow(T) \wedge \widehat{pol}(e)\}$

if  $\exists pol \in P_{app} : e, pol \xrightarrow{P, \Sigma} (\emptyset, \emptyset)$

return query failure // all triples matching  $e$  are denied

$\forall pol \in P_{app}$

$e, pol \xrightarrow{P, \Sigma} (PE', BE')$

if  $PE' = \emptyset$

$BE_{or}^- \cup = append(BE', 'AND')$

else if  $\exists \theta, \widetilde{PE} \in PE_{new}^- : \theta = mgu(\widetilde{PE}, PE')$

$BE_{or}^- \cup = append(\theta BE', 'AND')$

else

$PE_{new}^- \cup = PE'$

$BE_{or}^- \cup = append(BE', 'AND')$

$BE_{new}^- \cup = append(BE_{or}^-, 'OR')$

Let  $copy(RF, PE)$  be a function that copies (replacing previous content) into  $RF$  either the variables (for SELECT queries) or the path expressions (for CONSTRUCT queries) from  $PE$ .

Input:

a query  $q = (RF, PE, BE)$

$PE_{new}^+ \equiv$  new optional path expressions (from allow policies)

$PE_{new}^- \equiv$  new optional path expressions (from disallow policies)

$BE_{new}^+ \equiv$  conjunction of boolean expressions (from allow policies)

$BE_{new}^- \equiv$  conjunction of boolean expressions (from disallow policies)

Output:

an expanded query  $q = (RF^+, PE^+, BE^+) \text{ MINUS } (RF^-, PE^-, BE^-)$

$expandQuery(q, PE_{new}^+, PE_{new}^-, BE_{new}^+, BE_{new}^-)$

$RF^+ = RF^- = copy(RF, PE)$

$PE^+ = PE \cup prefix(PE_{new}^+, 'OPT')$

$PE^- = PE \cup prefix(PE_{new}^-, 'OPT')$

$BE^+ = BE \cup append(BE_{new}^+, 'AND')$

$BE^- = BE \cup append(BE_{new}^-, 'AND')$

where the connective 'OPT' represents the "optional path expression" modifier in the chosen query language (e.g., '[' and ']' in [SeRQL \[15\]](#)).

Briefly, the algorithm extracts the new path expressions found in the body of the policy rules. It extracts their variable bindings in order to reuse them in case they appear in more than one policy rule. However, if the same path expression is found in policies being applied to different from clauses, then they cannot be reused (since conditions on different expressions are connected conjunctively). After prefiltering each policy, a set of AND boolean expressions are extracted. The set of all boolean expressions from applicable allow policies to one from clause are connected by OR. The set of all boolean expressions applicable to different from clauses are connected by AND. From that query we have to remove the triples affected by disallow policies, which are specified in a similar fashion and added to the query using the MINUS operator.

The result of applying the above algorithm to the query in [Figure 1](#) and the policies in [Table 1](#) (assuming that time is 15:00, the requester is 'RecommenderService' and it is trusted) is shown in [Figure 2](#).

## 5 Implementation and Experiments

In this section we present the implementation of the ideas (and algorithm) presented in previous sections in order to transform an arbitrary RDF repository into an *SPARQL query engine*. We first briefly describe the general architecture and evaluate the implementation in terms of scalability afterwards.

### 5.1 Architecture

Our implementation adds an additional layer on top of an arbitrary RDF repository (therefore being suitable for reusability among different ones). Incoming

```

CONSTRUCT {Person} foaf:name {Name};
           foaf:phone {Phone}; foaf:interest {Document}
FROM {Person} foaf:name {Name};
     foaf:phone {Phone}; foaf:interest {Document}
  [ {Var1} foaf:knows {Var2} ]
  [ {Var3} rdf:type {Var4}, {Var3} foaf:topic {Var5},
    {Var6} foaf:currentProject {Var7}, {Var7} foaf:topic {Var5} ]
WHERE ( ( (Person != 13s:tom) ) ) AND
      ( ( (Var2 = Person) AND (Var1 = 13s:alice) ) OR
        (Person = 13s:alice) ) ) AND
      ( ( (Var3 = Document) AND (Var2 = Person) AND
        (Person = 13s:alice) AND (Var4 = foaf:Document) ) ) )

MINUS
CONSTRUCT {Person} foaf:name {Name};
           foaf:phone {Phone}; foaf:interest {Document}
FROM {Person} foaf:name {Name};
     foaf:phone {Phone}; foaf:interest {Document}
  [ {Var8} foaf:currentProject {Var9} ]
WHERE ( ( (Var8 = Person) AND (Var9 = 13s:reverse) ) )

```

Fig. 2. Expanded RDF query

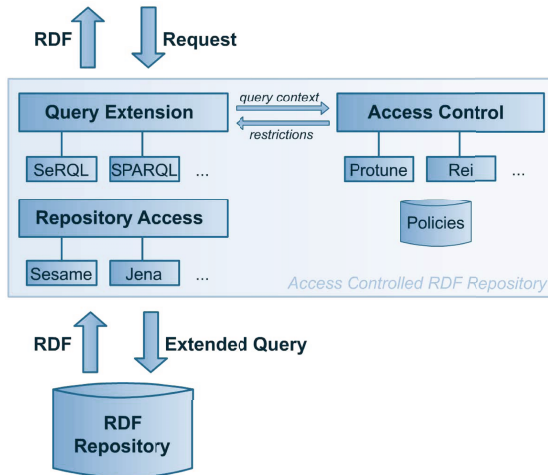


Fig. 3. Architecture - Access Controlled RDF Repository

queries are first processed and extended according to access control policies before they are directed to the underlying RDF repository.

The Architecture of our implementation, illustrated in Figure 3, is composed of three main modules: *Query Extension*, *Access Control* and *Repository Access*.

**Query Extension.** The main task of this core module is to rewrite a given query in a way that only allowed RDF statements are accessed and returned.

It is in charge of querying the policy engine for each FROM clause of the original query in order and expand it with the extra path expressions and constraints (cf. §4.3). Our initial implementation provides query extension capabilities for the [\[13\]](#) query language.

**Policy Engine.** This module is responsible for the policy (partial) evaluation. Input information (e.g., requester or disclosed credentials) may be used as well. In the actual implementation we use the [\[12\]](#) policy language [\[12\]](#) and its framework.

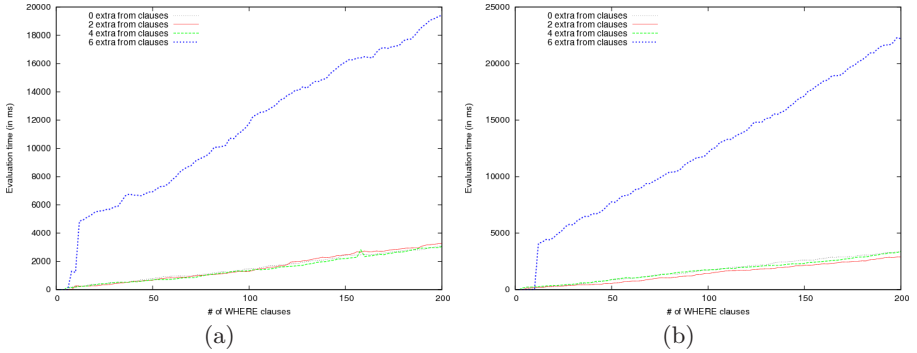
**RDF Repository.** After extending a query the extended query can be passed to the underlying RDF repository. Since our solution is repository-independent, any store supporting SeRQL, such as [\[2\]](#) (which we integrated in our actual implementation), can be used. The result set returned contains only allowed statements and can be directly returned to the requester.

## 5.2 Experiments and Evaluation

We set up a Sesame database with more than 3,000,000 RDF statements about persons and their exchanged e-mails into an AMD Opteron 2.4GHz with 32GB memory and send the queries through the network from a Dual Pentium 3.00GHz with 2GB. We checked our approach in the “worst” scenario by setting an initial query returning a very large number of results (1,280,000 in this case). We then automatically generated extra path and boolean expressions, extracted from policies as described in §4.3, which were added to the original query. Since we wanted to test the impact of allow and disallow policies in our algorithm, we evaluated both expansion options: with and without the MINUS operator. All these queries were executed and we measured the time needed for its evaluation in order to see its impact. The results are shown in Figure [4](#).

Both graphs show how adding many WHERE clauses (extracted from allow policies) increases linearly the evaluation time. The reason for this increasement is a) each WHERE clause specifies new triples that are allowed to be accessed and as a consequence, the number of triples that will be returned and b) the new added clauses require time for its evaluation. The reason for the strong increase when adding 6 FROM clauses is that the new clauses produced a triplication in the number of triples to consider, even though none of the new ones were to be returned (so we believe that appropriate optimizations in the repository would help to reduce this impact too). We also made other experiments where the initial query was more selective and the addition of FROM clauses produced only linear increase on the evaluation time.

These results demonstrate that the approach described in this paper scales to a larger number of policies, especially when policies specify only boolean constraints or when the path expressions are selective, since the cost might well be accepted in order to provide fine-grained access control. For low selective queries, optimizations are required. It is also important to note that even if thousands policies are specified, not all will contribute to expand the original query with new expressions. Only those protecting the triples in the FROM



**Fig. 4.** (a) Response time when increasing the number of FROM and WHERE clauses (with allow policies), and (b) same as before including a MINUS between two queries (allow and disallow policies)

clause of the original query which have satisfied their contextual constraints will be taken into account, therefore reducing the number of applicable policies.

As main conclusions we would highlight that fine-grained access control comes with a cost. However, this cost may be acceptable for semantic web applications and services that must deal with sensitive data. For example, applications that retrieve personal data from the user, as in our Personal Reader scenario, and which use highly selective queries, may benefit from this solution and allow users to define their own policies over their data. In addition, optimizations on the generation of new queries and reordering of constraints as well as natively implemented optimizations in the RDF repository may help to further reduce the response query time.

## 6 Conclusions and Future Work

Semantic Web applications might require to store and access metadata while still preserving the sensitive nature of such data, especially in multi-service and multi-user environments. However, current RDF stores do not provide access control mechanisms that suit this requirement. In this paper we presented an approach (independent of the RDF store used) for the integration of expressive policies in order to provide a fine-grained access control mechanism for RDF repositories. These policies may state conditions on the RDF nature and content of the RDF store as well as other external (e.g., contextual) conditions. The evaluation of the process is divided in order to pre-evaluate conditions of the policy engine not depending on the RDF store and relying on the highly optimized query evaluation of semantic databases for RDF pattern and content constraints. We presented an implementation of those ideas and showed with our evaluation how the cost of this access control layer scales and might be acceptable for applications requiring fine-grained access control over their (possibly sensitive) data.

We are currently investigating other optimizations in order to improve the evaluation of our implementation such as reordering of constraints in a given query as well as caching techniques. In addition, we are applying and implementing this approach to the SPARQL query language.

We thank the anonymous reviewers, Axel Polleres and Piero Bonatti for their feedback and help to improve this paper.

## References

1. Abel, F., Baumgartner, R., Brooks, A., Enzi, C., Gottlob, G., Henze, N., Herzog, M., Kriesell, M., Nejd, W., Tomaschewski, K.: The Personal Publication Reader. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
2. Broekstra, J., Kampman, A., Harmelen, F.: Sesame: A generic architecture for storing and querying RDF and RDF Schema. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, Springer, Heidelberg (2002)
3. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. In: WWW, NY, USA (2005)
4. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF, <http://www.w3.org/TR/rdf-sparql-query/>
5. Dietzold, S., Auer, S.: Access control on RDF triple stores from a semantic wiki perspective. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)
6. Bizer, C., Oldakowski, R.: Using context- and content-based trust policies on the semantic web. In: WWW, NY, USA (2004)
7. Reddivari, P., Finin, T., Joshi, A.: Policy based access control for a RDF store. In: Proceedings of the Policy Management for the Web Workshop. A WWW 2005 Workshop, W3C, May 2005, pp. 78–83 (2005)
8. Jain, A., Farkas, C.: Secure Resource Description Framework: an access control model. In: ACM SACMAT, CA, USA, ACM Press, New York (2006)
9. Uszok, A., Bradshaw, J.M., Jeffers, R., Suri, N., Hayes, P.J., Breedy, M.R., Bunch, L., Johnson, M., Kulkarni, S., Lott, J.: KAoS policy and domain services: Toward a description-logic approach to policy representation, deconfliction, and enforcement. In: POLICY (2003)
10. Kagal, L., Finin, T.W., Joshi, A.: A policy language for a pervasive computing environment. In: POLICY (June 2003)
11. Gavriloaie, R., Nejd, W., Olmedilla, D., Seamons, K.E., Winslett, M.: No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWC 2004. LNCS, vol. 3053, Springer, Heidelberg (2004)
12. Bonatti, P.A., Olmedilla, D.: Driving and monitoring provisional trust negotiation with metapolicies. In: POLICY 2005, Stockholm, Sweden (June 2005)
13. Broekstra, J., Kampman, A.: SeRQL: An RDF query and transformation language. (2004)
14. Polleres, A.: From SPARQL to rules (and back). In: WWW, Banff, Canada (2007)
15. Aduna: The SeRQL query language (revision 1.2), <http://www.openrdf.org/doc/sesame/users/ch06.html>

# Automatically Composing Data Workflows with Relational Descriptions and Shim Services<sup>\*</sup>

José Luis Ambite and Dipsy Kapoor

University of Southern California, Information Sciences Institute  
4676 Admiralty Way, Marina del Rey, CA 90292, USA  
{ambite,dipsy}@isi.edu  
<http://www.isi.edu/~argos>

**Abstract.** Many scientific problems can be represented as computational workflows of operations that access remote data, integrate heterogeneous data, and analyze and derive new data. Even when the data access and processing operations are implemented as web or grid services, workflows are often constructed *manually* in languages such as BPEL. Adding semantic descriptions of the services enables automatic or mixed-initiative composition. In most previous work, these descriptions consists of semantic types for inputs and outputs of services or a type for the service as a whole. While this is certainly useful, we argue that is not enough to model and construct complex data workflows.

We present a planning approach to *automatically* constructing data processing workflows where the inputs and outputs of services are *relational* descriptions in an expressive logic. Our workflow planner uses relational subsumption to connect the output of a service with the input of another. This modeling style has the advantage that adaptor services, so-called *shims*, can be *automatically* inserted into the workflow where necessary.

## 1 Introduction

Much of the work of scientists, economists, and engineers is consumed by accessing, integrating, and analyzing data. Recently, there has been a significant effort to support computational workflows in fields such as physics (e.g., [16]) and bioinformatics (e.g. [22]). This research leverages domain ontologies to facilitate workflow construction, usually by defining the workflow components as semantic web services. Such semantic descriptions are of two kinds: (1) the service as a whole is classified according to an ontology of service types (e.g., [11]), or (2) the inputs and outputs of services are typed with concepts defined in a domain ontology (e.g. [16, 20]). Though useful for service discovery, these approaches do not describe the data manipulated by the service in sufficient detail. First, the inputs and outputs of a service are usually related, so that a service is better described as having relational inputs and outputs instead of a list of apparently independent single-type inputs or outputs. In data-intensive

---

<sup>\*</sup> Work supported by National Science Foundation Award EIA-0306905.

applications tabular data is the norm, so services that process such data must consume and produce tables natively. Second, having relational descriptions of the services' inputs and outputs allows our planner to *automatically* introduce adaptor services, so-called *shims* [10], that transform the output of one service to the input of another. In this paper, we present the Argos framework to (1) describe data processing services and (2) automatically generate new data on demand by automatically composing data processing workflows.

We have applied the Argos approach to a transportation modeling domain [7,2], which we use as an example to ground the discussion. However, our approach is general and can be applied to produce data processing workflows in any other domain, as long as the data and operations are described in a suitable ontology.

As an example consider Figure 1(a) that shows an abstract workflow that computes truck traffic in the highways of the Los Angeles Consolidated Statistical Metropolitan Area (LACMSA). The workflow estimates the *intra-regional* trade based on employment data and an input-output transaction model of the local economy (provided by the Southern California Association of Governments – SCAG), resulting in a table of attractions and productions of different commodity sectors for each traffic analysis zone (TAZ) within the region.<sup>1</sup> To estimate the *inter-regional* trade, the model uses a variety of data sources, including data from the Commodity Flow Survey (CFS) of the US Census Bureau, the Waterborne Commerce of the United States (WCUS), and airport statistics from RAND. The inter-regional attractions and productions per commodity are assigned to the TAZs of the entry/exit points in the region. For example, airborne imports of computer equipment are assigned to the TAZs corresponding to the airports in the region. The intra- and inter-regional attractions and productions are converted to an Origin-Destination matrix between pairs of TAZs using a gravity model. Finally, a network equilibrium algorithm assigns the freight flow to specific highway links. Figure 1(c) shows graphically the final result of the workflow: the flow of freight in the LACMSA highway network.

There are many challenges in producing data processing workflows such as the transportation model described above. Since the data comes from a variety of sources, it may be expressed in different schemas, formats, and units. Therefore, the workflow needs to perform different types of data conversion, for example, to translate between different units (e.g., from tons to dollars to jobs to container units to passenger-car-equivalents), or to translate economic data described in one industry/sector classification to another (e.g., from the North American Industry Classification System – NAICS – to the Standard Classification of Transported Goods – SCTG –, or from the NAICS 1997 version of the standard to the NAICS 2002 version).

The workflow of Figure 1(a) abstracts many details. The full workflow, whose structure appears in Figure 1(b), contains over 50 data access and data processing operations. This estimation model was originally implemented by a combination

---

<sup>1</sup> A TAZ is a spatial region consisting of several census blocks. The LACMSA is partitioned into 3165 TAZs.



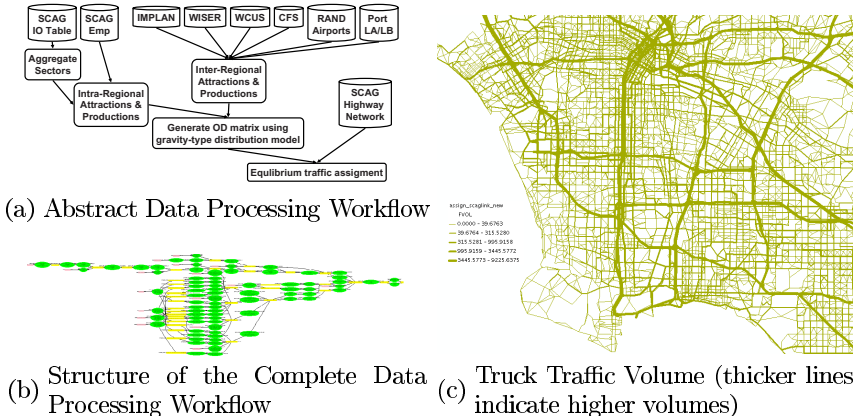


Fig. 1. Estimating Truck Traffic in the Los Angeles Highway Network

of manual steps and custom-designed programs. Our approach *automatically* generates such a data processing workflow in response to a user data request, including all the necessary data integration and translation operations.

The remainder of the paper is structured as follows. First, we describe the domain ontology and the data representation. Second, we present our workflow planning algorithm that uses both domain-dependent and domain-independent services. Third, we provide an empirical evaluation. Fourth, we compare with related work. Finally, we discuss our contributions and plans for future work.

## 2 Domain and Data Modeling

One of the major challenges to automating computational workflows is understanding source data, and data consumed or produced by services. To provide a clear understanding of the semantics of the data, we describe the data according to an ontology of the application domain.

**Domain Ontology.** We represent the domain ontology in the first-order logic language of the PowerLoom system [14,18]. PowerLoom is the more expressive successor of the Loom [13] description logic. PowerLoom is specially optimized to compute both concept and relation subsumption. First-order logical inference is undecidable, hence PowerLoom is incomplete. Nevertheless, in our experience we have defined an expressive domain ontology and PowerLoom proves the required inferences efficiently. However, the techniques we present in this paper rely only on relational subsumption, i.e., query containment, so any other knowledge representation formalism from relational conjunctive queries [8] to description logics such as DLR [9] could be applied.

Sample concept, instance, rule and constraint definitions of our transportation modeling ontology appear in Figure 2. The Flow concept represents a transfer

of a product between two geospatial areas using a transportation mode measured during a particular time interval. For example, an instance of `Flow` is the domestic exports by `air` (a `TransportationMode`) of Pharmaceutical products from LACMSA in 2000, which amounts to (`hasValue`) 2226 million US dollars. The ontology also encodes information about well-known entities in the domain. For example, Figure 2(b) shows the fact that Los Angeles County (`g-LA`) is geographically contained in (is a `geoPartOf`) the LACMSA region (`g-LACMSA`), as is Ventura County (`g-VT`), something not immediately apparent from the LACMSA name.<sup>2</sup> The ontology includes rules definitions, such as the *recursive* rule in Figure 2(c) that specifies the transitivity of geospatial containment (`geoPartOf`), as well as disjointness constraints, such as the statement in Figure 2(d) that specifies that different product classifications do not have instances in common.

```
(defconcept Flow (?x) :<=>
  (exists (?o ?d ?p ?t ?u ?m ?v)
    (and (Data ?x)
      (hasOrigin ?x ?o) (Geo ?o)
      (hasDestination ?x ?d) (Geo ?d)
      (hasProduct ?x ?p) (Product ?p)
      (hasTimeInterval ?x ?t)
        (TimeInterval ?t)
      (hasUnit ?x ?u) (Unit ?u)
      (hasMode ?x ?m)
        (TransportationMode ?m)
      (hasValue ?x ?v) (Number ?v) )))
  (a) Concept Definition

  (geoPartOf g-LA g-LACMSA) (USCounty g-LA)
  (geoPartOf g-VT g-LACMSA) (USCounty g-VT)
  (USGeo g-LACMSA) (TransportationMode tm-air)
  (b) Instance Assertions

  (forall (?x ?z)
    (=> (exists (?y) (and (geoPartOf ?x ?y)
      (geoPartOf ?y ?z))))
    (geoPartOf ?x ?z)))
  (c) Inference Rule

  (mutually-disjoint-collection
    (setof IMPLAN NAICS SCTG SIC USC USCCOM))
  (d) Concept Disjointness Assertion
```

Fig. 2. Argos Ontology: Sample Definitions

**Relational Data Descriptions.** Using this ontology we describe the data provided by sources, and required or computed by services. In our domain, data are commonly represented as relational tables. We formally describe the tuples in such tables by formulas over concepts and relations of the ontology. Essentially, we associate a Local-as-View [8] definition to each source relation and to each input and output of a data processing operation. For example, Figure 3(a) shows the description of a source table `LACMSA-2000-EMP` that provides the number of jobs in 2000 for each TAZ in the LACMSA for products categorized following the 1999 Standard Industrial Classification (SIC) with a granularity of 4 digits.

In Argos we used *factored* data descriptions. Instead of considering the body of a description as a collection of predicates all of equal importance, we group the predicates into meaningful concepts for the domain, and then use these concepts as appropriate in the body of the relational definitions. For example, Argos uses the factored definitions of Figure 3(b) instead of the direct description of Figure 3(a). Although the direct and the factored representations are semantically

<sup>2</sup> The LACMSA comprises the counties of Los Angeles, Ventura, San Bernardino, Riverside and Orange.

```

(defrelation R-LACMSA-2000-EMP
  (?county ?jobs ?product ?taz) :<=>
  (exists (?o)
    (and (Measurement ?o)
      (hasGeo ?o ?taz) (TAZ ?taz)
      (geoPartOf ?taz ?county)
      (USCounty ?county)
      (geoPartOf ?taz g-LACMSA)
      (hasProduct ?o ?product)
      (Product-SIC-4-1999 ?product)
      (hasTimeInterval ?o year2000)
      (hasUnit ?o u-NumberOfJobs)
      (hasValue ?o ?jobs) (Number ?jobs)))

(a) Direct Definition

(relation-concept-mapping
  LACMSA-2000-EMP
  Employment-2000-LACMSA-TAZ-SIC))

(c) Bookkeeping

(defrelation LACMSA-2000-EMP
  ((?county USCounty) (?jobs Number)
  (?p Product-SIC-4-1999) (?taz TAZ)) :<=>
  (exists (?o)
    (and (Employment-2000-LACMSA-TAZ-SIC ?o)
      (hasProduct ?o ?p) (hasValue ?o ?jobs)
      (hasGeo ?o ?taz) (geoPartOf ?taz ?county))))

(defconcept Emp-2000-LACMSA-TAZ-SIC (?o) :<=>
  (exists (?p ?taz ?jobs)
    (and (Measurement ?o)
      (hasProduct ?o ?p) (Product-SIC-4-1999 ?p)
      (hasGeo ?o ?taz) (TAZ ?taz)
      (geoPartOf ?taz g-LACMSA)
      (hasTimeInterval ?o 2000)
      (hasUnit ?o u-NumberOfJobs)
      (hasValue ?o ?jobs))))

(b) Factored Definition

```

Fig. 3. Data Description for the LACMSA-2000-EMP source table

equivalent, factoring has knowledge engineering and reasoning advantages. From the knowledge engineering perspective, the concept definitions are more modular and can be reused in the definitions of many data relations. From the reasoning perspective, it is more efficient to compute subsumption between concepts than between relations (although PowerLoom can compute both). For example, two relations of the same arity may represent identical semantic information, but have their arguments in a different order. In order to prove semantic equivalence, our system would need to explore the permutations of the arguments, which may be expensive. Using the associated concept descriptions, the system can easily prove concept equivalence first and worry about the arguments in a second phase (cf. Section 3.3). As we describe later, storing the mapping between the relational description and distinguished concepts in its body (cf. Figure 3(c)), contributes to more efficient reasoning during planning.

In our representation we can describe both complete and incomplete data sources, corresponding respectively, to the *closed-world* and *open-world* semantics in data integration systems [8]. Defining a data source relation as *complete* means that it contains *all* the tuples that satisfy the ontology definition. We indicate completeness by using an if-and-only-if definition. For example, the relation definition in Figure 3 states that the table contains values for *all* the products of type `Product-sic-4-1999` for *all* the TAZs in the LACMSA. Conversely, defining a data source relation as *incomplete* means that although all the tuples in the source relation satisfy the definition, the relation may not contain all possible such tuples. That is, there may be other sources that contain additional tuples that satisfy the definition. We indicate incompleteness by using an if definition. For example, defining the table `LACMSA-2000-EMP` as incomplete would mean that there could be tuples missing for some TAZ or some product. In our transportation modeling domain complete descriptions are customary. In this paper

we focus primarily on reasoning with complete descriptions, but our approach can handle both complete and incomplete data descriptions (see Section 3.1).

### 3 Automatically Composing Data Processing Workflows

Argos automatically generates a workflow that answers user data requests by composing available sources and data processing operations. We assume that sources and operations are outside our control. For example, operations may be web services or functions from third-party libraries. Similarly, sources can be databases or other web services. We use the terms operation and service interchangeably. A source is an operation that does not require inputs.

Automatically generating a workflow presents two main challenges. First, services may use different schemas. Second, the data produced by a service may not be input directly into another, but may need some kind of transformation (*shim*). Argos addresses both these challenges. First, we resolve the semantic heterogeneity by describing the data in a common ontology. We associate an expressive Local-as-View [8] description with each data relation consumed or produced by a service. Second, we provide a set of *domain-independent relational* operations and a framework to define *generic domain-dependent* operations that can bridge the differences between the inputs and outputs of services.

In this section, we first present the planning algorithm that generates the workflow. Then, we describe the three types of operations that Argos supports: domain-dependent, domain-independent, and generic domain-dependent.

#### 3.1 Planning Algorithm

Our planner for workflow composition performs a regression search in plan space in the same fashion as partial-order planners such as UCPOP [17] and Sage [12]. The planner starts with the user data request as a goal and terminates the search when it finds a complete plan, that is, a plan where all data inputs to the component services are produced by other operations or sources.

The planning algorithm appears in Figure 4. The algorithm keeps an agenda  $A$  of services with unachieved inputs. Each element in the agenda is a pair  $[g, s]$  that consists of a data description  $g$  which is an unachieved input of service  $s$ . The planner *non-deterministically* chooses a plan refinement that solves an element from the agenda, that is, it searches the space of plans. The Argos planner considers three types of plan refinements: domain-dependent, generic domain-dependent and domain-independent.

In each plan refinement, the basic operation is to satisfy the input of a service with the output of another service. In order to ensure that the input and the output data relations are semantically compatible, the planner performs a *relational* subsumption or equivalence test. If the test succeeds, the planner establishes a *data link* from the output relation of one service to the input relation of the other service. This mechanism is analogous to the establishment of a causal link in plan-space planning [17] where the effect of an operator produces a precondition of another operator. However, instead of using simple unification to match

```

planner( $P, A$ )
  select pair  $[g, s_1]$  from agenda  $A$ 
  choose  $[P', A'] \in \text{plan-refinements}(P, A, [g, s_1])$ 
  planner( $P', A'$ )

add-domain-service( $P, A, [g, s_1]$ )
   $\forall s_2 \in \text{services}(\text{Domain})$  such that
     $\exists g_2 \in \text{outputs}(s_2)$  such that
      equivalent( $g_2, g$ ) (or subset( $g_2, g$ ))
       $P' := \text{add-data-link}(\text{add-step}(P, s_2), [s_2, g_2, g, s_1])$ 
       $A' := A - \{[g, s_1]\} \cup \{[g_2i, s_2] \mid g_2i \in \text{inputs}(s_2)\}$ 
      push  $[P', A']$  into refinements
  return refinements

plan-refinements( $P, A, [g, s_1]$ )
  return union(
    reuse-service( $P, A, [g, s_1]$ )
    add-domain-service( $P, A, [g, s_1]$ )
    ;; generic domain-dependent ops
    add-product-conversion( $P, A, [g, s_1]$ )
    add-unit-conversion( $P, A, [g, s_1]$ )
    ;; relational domain-independent ops
    add-selection( $P, A, [g, s_1]$ )
    add-projection( $P, A, [g, s_1]$ )
    add-join( $P, A, [g, s_1]$ )
    add-union( $P, A, [g, s_1]$ ) )

```

**Fig. 4.** Argos Planning Algorithm

a precondition with an effect predicate, our planner uses relational subsumption (or equivalence), because our inputs and outputs are universally quantified formulas that represent data relations. If an operation allows incomplete inputs, it suffices to prove that the description for the output of the provider operation is contained in the input of the consumer operation.

The Argos planner uses a best-first search strategy with a simple (non-admissible) heuristic that prefers to work on plans with the least number of unachieved inputs and contains the least operations already in the plan. As a heuristic optimization, the planning algorithm prefers to reuse an operation already in the plan rather than to insert a new one. This strategy usually leads to plans with a minimal number of operators in an efficient manner.

Since in our planning domain there are no negated effects (the operations do not destroy information), there is no need for threat detection as in classical partial-order planning. Our planner is limited to generating workflows that are directed acyclic graphs.

### 3.2 Domain-Dependent Services

Domain-dependent services are described by an input/output signature with predefined data relations. For example, the relation `LACMSA-2000-EMP` of Figure 3 describes the output of a service. A service can have multiple input and output data relations. Each service description is bound to a service implementation that can be a web service or a local program.

### 3.3 Domain-Independent Adaptors

In order to bridge the inputs and outputs of different services, Argos provides a set of built-in domain-independent *adaptor* services that correspond to the relation algebra operations: selection, projection, join, and union.

**Selection.** The selection plan refinement checks whether a data relation can be achieved by a selection operation on the output of an existing service. Since our descriptions are significantly more expressive than those used in databases, proving the applicability of selection involves reasoning with background information in the ontology, not just the data definitions.

```

add-selection( $P, A, \langle wantR, s_c \rangle$ )
   $\forall haveR \in outputs(s_p) \wedge s_p \in services(Domain) \wedge$ 
    relation-concept-mapping( $haveR, wantSC$ )  $\wedge$  superconcept( $wantSC, wantC$ )  $\wedge$ 
    relation-concept-mapping( $wantR, wantC$ )
    [ $compatible, mapping, selections$ ] := compatible-signatures( $haveR, wantR$ )
  if  $compatible$  then
     $argsSelR$  := apply( $mapping, args(haveR)$ )
     $selR$  := ( $\kappa$  ( $argsSelR$ ) (and  $haveR$  selections))
  if equivalent( $selR, wantR$ ) then
     $s_\sigma$  := create-select-service( $haveR, selections, selR$ )
     $P'$  := add-data-links(add-step( $P, s_\sigma$ ),  $\langle s_\sigma, wantR, s_c \rangle$ )
     $A'$  :=  $A - \{\langle wantR, s_c \rangle\} \cup \{\langle haveR, s_\sigma \rangle\}$ 
    push  $\langle P', A' \rangle$  into refinements
return refinements

compatible-signatures( $haveR, wantR, wantC$ )
  Check that  $args(haveR)$  and  $args(wantR)$  satisfy conditions:
  1.  $\forall wa \in args(wantR) \exists ha \in args(haveR)$  equivalent( $type(wa), type(ha)$ )
     push [ $wa \mapsto ha$ ] into  $mapping$ 
  2.  $\forall a \in args(haveR) \wedge a \notin mapping$ 
      $\exists K \in definition(wantC) \wedge individual(K) \wedge$  equivalent( $type(K), type(a)$ )
     push ( $= a K$ ) into  $selections$ 
  return [ (and 1 2),  $mapping, selections$ ]

```

Fig. 5. Selection Plan Refinement

The algorithm for the `add-selection` plan refinement appears in Figure 5. The process is better described using an example. Assume that a service  $S_c$  requires as input the employment data for the TAZs in Los Angeles County, as described by the `LA-2000-EMP` relation of Figure 6(a). Further assume that there is another service  $S_p$  that is able produce the employment data for all the TAZs of the LACMSA, that is, one of the outputs of  $S_p$  is the `LACMSA-2000-EMP` relation described in Figure 3. Note the differences between the two data descriptions. The `LA-2000-EMP` table has three columns and contains data only for TAZs in Los Angeles county (`g-LA`). The `LACMSA-2000-EMP` table has four columns and contains data for TAZs in the five-county LACMSA. Intuitively, the system can prove that since Los Angeles is a geographical part of the LACMSA (cf. Figure 2), `LA-2000-EMP` is a subset of `LACMSA-2000-EMP`, and thus the planner can just select from it to achieve the goal.

The relations that are candidate for a selection operation must be a superset of the goal relation. The `add-selection` refinement first finds each candidate relation  $haveR$ , which is the output of an existing service  $S_c$ , and whose associated concept  $wantSC$  is a superconcept of the associated concept  $wantC$  of the goal relation  $wantR$ . In our example,  $wantR$  is `LA-2000-EMP`,  $wantC$  is `Employment-2000-LA-TAZ-SIC`,  $wantSC$  is `Employment-2000-LACMSA-TAZ-SIC`, and  $haveR$  is `LACMSA-2000-EMP`. The `compatible-signatures` procedure tests whether the arguments of the candidate and the goal relations are compatible by testing two conditions. The first condition finds a mapping between arguments of the same

```
(defrelation LA-2000-EMP ((?taz TAZ) (?p Product-SIC-4-1999) (?jobs Number)) :<=>
  (exists (?o) (and (Employment-2000-LA-TAZ-SIC ?o)
    (hasProduct ?o ?p) (hasGeo ?o ?taz) (TAZ ?taz) (hasValue ?o ?jobs))))

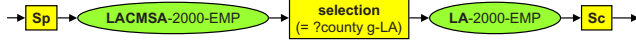
(defconcept Employment-2000-LA-TAZ-SIC (?o) :<=> (exists (?p ?taz ?jobs)
  (and (Measurement ?o) (hasGeo ?o ?taz) (TAZ ?taz) (geoPartOf ?taz g-LA)
    (hasProduct ?o ?p) (Product ?p) (ofClassification ?p SIC-4-1999)
    (hasUnit ?o u-NumberOfJobs) (hasTimeInterval ?o 2000) (hasValue ?o ?jobs))))

(relation-concept-mapping LA-2000-EMP Employment-2000-LA-TAZ-SIC)
```

(a) Input Required by Service Sc

```
(kappa ((?taz TAZ) (?p Product-SIC-4-1999) (?jobs Number))
  (and (LACMSA-2000-EMP ?c ?jobs ?p ?taz) (= ?c g-LA)))
```

(b) Output of Selection Operator (which is equivalent to input required by Service Sc)



(c) Plan with Relational Selection Adaptor

Fig. 6. Domain-Independent Adaptor: Selection

type. In our example, the arguments types of `LA-2000-EMP` map to the fourth (`TAZ`), third (`Product-SIC-4-1999`), and second (`Number`) arguments of `LACMSA-2000-EMP`. The second condition tests whether there are constants/individuals in the definition of the concept *wantC* associated with the goal relation *wantR* of the same type as the unmapped arguments of the candidate relation *haveR*. In our example, the unmapped argument is `USCounty` and the definition of `Employment-2000-LA-TAZ-SIC` contains the individual `g-LA` which is an instance of `USCounty`. Finally, the `add-selection` refinement checks whether the anonymous<sup>3</sup> relation definition *selR*, the conjunction of relation *haveR* and found *selections* (shown in Figure 6(b)), is equivalent to the goal relation *wantR* (`LA-2000-EMP`). If so, the refinement succeeds and adds a selection service to the plan, as shown in Figure 6(c).

**Projection, Join and Union.** The projection, join and union plan refinements have a similar purpose as the corresponding relational algebra operations. They introduce a projection, join or union service to enable the match of inputs and outputs. The algorithms for these refinements are analogous to the selection refinement we described above. The algorithm for the projection plan refinement searches for an output relation of a service whose projection into the desired attributes is equivalent to/contained in the desired input relation. The algorithm for the join (union) plan refinement searches for outputs relations whose conjunction (disjunction) is equivalent to/contained in the desired input relation.

### 3.4 Generic Domain-Dependent Adaptors

There are a variety of operators that lie between the completely domain-specific operators that are described by predefined input and output datasets (cf.

<sup>3</sup> Anonymous relations in PowerLoom are denoted with the `kappa` symbol (by analogy to anonymous functions in the lambda calculus).



Section 3.2), and the domain-independent operators that are applicable to any dataset description regardless of the domain (cf. Section 3.3).

Product conversion is a prime example of a generic, but domain-dependent operator. Economic data is reported in a variety of classifications, such as SIC or NAICS. Thus, when integrating data from different sources, the system must translate between classifications. Instead of defining a host of domain-specific operators, we added a generic product conversion refinement to the Argos library.

When the planner needs to satisfy a given request for products in a classification C2, the `add-product-conversion` refinement introduces a product conversion service and subgoals for obtaining a conversion table from C1 to C2 and the desired data in classification C1. As an optimization, the refinement checks the service descriptions to ensure that the C1-to-C2 conversion table is the output of an available service. Figure 7 shows an example. Assume that a service `Sc` requires as input the employment data for LACMSA by 6-digit NAICS industry codes as described by the relation in Figure 7(a) (for simplicity, we show the direct not the factored representation). First, the algorithm finds sources for conversion tables into 6-digit NAICS by issuing the query shown in Figure 7(b) against the ontology. Assume that it finds a source `Sp` that produces the relation `SIC2NAICS` of Figure 7(c). Second, the algorithm adds a product conversion service and subgoals on obtaining a data relation with the same definition as the originally desired relation except that the product classification is in SIC instead of NAICS codes. The resulting plan is shown in Figure 7(d). Our system also includes a unit conversion operator that works in a similar fashion.

## 4 Empirical Evaluation

We tested our workflow planner using two ontologies. The first is our production ontology, *Argos* (A), that was created by consulting our domain experts. It contains 162 concepts, 67 relations, and 28 domain service descriptions (17 sources and 11 operations, with a total of 37 input and 32 output data relations). The data sources used different product classifications (SCTG, NAICS, SIC, IMPLAN). With the help of 5 conversion tables defined by the domain experts, these products classifications are eventually mapped into one product classification for uniformity. The second ontology, *Extended Argos* (EA), we defined with the purpose of testing the planner in a domain where cycles of operations are possible. This ontology includes 17 product conversion tables that can convert between any pair of classifications and may lead to infinite cycles of product conversions. It contains 173 concepts, 79 relations, 40 domain services (29 sources and 11 operations, with a total of 37 inputs and 44 outputs).

Table 1 shows the planning performance on 18 typical queries. For example, the third row, query Q3, shows the results for the `LA-2000-EMP` relation of Figure 6. Query Q17 asks for the total (intra- plus inter-regional) demand for all TAZs in the LACMSA region for all transportation modes. Query Q18 asks for the truck traffic volume for all links of the LACMSA highway network. In



```
(defrelation LACMSA-2000-EMP-NAICS (?county ?jobs ?product ?taz)
:<=>
  (exists (?o) (and (Measurement ?o) (hasGeo ?o ?taz) (TAZ ?taz)
    (geoPartOf ?taz ?county) (USCounty ?county) (geoPartOf ?taz g-LACMSA)
    (hasProduct ?o ?product) (Product-NAICS-6-2002 ?product)
    (hasUnit ?o u-NumberOfJobs) (hasTimeInterval ?o year2000) (hasValue ?o ?jobs))))
```

(a) Desired Data Relation

```
(retrieve all (?s ?x) (and (source ?s) (hasOutput ?s ?x)
(subset-of ?x
  (kappa (?fp ?tp ?proportion) (exists (?o) (and (ProductConversion ?o)
    (fromProduct ?o ?fp) (toProduct ?o ?tp) (Product-NAICS-6-2002 ?tp))))))
```

(b) Ontology Query to Retrieve Product Conversion Relations

```
(defrelation SIC2NAICS (?fp ?tp ?proportion) :<=>
  (exists (?o) (and (ProductConversion ?o) (fromProduct ?o ?fp) (Product-SIC-4-1999 ?fp)
    (toProduct ?o ?tp) (Product-NAICS-6-2002 ?tp) (hasValue ?o ?proportion) )))
```

(c) Product Conversion Relation



(d) Plan with Product Conversion Adaptor

**Fig. 7.** Generic Domain-Dependent Adaptor: Product Conversion

response to Q18, Argos generates the workflow of Figure 11(b). Figure 11(c) shows the results of query Q18 displayed in ArcGIS.

We tested the workflow planning algorithm with ontologies A and EA. We report the number of services and data links in the resulting workflows, the total workflow generation time and the portion spent in PowerLoom reasoning (both in seconds), the number of subsumption tests, and the number of search nodes (partial plans) generated and visited. The experiments were run on a laptop running Windows XP with a 2GHz processor and 2GB of memory.

Overall, we find the planning performance satisfactory for data processing workflows, where the execution time of the workflow dominates. Consider queries Q17 and Q18 that generate the largest workflows. For Q17 the planning time is 61.58 seconds, generating a workflow with 53 services. The execution time is 224 seconds, processing a total of 2056247 tuples, and producing a result relation with 74350 tuples. For Q18 the planning time is 73.29 seconds, generating a workflow with 54 services. The execution time is 1280 seconds, processing a total of 1980860 tuples, and producing a result relation with 89356 tuples<sup>4</sup>.

The results show that the increased possibilities for product conversion in ontology EA increases the planning time in the largest plans (from approx. 73 to 127 seconds) due to an increased number of subsumption queries, but on the other hand they lead to shorter plans (from 54 to 51 services).

<sup>4</sup> The last step in the workflow for Q18, the network equilibrium algorithm which computes truck traffic in each highway link, is particularly time consuming.

We also tested unsatisfiable requests that could lead to an infinite chain of product conversion operators. In ontology A the products conversions form an acyclic directed graph, so there is no possibility of infinite subgoaling. However, in ontology EA there are cycles. Thus, we added a limit to the depth of chains of instantiations of the same operation, in our example, product conversion chains. Experimentally, to prove a query unsatisfiable with chain limits of length 3 and 5, the planner using ontology A takes 4 and 4.17 seconds, respectively, and using ontology EA it takes 89.70 and 236.42 seconds, respectively. There is also a practical reason for such limit. Since each product conversion is an approximation, a long chain would produce very low quality data.

**Table 1.** Experimental Results

Query	Services		Data Links		Planning Time (s)		PowerLoom Time (s)		Subsumptions		Plans Generated		Plans Visited	
	A	EA	A	EA	A	EA	A	EA	A	EA	A	EA	A	EA
Q1	2	2	1	1	1.98	2.22	1.98	2.2	8	8	1	1	1	1
Q2	3	3	2	2	2.74	3.36	2.74	3.36	16	16	2	2	2	2
Q3	4	4	3	3	4.36	5.23	4.33	5.15	30	36	3	9	3	3
Q4	5	5	4	4	3.69	3.61	3.52	3.59	34	42	4	12	4	4
Q5	6	4	5	3	3.27	5.89	3.23	5.84	50	82	8	23	5	7
Q6	11	11	10	10	5.22	6.2	5.19	6.19	76	76	10	10	10	10
Q7	11	11	10	10	5.47	5.89	5.45	5.84	76	76	10	10	10	10
Q8	11	11	10	10	5.52	6.22	5.48	6.2	76	76	10	10	10	10
Q9	11	11	10	10	5.5	5.89	5.44	5.86	76	76	10	10	10	10
Q10	12	9	11	8	5.94	16.45	5.94	16.3	117	325	16	98	13	31
Q11	17	12	17	12	6.22	19	6.09	18.66	151	389	24	121	19	39
Q12	18	13	19	14	7.09	19.61	6.92	19.22	159	397	27	124	21	41
Q13	18	13	19	14	7.14	19.61	7	19.33	159	397	27	124	21	41
Q14	33	33	44	44	15.64	16.84	13.36	14.44	250	250	44	44	44	44
Q15	33	33	44	44	15.88	16.86	13.61	14.48	250	250	44	44	44	44
Q16	53	48	69	64	60.25	89.59	22.34	68.59	442	722	78	187	71	95
Q17	53	48	69	64	61.58	90.39	23.54	69.1	442	722	78	187	71	95
Q18	54	51	71	67	73.29	127.67	69.16	115.75	435	600	78	171	75	110

## 5 Related Work

Our planner is inspired by the representation of information gathering actions of the Sage planner [12] of the SIMS [4] mediator, where knowledge preconditions and effects were represented as queries. However, our work has several major differences with SIMS. First, our domain and service descriptions, expressed in PowerLoom, are significantly more expressive than SIMS’s use of the Loom description logic. For example, relational subsumption over recursive descriptions, as required by the `geoPartOf` relation in the selection example of Figure 6 could not be performed in SIMS. Second, Argos uses subsumption to established data links, while SIMS relied on syntactic matching to establish causal links. On the other hand, SIMS included cost optimization techniques [3], like pushing selections, that we have not (yet) incorporated in our planner.

The TAMBIS system [5] integrated heterogeneous data and analysis tools in a bioinformatics domain. They used a domain ontology expressed in the GRAIL description logic as a basis of the integration. We share many of the goals of

TAMBIS. However, our local-as-view relational descriptions and use of relation subsumption (as opposed to TAMBIS’s concept subsumption) yields a more expressive and principled system that can produce more flexible workflows.

With respect to the shim classification of [10], our adaptors fall mostly in the semantic translator category (e.g., product conversion). However, the relational-algebra adaptors (Section 3.3) constitute a new class of adaptors.

Szomszor et al. [23] present an approach to syntactic mediation in web service workflows that automatically inserts type adaptors that translate XML data. In contrast to their work, we do not focus on specifying the implementation of a translation operator, only on describing semantically the input/output signature. Incidentally, since we deal with relational data, sometimes translations can be implemented as SQL queries (with aggregation), as is the case in the product conversion adaptor.

Our domain modeling builds on the idea of faceted representations (e.g. [19]). In particular we structured some of the main concepts in the ontology along basic dimensions like location, time, product, similarly to the Energy Data Collection (EDC) project [1]. However, we have a more refined domain ontology. More critically, since EDC was based on SIMS, its ability to describe data was limited.

Research on mixed-initiative composition of scientific workflows [11,22] also leverages semantic descriptions. However, Argos uses a more expressive description language and focuses on automatic composition.

There has been research on automatic web service composition within the AI planning community: executable grid workflows [6], HTN-based composition [20], composition using Golog-procedures [15]. Some of these systems are more expressive than Argos since they model state change. However, Argos provides more expressive data representation and manipulation.

## 6 Discussion and Future Work

We have presented a logic-based planning approach to automatically compose data processing workflows. We describe data and services using a formal ontology. The planner uses the ontology and relational subsumption, provided by the PowerLoom reasoner, to construct workflows that answer user data requests.

We are currently working on using query reformulation to link services’ inputs and outputs. Inputs can be seen as queries and outputs as views (in the database sense), so standard techniques for answering queries using views [8] can be applied. We are exploring more restricted languages like conjunctive queries, as well as developing a query reformulation algorithm for the PowerLoom first-order logic based on abductive reasoning.

Finally, we plan to incorporate cost optimization knowledge into the planner, so that it generates more efficient workflows, for example, by pushing selections closer to the sources and other optimizations.

## References

1. Ambite, J.L., Arens, Y., Hovy, E., Philpot, A., Gravano, L., Hatzivassiloglou, V., Klavans, J.: Simplifying data access: The energy data collection (EDC) project. *IEEE Computer*, Special Issue on Digital Government 34(2) (February 2001)
2. Ambite, J.L., Kapoor, D.: Automatic generation of data processing workflows for transportation modeling. In *Proceedings of the 8th Annual International Conference on Digital Government Research (dg.o2007)*, Philadelphia, PA, USA (2007)
3. Ambite, J.L., Knoblock, C.A.: Flexible and scalable cost-based query planning in mediators: A transformational approach. *Artificial Intelligence* 118(1-2), 115–161 (2000)
4. Arens, Y., Knoblock, C.A., Shen, W.-M.: Query reformulation for dynamic information integration. *Journal of Intelligent Information Systems*, Special Issue on Intelligent Information Integration 6(2/3), 99–130 (1996)
5. Baker, P.G., Brass, A., Bechhofer, S., Goble, C., Paton, N., Stevens, R.: TAMBIS: Transparent access to multiple bioinformatics information sources. In *Proceedings of the 6th International Conference on Intelligent Systems for Molecular Biology*, Montreal, Canada (1998)
6. Blythe, J., Deelman, E., Gil, Y.: Automatically composed workflows for grid environments. *IEEE Intelligent Systems*, 16–23 (July/August 2004)
7. Giuliano, G., Gordon, P., Pan, Q., Park, J., Wang, L.: Estimating freight flows for metropolitan highway networks using secondary data sources. In *Proceedings of the Transportation Research Board Commodity Flow Survey Conference*, Transportation Research Circular, E-C088, 154–158 (July 2005)
8. Halevy, A.Y.: Answering queries using views: A survey. *The VLDB Journal* 10(4), 270–294 (2001)
9. Horrocks, I., Sattler, U., Tessaris, S., Tobies, S.: How to decide query containment under constraints using a description logic. In: Parigot, M., Voronkov, A. (eds.) *LPAR 2000. LNCS (LNAI)*, vol. 1955, Springer, Heidelberg (2000)
10. Hull, D., Stevens, R., Lord, P., Wroe, C., Goble, C.: Treating shimantic web syndrome with ontologies. In *Proceedings of the 1st AKT workshop on Semantic Web Services (AKT-SWS04)*, Milton Keynes, UK (2004)
11. Kim, J., Spraragen, M., Gil, Y.: An intelligent assistant for interactive workflow composition. In *Proceedings of the International Conference on Intelligent User Interfaces*, Madeira, Portugal (2004)
12. Knoblock, C.A.: Building a planner for information gathering: A report from the trenches. In *Proceedings of the 3rd International Conference on Artificial Intelligence Planning Systems*, Edinburgh, Scotland (1996)
13. MacGregor, R.: A deductive pattern matcher. In *Proceedings of the 7th National Conference on Artificial Intelligence*, Saint Paul, MN (1988)
14. MacGregor, R.: A description classifier for the predicate calculus. In *Proceedings of the Proceedings of the 12th National Conference on Artificial Intelligence*, Seattle, WA (1994)
15. McIlraith, S., Son, T.C.: Adapting Golog for composition of semantic web services. In *Proceedings of the 8th International Conference on Knowledge Representation and Reasoning* (2002)
16. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic matching of web services capabilities. In *Proceedings of the International Semantic Web Conference*, Sardinia, Italy (2002)

17. Penberthy, J.S., Weld, D.S.: UCPOP: A sound, complete, partial order planner for ADL. In Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning, Cambridge, MA (1992)
18. PowerLoom. The PowerLoom knowledge representation & reasoning system (2003), [www.isi.edu/isd/LOOM/PowerLoom](http://www.isi.edu/isd/LOOM/PowerLoom)
19. Pratt, W., Hearst, M.A., Fagan, L.M.: A knowledge-based approach to organizing retrieved documents. In Proceedings of the 16th National Conference on Artificial intelligence, Menlo Park, CA, USA (1999)
20. Sirin, E., Parsia, B., Wu, D., Hendler, J., Nau, D.: HTN planning for web service composition using SHOP2. *Web Semantics* 1(4), 377–396 (2004)
21. Sivashanmugam, K., Verma, K., Sheth, A., Miller, J.: Adding semantics to web services standards. In Proceedings of the 1st International Conference on Web Services. Las Vegas, NV (2003)
22. Stevens, R.D., Robinson, A.J., Goble, C.A.: myGrid: personalised bioinformatics on the information grid. *Bioinformatics* 19(1), 302–304 (2003)
23. Szomszor, M., Payne, T., Moreau, L.: Automated syntactic mediation for web service integration. In Proceedings of the International Conference on Web Services, Chicago, IL (2006)

# COMM: Designing a Well-Founded Multimedia Ontology for the Web

Richard Arndt<sup>1</sup>, Raphaël Troncy<sup>2</sup>, Steffen Staab<sup>1</sup>, Lynda Hardman<sup>2,\*</sup>,  
and Miroslav Vacura<sup>3</sup>

<sup>1</sup> ISWeb, University of Koblenz-Landau, Germany  
{rarndt,staab}@uni-koblenz.de

<sup>2</sup> CWI, Amsterdam, The Netherlands

{Raphael.Troncy,Lynda.Hardman}@cwi.nl

<sup>3</sup> University of Economics, Prague  
vacuram@vse.cz

**Abstract.** Semantic descriptions of non-textual media available on the web can be used to facilitate retrieval and presentation of media assets and documents containing them. While technologies for multimedia semantic descriptions already exist, there is as yet no formal description of a high quality multimedia ontology that is compatible with existing (semantic) web technologies. We explain the complexity of the problem using an annotation scenario. We then derive a number of requirements for specifying a formal multimedia ontology before we present the developed ontology, COMM, and evaluate it with respect to our requirements. We provide an API for generating multimedia annotations that conform to COMM.

## 1 Introduction

Multimedia objects on the Web are ubiquitous, whether found via web-wide search (e.g., Google or Yahoo! images<sup>[1]</sup>) or via dedicated sites (e.g., Flickr or YouTube<sup>[2]</sup>). These media objects are produced and consumed by professionals and amateurs alike. Unlike textual assets, whose content can be searched for using text strings, media search is dependent on processes that have either cumbersome requirements for feature comparison (e.g. color or texture) or rely on associated, more easily processable descriptions, selecting aspects of an image or video and expressing them as text, or as concepts from a predefined vocabulary. Individual annotation and tagging applications have not yet achieved a degree of interoperability that enables effective sharing of semantic metadata and that links the metadata to semantic data and ontologies found in the Semantic Web.

MPEG-7 [1, 2] is an international standard that specifies how to connect descriptions to parts of a media asset. The standard includes descriptors representing low-level media-specific features that can often be automatically extracted

---

\* Lynda Hardman is also affiliated with the Technical University of Eindhoven.

<sup>1</sup> <http://images.google.com/>, <http://images.search.yahoo.com/>

<sup>2</sup> <http://www.flickr.com/>, <http://www.youtube.com/>

from media types. Unfortunately, MPEG-7 is not currently suitable for describing multimedia content on the Web, because *i)* its XML Schema-based nature prevents direct machine processing of semantic descriptions and its use of URNs is cumbersome for the Web; *ii)* it is not open to Web standards, which represent knowledge and make use of existing controlled vocabularies.

The Web provides an open environment where information can be shared and linked to. It has, however, no agreed-upon means of describing and connecting semantics with (parts of) multimedia assets and documents. While multimedia description frameworks, such as MPEG-7, already exist, no formal description of a multimedia ontology is compatible with existing (semantic) web technologies. Our contribution is thus to combine the advantages of the extensibility and scalability of web-based solutions with the accumulated experience of MPEG-7. Our approach advocates the use of formal semantics, grounded in a sound ontology development methodology, to describe the required multimedia semantics in terms of current semantic web languages. We develop COMM, a Core Ontology for MultiMedia.

In the next section, we illustrate the main problems when using MPEG-7 for describing multimedia resources on the web. In section 3, we review existing multimedia ontologies and show why the proposals made so far are inadequate for our purposes. Subsequently, we define the requirements that a multimedia ontology should meet (section 4) before we present COMM – an MPEG-7 based ontology, designed using sound design principles – and discuss our design decisions based on our requirements (section 5). In section 6, we demonstrate the use of the ontology with the scenario from section 2 and then conclude with some observations and future work.

## 2 Annotating Multimedia Documents on the Web

Let us imagine that an employee of an encyclopedia company wants to create a multimedia presentation of the Yalta Conference. For that purpose, s/he uses an MPEG-7 compliant authoring tool for detecting and labeling relevant multimedia objects automatically. On the web, the employee finds three different face recognition web services, each of them providing very good results for detecting Winston Churchill, Franklin D. Roosevelt and Josef Stalin respectively. Having these tools, the employee would like to run the face recognition web services on images and import the extraction results into the authoring tool in order to automatically generate links from the detected face regions to detailed textual information about Churchill, Roosevelt and Stalin. Fig. 1-A is an example of such an image; the bounding boxes are generated by the face recognition web services and linked to textual data by the authoring tool. This scenario, however, causes several problems with existing solutions:

**Fragment identification.** Particular regions of the image need to be localized (anchor value in 3). However, the current web architecture does not provide a means for uniquely identifying sub-parts of multimedia assets, in the same way that the fragment identifier in the URI can refer to part of an HTML or XML



**Fig. 1.** MPEG-7 annotation example (Image adapted from Wikipedia), [http://en.wikipedia.org/wiki/Yalta\\_Conference](http://en.wikipedia.org/wiki/Yalta_Conference)

document. Actually, for almost all other media types, the semantics of the fragment identifier has not been defined or is not commonly accepted. Providing an agreed upon way to localize sub-parts of multimedia objects (e.g. sub-regions of images, temporal sequences of videos or tracking moving objects in space and in time) is fundamental [3, 4]. For images, one can use either MPEG-7 or SVG snippet code to define the bounding box coordinates of specific regions. For temporal location, one can use MPEG-7 code or the TemporalURI RFC [4]. MPEG-21 specifies a normative syntax to be used in URIs for addressing parts of any resource but whose media type is restricted to MPEG [5]. The MPEG-7 approach requires an indirection: an annotation is *about* a fragment of a XML document that *refers* to a multimedia document, whereas the MPEG-21 approach does not have this limitation.

**Semantic annotation.** MPEG-7 is a natural candidate for representing the extraction results of multimedia analysis software such as a face recognition web service. The language, standardized in 2001, specifies a rich vocabulary of multimedia descriptors, which can be represented in either XML or a binary format. While it is possible to specify very detailed annotations using these descriptors, it is not possible to guarantee that MPEG-7 metadata generated by different agents will be mutually understood due to the lack of formal semantics of this language [6, 7]. The XML code of Fig. 1-B illustrates the inherent interoperability problems of MPEG-7: several descriptors, semantically equivalent and representing the same information while using different syntax can coexist [8]. As our employee used three different face recognition web services, the extraction results of the regions SR1, SR2 and SR3 differ from each other even though they are all

<sup>3</sup> See also the related discussion in the W3C Multimedia Semantics XG

<http://lists.w3.org/Archives/Public/public-xg-mmsem/2007Apr/0007.html>

<sup>4</sup> [http://www.annotex.net/TR/URI\\_fragments.html](http://www.annotex.net/TR/URI_fragments.html)



syntactically correct. While the first service uses the MPEG-7 `SemanticType` for assigning the `<Label> Roosevelt` to still region SR1, the second one makes use of a `<KeywordAnnotation>` for attaching the keyword *Churchill* to still region SR2. Finally the third service uses a `<StructuredAnnotation>` (which can be used within the `SemanticType`) in order to label still region SR3 with *Stalin*. Consequently, alternative ways for annotating the still regions render almost impossible the retrieval of the face recognition results within the authoring tool since the corresponding XPath query has to deal with these syntactic variations. As a result, the authoring tool will not link occurrences of Churchill in the image with, for example, his biography as it does not expect semantic labels of still regions behind the `<KeywordAnnotation>` element.

**Web interoperability.** Finally, our employee would like to link the multimedia presentation to historical information about the key figures of the Yalta Conference that is already available on the web. S/He has also found semantic metadata about the relationships between these figures that could improve the automatic generation of the multimedia presentation. However, s/he realizes that MPEG-7 cannot be combined with these concepts defined in domain-specific ontologies because of its closing to the web. As this example demonstrates, although MPEG-7 provides ways of associating semantics with (parts of) non-textual media assets, it is incompatible with (semantic) web technologies and has no formal description of the semantics encapsulated implicitly in the standard.

### 3 Related Work

In the field of semantic image understanding, using a multimedia ontology infrastructure is regarded to be the first step for closing the, so-called, semantic gap between low-level signal processing results and explicit semantic descriptions of the concepts depicted in images. Furthermore, multimedia ontologies have the potential to increase the interoperability of applications producing and consuming multimedia annotations. The application of multimedia reasoning techniques on top of semantic multimedia annotations is also a research topic which is currently investigated [9]. A number of drawbacks of MPEG-7 have been reported [10, 11]. As a solution, multimedia ontologies based on MPEG-7 have been proposed.

Hunter [6] provided the first attempt to model parts of MPEG-7 in RDFS, later integrated with the ABC model. Tsinaraki et al. [12] start from the core of this ontology and extend it to cover the full Multimedia Description Scheme (MDS) part of MPEG-7, in an OWL DL ontology. A complementary approach was explored by Isaac and Troncy [13], who proposed a core audio-visual ontology inspired by several terminologies such as MPEG-7, TV Anytime or ProgramGuideML. Garcia and Celma [14] produced the first complete MPEG-7 ontology, automatically generated using a generic mapping from XSD to OWL. Finally, Simou proposed an OWL DL Visual Descriptor Ontology<sup>5</sup> (VDO) based on the Visual part of MPEG-7 and used for image and video analysis.

<sup>5</sup> <http://image.ece.ntua.gr/~gstoil/VDO>

All these methods perform a one to one translation of MPEG-7 types into OWL concepts and properties. This translation does not, however, guarantee that the intended semantics of MPEG-7 is fully captured and formalized. On the contrary, the syntactic interoperability and conceptual ambiguity problems illustrated in section 2 remain.

## 4 Requirements for Designing a Multimedia Ontology

Requirements for designing a multimedia ontology have been gathered and reported in the literature, e.g. in [15]. Here, we compile these and use our scenario to present a list of requirements for a web-compliant multimedia ontology.

**MPEG-7 compliance.** MPEG-7 is an existing international standard, used both in the signal processing and the broadcasting communities. It contains a wealth of accumulated experience that needs to be included in a web-based ontology. In addition, existing annotations in MPEG-7 should be easily expressible in our ontology.

**Semantic interoperability.** Annotations are only re-usable when the captured semantics can be shared among multiple systems and applications. Obtaining similar results from reasoning processes about terms in different environments can only be guaranteed if the semantics is sufficiently explicitly described. A multimedia ontology has to ensure that the intended meaning of the captured semantics can be shared among different systems.

**Syntactic interoperability.** Systems are only able to share the semantics of annotations if there is a means of conveying this in some agreed-upon syntax. Given that the (semantic) web is an important repository of both media assets and annotations, a semantic description of the multimedia ontology should be expressible in a web language (e.g. OWL, RDF/XML or RDFa).

**Separation of concerns.** Clear separation of domain knowledge (i.e. knowledge about depicted entities, such as the person Winston Churchill) from knowledge that is related to the administrative management or the structure and the features of multimedia documents (e.g. Churchill's face is to the left of Roosevelt's face) is required. Reusability of multimedia annotations can only be achieved if the connection between both ontologies is clearly specified by the multimedia ontology.

**Modularity.** A complete multimedia ontology can be, as demonstrated by MPEG-7, very large. The design of a multimedia ontology should thus be made modular, to minimize the execution overhead when used for multimedia annotation. Modularity is also a good engineering principle.

**Extensibility.** While we intend to construct a comprehensive multimedia ontology, as ontology development methodologies demonstrate, this can never be complete. New concepts will always need to be added to the ontology. This requires a design that can always be extended, without changing the underlying model and assumptions and without affecting legacy annotations.

## 5 Adding Formal Semantics to MPEG-7

MPEG-7 specifies the connection between semantic annotations and parts of media assets. We take it as a base of knowledge that needs to be expressible in our ontology. Therefore, we re-engineer MPEG-7 according to the intended semantics of the written standard. We satisfy our semantic interoperability not by aligning our ontology to the XML Schema definition of MPEG-7, but by providing a formal semantics for MPEG-7. We use a methodology based on a foundational, or top level, ontology as a basis for designing COMM. This provides a domain independent vocabulary that explicitly includes formal definitions of foundational categories, such as processes or physical objects, and eases the linkage of domain-specific ontologies because of the definition of top level concepts. We briefly introduce our chosen foundational ontology in section 5.1, and then present our multimedia ontology, COMM, in sections 5.2 and 5.3. Finally, we discuss why our ontology satisfies all our stated requirements in section 5.4.

COMM is available at <http://multimedia.semanticweb.org/COMM/>.

### 5.1 DOLCE as Modeling Basis

Using the review in [16], we select the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [17] as a modeling basis. Our choice is influenced by two of the main design patterns: *Descriptions & Situations* (D&S) and *Ontology of Information Objects* (OIO) [18]. The former can be used to formalize contextual knowledge, while the latter, based on D&S, implements a semiotics model of communication theory. We consider that the annotation process is a *situation* (i.e. a reified context) that needs to be described.

### 5.2 Multimedia Patterns

The patterns for D&S and OIO need to be extended for representing MPEG-7 concepts since they are not sufficiently specialized to the domain of multimedia annotation. This section introduces these extended multimedia design patterns, while section 5.3 details two central concepts underlying these patterns: digital data and algorithms. In order to define design patterns, one has to identify repetitive structures and describe them at an abstract level. We have identified the two most important functionalities provided by MPEG-7 in the scenario presented in section 2: the *decomposition* of a media asset and the (semantic) *annotation* of its parts, which we include in our multimedia ontology.

**Decomposition.** MPEG-7 provides descriptors for spatial, temporal, spatio-temporal and media source decompositions of multimedia content into segments. A segment is the most general abstract concept in MPEG-7 and can refer to a region of an image, a piece of text, a temporal scene of a video or even to a moving object tracked during a period of time.

**Annotation.** MPEG-7 defines a very large collection of descriptors that can be used to annotate a segment. These descriptors can be low-level visual features,

audio features or more abstract concepts. They allow the annotation of the content of multimedia documents or the media asset itself.

In the following, we first introduce the notion of multimedia data and then present the patterns that formalize the decomposition of multimedia content into segments, or allow the annotation of these segments. The decomposition pattern handles the structure of a multimedia document, while the media annotation pattern, the content annotation pattern and the semantic annotation pattern are useful for annotating the media, the features and the semantic content of the multimedia document respectively.

*Multimedia Data.* This encapsulates the MPEG-7 notion of multimedia content and is a subconcept of `DigitalData`<sup>6</sup> (introduced in more detail in section 5.3). `MultimediaData` is an abstract concept that has to be further specialized for concrete multimedia content types (e.g. `ImageData` corresponds to the pixel matrix of an image). According to the OIO pattern, `MultimediaData` is realized by some physical `Media` (e.g. an `Image`). This concept is needed for annotating the physical realization of multimedia content (see section 5.3).

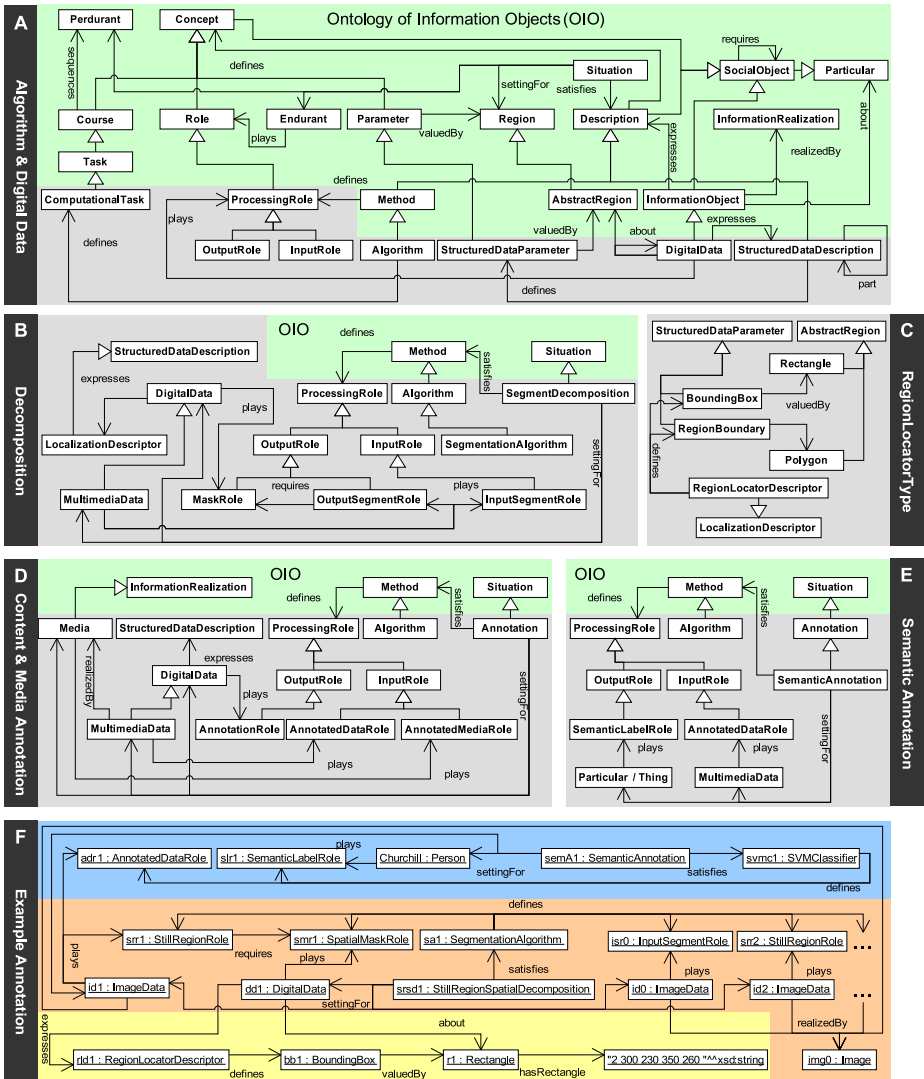
*Decomposition Pattern.* Following the D&S pattern, we consider that a decomposition of a `MultimediaData` entity is a `Situation` (a `SegmentDecomposition`) that satisfies a `Description`, such as a `SegmentationAlgorithm` or a `Method` (e.g. a user drawing a bounding box around a depicted face), which has been applied to perform the decomposition, see Fig. 2-B. Of particular importance are the `Roles` that are defined by a `SegmentationAlgorithm` or a `Method`. `OutputSegmentRoles` express that some `MultimediaData` entities are segments of a `MultimediaData` entity that plays the role of an input segment (`InputSegmentRole`). These data entities have as setting a `SegmentDecomposition` situation that satisfies the roles of the applied `SegmentationAlgorithm` or `Method`. `OutputSegmentRoles` as well as `SegmentDecompositions` are then specialized according to the segment and decomposition hierarchies of MPEG-7 ([1], part 5, section 11).

The decomposition pattern also reflects the need for localizing segments within the input segment of a decomposition as each `OutputSegmentRole` requires a `MaskRole`. Such a role has to be played by one or more `DigitalData` entities which express one `LocalizationDescriptor`. An example of such a descriptor is an ontological representation of the MPEG-7 `RegionLocatorType`<sup>7</sup> for localizing regions in an image (see Fig. 2-C, details in section 5.3). Hence, the `MaskRole` concept corresponds to the notion of a mask in MPEG-7.

*Content Annotation Pattern.* This formalizes the attachment of metadata (i.e. annotations) to `MultimediaData` (Fig. 2-D). Using the D&S pattern, `Annotations` also become `Situations` that represent the state of affairs of all related `DigitalData` (metadata and annotated `MultimediaData`). `DigitalData` entities represent the attached metadata by playing an `AnnotationRole`. These `Roles` are defined

<sup>6</sup> Sans serif font indicates ontology concepts.

<sup>7</sup> Type writer font indicates MPEG-7 language descriptors.



**Fig. 2.** COMM: Design patterns in UML notation: Basic design patterns (A), multimedia patterns (B, D, E), modeling example (C) and example annotation graph (F)

by Methods or Algorithms. The former are used to express manual (or semi-automatic) Annotation while the latter serve as an explanation for the attachment of automatically computed features, such as the dominant colors of a still region. It is mandatory that the MultimediaData entity being annotated plays an AnnotatedDataRole.

The actual metadata that is carried by a DigitalData entity depends on the StructuredDataDescription that is expressed by it. These descriptions are

formalized using the digital data pattern (see section 5.3). Applying the content annotation pattern for formalizing a specific annotation, e.g. a `DominantColorAnnotation` which corresponds to the connection of a `MPEG-7 DominantColorType` with a segment, requires only the specialization of the concept `Annotation`, e.g. `DominantColorAnnotation`. This concept is defined by being a setting for a `DigitalData` entity that expresses one `DominantColorDescriptor` (a subconcept of `StructuredDataDescription` which corresponds to the `DominantColorType`).

*Media Annotation Pattern.* This forms the basis for describing the physical instances of multimedia content (Fig. 2-D). It differs from the content annotation pattern in only one respect: it is the `Media` that is being annotated and therefore plays an `AnnotatedMediaRole`.

One can thus represent that the content of Fig. 1-A is realized by a `JPEG Image` with a size of 462848 byte, using the `MPEG-7 MediaFormatType`. Using the media annotation pattern, the metadata is attached by connecting a `DigitalData` entity with the `Image`. The `DigitalData` plays an `AnnotationRole` while the `Image` plays an `AnnotatedMediaRole`. An ontological representation of the `MediaFormatType`, namely an instance of the `StructuredDataDescription` subconcept `MediaFormatDescriptor`, is expressed by the `DigitalData` entity. The tuple formed with the scalar “462848” and the string “JPEG” is the value of the two instances of the concepts `FileSize` and `FileFormat` respectively. Both concepts are subconcepts of `StructuredDataParameter` (Fig. 2-C).

*Semantic Annotation Pattern.* Even though MPEG-7 provides some general concepts (see [1], part 5, section 12) that can be used to describe the perceivable content of a multimedia segment, independent development of domain-specific ontologies is more appropriate for describing possible interpretations of multimedia — it is useful to create an ontology specific to multimedia, it is not useful to try to model the real world within this. An ontology-based multimedia annotation framework should rely on domain-specific ontologies for the representation of the real world entities that might be depicted in multimedia content. Consequently, this pattern specializes the content annotation pattern to allow the connection of multimedia descriptions with domain descriptions provided by independent world ontologies (Fig. 2-E).

An `OWL Thing` or a `DOLCE Particular` (belonging to a domain-specific ontology) that is depicted by some multimedia content is not directly connected to it but rather through the way the annotation is obtained. Actually, a manual annotation `Method` or its subconcept `Algorithm`, such as a classification `Algorithm`, has to be applied to determine this connection. It is embodied through a `SemanticAnnotation` that satisfies the applied `Method`. This `Description` specifies that the annotated `MultimediaData` has to play an `AnnotatedDataRole` and the depicted `Thing / Particular` has to play a `SemanticLabelRole`. The pattern also allows the integration of features which might be evaluated in the context of a classification `Algorithm`. In that case, `DigitalData` entities that represent these features would play an `InputRole`.

### 5.3 Basic Patterns

Specializing the D&S and OIO patterns for defining multimedia design patterns is enabled through the definition of basic design patterns, which formalize the notion of digital data and algorithm.

*Digital Data Pattern.* Within the domain of multimedia annotation, the notion of digital data is central — both the multimedia content being annotated and the annotations themselves are expressed as digital data. We consider `DigitalData` entities of arbitrary size to be `InformationObjects`, which are used for communication between machines. The OIO design pattern states that `Descriptions` are expressed by `InformationObjects`, which have to be about facts (represented by `Particulars`). These facts are settings for `Situations` that have to satisfy the `Descriptions` that are expressed by `InformationObjects`. This chain of constraints allows the modeling of complex data structures to store digital information. Our approach is as follows (see Fig. 2-A): `DigitalData` entities express `Descriptions`, namely `StructuredDataDescriptions`, which define meaningful labels for the information contained by `DigitalData`. This information is represented by numerical entities such as scalars, matrices, strings, rectangles or polygons. In DOLCE terms, these entities are `AbstractRegions`. In the context of a `Description`, these `Regions` are described by `Parameters`. `StructuredDataDescriptions` thus define `StructuredDataParameters`, for which `AbstractRegions` carried by `DigitalData` entities assign values.

The digital data pattern can be used to formalize complex MPEG-7 low-level descriptors. Fig. 2-C shows the application of this pattern by formalizing the MPEG-7 `RegionLocatorType`, which mainly consists of two elements: a `Box` and a `Polygon`. The concept `RegionLocatorDescriptor` corresponds to the `RegionLocatorType`. The element `Box` is represented by the `StructuredDataParameter` subconcept `BoundingBox` while the element `Polygon` is represented by the `RegionBoundary` concept.

The MPEG-7 code example given in Fig. 1 highlights that the formalization of data structures, so far, is not sufficient — complex MPEG-7 types can include nested types that again have to be represented by `StructuredDataDescriptions`. In our example, the MPEG-7 `SemanticType` contains the element `Definition` which is of complex type `TextAnnotationType`. The digital data pattern covers such cases by allowing a `DigitalData` instance `dd1` to be about a `DigitalData` instance `dd2` which expresses a `StructuredDataDescription` that corresponds to a nested type (see Fig. 2-A). In this case the `StructuredDataDescription` of instance `dd2` would be a part of the one expressed by `dd1`.

*Algorithm Pattern.* The production of multimedia annotation can involve the execution of `Algorithms` or the application of computer assisted `Methods` which are used to produce or manipulate `DigitalData`. The automatic recognition of a face in an image region is an example of the former, while manual annotation of the characters is an example of the latter.

We consider `Algorithms` to be `Methods` that are applied to solve a computational problem (see Fig. 2-A). The associated (DOLCE) `Situations` represent the



work that is being done by Algorithms. Such a Situation encompasses DigitalData<sup>8</sup> involved in the computation, Regions that represent the values of Parameters of an Algorithm, and Perdurants<sup>9</sup> that act as ComputationalTasks (i.e. the processing steps of an Algorithm). An Algorithm defines Roles which are played by DigitalData. These Roles encode the meaning of data. In order to solve a problem, an Algorithm has to process input data and return some output data. Thus, every Algorithm defines at least one InputRole and one OutputRole which both have to be played by DigitalData.

#### 5.4 Comparison with Requirements

We discuss now whether the requirements stated in section 4 are satisfied with our proposed modeling of the multimedia ontology.

The ontology is **MPEG-7 compliant** since the patterns have been designed with the aim of translating the standard into DOLCE. It covers the most important part of MPEG-7 that is commonly used for describing the structure and the content of multimedia documents. Our current investigation shows that parts of MPEG-7 which have not yet been considered (e.g. navigation & access) can be formalized analogously to the other descriptors through the definition of further patterns. The technical realization of the basic MPEG-7 data types (e.g. matrices and vectors) is not within the scope of the multimedia ontology. They are represented as ontological concepts, because the **about** relationship which connects DigitalData with numerical entities is only defined between concepts. Thus, the definition of OWL data type properties is required to connect instances of data type concepts (subconcepts of the DOLCE AbstractRegion) with the actual numeric information (e.g. xsd:string). Currently, simple string representation formats are used for serializing data type concepts (e.g. Rectangle) that are currently not covered by W3C standards. Future work includes the integration of the extended data types of OWL 1.1.

**Syntactic and semantic interoperability** of our multimedia ontology is achieved by an OWL DL formalization<sup>10</sup>. Similar to DOLCE, we provide a rich axiomatization of each pattern using first order logic. Our ontology can be linked to any web-based domain-specific ontology through the semantic annotation pattern.

A clear **separation of concerns** is ensured through the use of the multimedia patterns: the decomposition pattern for handling the structure and the annotation pattern for dealing with the metadata.

These patterns form the core of the **modular** architecture of the multimedia ontology. We follow the various MPEG-7 parts and organize the multimedia ontology into modules which cover *i*) the descriptors related to a specific media type (e.g. visual, audio or text) and *ii*) the descriptors that are generic to a

<sup>8</sup> DigitalData entities are DOLCE Endurants, i.e. entities which exist in time and space.

<sup>9</sup> Events, processes or phenomena are examples of Perdurants. Endurants participate in Perdurants.

<sup>10</sup> Examples of the axiomatization are available on the COMM website.



particular media (e.g. media descriptors). We also design a separate module for data types in order to abstract from their technical realization.

Through the use of multimedia design patterns, our ontology is also **extensible**, allowing the inclusion of further media types and descriptors (e.g. new low-level features) using the same patterns. As our patterns are grounded in the D&S pattern, it is straightforward to include further contextual knowledge (e.g. about provenance) by adding Roles or Parameters. Such extensions will not change the patterns, so that legacy annotations will remain valid.

## 6 Expressing the Scenario in COMM

The interoperability problem with which our employee was faced in section 2 can be solved by employing the COMM ontology for representing the metadata of all relevant multimedia objects and the presentation itself throughout the whole creation workflow. The employee is shielded from details of the multimedia ontology by embedding it in authoring tools and feature analysis web services.

The application of the Winston Churchill face recognizer results in an annotation RDF graph that is depicted in Fig. 2-F (visualized by an UML object diagram 11). The decomposition of Fig. 1-A, whose content is represented by id0, into one still region (the bounding box of Churchill's face) is represented by the large middle part of the UML diagram. The segment is represented by the ImageData instance id1 which plays the StillRegionRole srr1. It is located by the DigitalData instance dd1 which expresses the RegionLocatorDescriptor rld1 (lower part of the diagram). Due to the semantic annotation pattern, the face recognizer can annotate the still region by connecting it with the instance Churchill of a domain ontology that contains historic Persons (upper part of Fig. 2-F).

Running the two remaining face recognizers for Roosevelt and Stalin will extend the decomposition further by two still regions, i.e. the ImageData instances id2 and id3 as well as the corresponding StillRegionRoles, SpatialMaskRoles and DigitalData instances expressing two more RegionLocatorDescriptors (indicated at the right border of Fig. 2-F). The domain ontologies which provide the instances Roosevelt and Stalin for annotating id2 and id3 with the semantic annotation pattern do not have to be identical to the one that contains Churchill. If several domain ontologies are used, the employee can use the OWL sameAs and equivalentClass constructs to align the three face recognition results to the domain ontology that is best suited for enhancing the automatic generation of the multimedia presentation.

In order to ease the creation of multimedia annotations with our ontology, we have developed a Java API 12 which provides an MPEG-7 class interface for the construction of meta-data at runtime. Annotations which are generated in memory can be exported to Java based RDF triple stores such as Sesame. For that purpose, the API translates the objects of the MPEG-7 classes into instances of the COMM concepts. The API also facilitates the implementation

---

<sup>11</sup> The scheme used in Fig. 2-F is instance:Concept, the usual UML notation.

<sup>12</sup> The Java API is available at <http://multimedia.semanticweb.org/COMM/api/>

of multimedia retrieval tools as it is capable of loading RDF annotation graphs (e.g. the complete annotation of an image including the annotation of arbitrary regions) from a store and converting them back to the MPEG-7 class interface. Using this API, the face recognition web service will automatically create the annotation which is depicted in Fig. 2-F by executing the following code:

```
Image img0 = new Image();
StillRegion isr0 = new StillRegion();
img0.setImage(isr0);
StillRegionSpatialDecomposition srsd1 = new StillRegionSpatialDecomposition();
isr0.addSpatialDecomposition(srsd1);
srsd1.setDescription(new SegmentationAlgorithm());
StillRegion srr1 = new StillRegion();
srsd1.addStillRegion(srr1);
SpatialMask smr1 = new SpatialMask();
srr1.setSpatialMask(smr1);
RegionLocatorDescriptor rld1 = new RegionLocatorDescriptor();
smr1.addSubRegion(rld1);
rld1.setBox(new Rectangle(300, 230, 50, 30));
Semantic s1 = new Semantic();
s1.addLabel("http://en.wikipedia.org/wiki/Winston_Churchill");
s1.setDescription(new SVMClassifier());
srr1.addSemantic(s1);
```

## 7 Conclusion and Future Work

We have developed COMM, an MPEG-7 based multimedia ontology, well-founded, composed of multimedia patterns. This satisfies the requirements, as they are described by the multimedia community itself, for a multimedia ontology framework. The ontology is completely formalized in OWL DL and a stable version is available with its API at: <http://multimedia.semanticweb.org/COMM/>.

The ontology already covers the main parts of the standard, and we are confident that the remaining parts can be covered by following our method for extracting more design patterns. Our modeling approach confirms that the ontology offers even more possibilities for multimedia annotation than MPEG-7 since it is interoperable with existing web ontologies. The explicit representation of algorithms in the multimedia patterns describes the multimedia analysis steps, something that is not possible in MPEG-7. The need for providing this kind of annotation is demonstrated in the use cases of the W3C Multimedia Semantics Incubator Group<sup>13</sup>. The intensive use of the D&S reification mechanism causes that RDF annotation graphs, which are generated according to our ontology, are quite large compared to the ones of more straightforwardly designed multimedia ontologies. However, the situated modeling allows COMM to represent very general annotations. Future work will improve our evaluation of the ontology, its scalability and its adequacy in the implementation of tools that use it for multimedia annotation, analysis and reasoning in large scale applications.

<sup>13</sup> <http://www.w3.org/2005/Incubator/mmsem/XGR-interoperability/>

## Acknowledgments

The research leading to this paper was partially supported by the European Commission under contract FP6-027026, Knowledge Space of semantic inference for automatic annotation and retrieval of multimedia content – K-Space and under contract FP6-026978, X-Media Integrated Project.

## References

- [1] MPEG-7: Multimedia Content Description Interface. ISO/IEC 15938 (2001)
- [2] Nack, F., Lindsay, A.T.: Everything you wanted to know about MPEG-7 (Parts I & II). *IEEE Multimedia* 6(3-4) (1999)
- [3] Halasz, F., Schwartz, M.: The Dexter Hypertext Reference Model. *Communications of the ACM* 37(2), 30–39 (1994)
- [4] Geurts, J., van Ossenbruggen, J., Hardman, L.: Requirements for practical multimedia annotation. In: *Workshop on Multimedia and the Semantic Web (2005)*
- [5] MPEG-21: Part 17: Fragment Identification of MPEG Resources. Standard No. ISO/IEC 21000-17 (2006)
- [6] Hunter, J.: Adding Multimedia to the Semantic Web - Building an MPEG-7 Ontology. In: *1<sup>st</sup> Int. Semantic Web Working Symposium*, pp. 261–281 (2001)
- [7] Troncy, R.: Integrating Structure and Semantics into Audio-visual Documents. In: *2<sup>nd</sup> Int. Semantic Web Conference*, pp. 566–581 (2003)
- [8] Troncy, R., Bailer, W., Hausenblas, M., Hofmair, P., Schlatte, R.: Enabling Multimedia Metadata Interoperability by Defining Formal Semantics of MPEG-7 Profiles. In: *1<sup>st</sup> Int. Conf. on Semantics And digital Media Technology*, pp. 41–55 (2006)
- [9] Neumann, B., Möller, R.: On Scene Interpretation with Description Logics. In: *Cognitive Vision Systems*, pp. 247–275. Springer, Heidelberg (2006)
- [10] van Ossenbruggen, J., Nack, F., Hardman, L.: That Obscure Object of Desire: Multimedia Metadata on the Web (Part I). *IEEE Multimedia* 11(4) (2004)
- [11] Nack, F., van Ossenbruggen, J., Hardman, L.: That Obscure Object of Desire: Multimedia Metadata on the Web (Part II). *IEEE Multimedia* 12(1) (2005)
- [12] Tsinaraki, C., Polydoros, P., Moumoutzis, N., Christodoulakis, S.: Integration of OWL ontologies in MPEG-7 and TV-Anytime compliant Semantic Indexing. In: *16<sup>th</sup> Int. Conference on Advanced Information Systemes Engineering (2004)*
- [13] Isaac, A., Troncy, R.: Designing and Using an Audio-Visual Description Core Ontology. In: *Workshop on Core Ontologies in Ontology Engineering (2004)*
- [14] Garcia, R., Celma, O.: Semantic Integration and Retrieval of Multimedia Metadata. In: *5<sup>th</sup> Int. Workshop on Knowledge Markup and Semantic Annotation (2005)*
- [15] Hunter, J., Armstrong, L.: A Comparison of Schemas for Video Metadata Representation. In: *8<sup>th</sup> Int. World Wide Web Conference*, pp. 1431–1451 (1999)
- [16] Oberle, D., Lamparter, S., Grimm, S., Vrandecic, D., Staab, S., Gangemi, A.: Towards Ontologies for Formalizing Modularization and Communication in Large Software Systems. *Journal of Applied Ontology* 1(2), 163–202 (2006)
- [17] Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A., Schneider, L.: *The WonderWeb Library of Foundational Ontologies (WFOL)*. Technical report, WonderWeb Deliverable 17 (2002)
- [18] Masolo, C., Vieu, L., Bottazzi, E., Catenacci, C., Ferrario, R., Gangemi, A., Guarino, N.: Social Roles and their Descriptions. In: *9<sup>th</sup> Int. Conference on Principles of Knowledge Representation and Reasoning*, pp. 266–277 (2004)

# How Service Choreography Statistics Reduce the Ontology Mapping Problem

Paolo Besana and Dave Robertson

School of Informatics, University of Edinburgh

**Abstract.** In open and distributed environments ontology mapping provides interoperability between interacting actors. However, conventional mapping systems focus on acquiring static information, and on mapping whole ontologies, which is infeasible in open systems. This paper shows that the interactions themselves between the actors can be used to predict mappings, simplifying dynamic ontology mapping. The intuitive idea is that similar interactions follow similar conventions and patterns, which can be analysed. The computed model can be used to suggest the possible mappings for the exchanged messages in new interactions. The suggestions can be evaluated by any standard ontology matcher: if they are accurate, the matchers avoid evaluating mappings unrelated to the interaction.

The minimal requirement in order to use this system is that it is possible to describe and identify the interaction sequences: the Open-Knowledge project has produced an implementation that demonstrates this is possible in a fully peer-to-peer environment.

## 1 Introduction

Most ontology mapping systems [9,15] available for the semantic web and for semantic web services focus on acquiring *static, a priori* information about mappings. Depending on the approach, matchers compare labels, ontology structures [10], use external dictionaries like WordNet to prove similarity between nodes in hierarchies [7], learn how instances are classified to find similarities between concepts [5] or combine information from different sources [4,6]. In an open and distributed environment ontology mapping systems aim at providing interoperability between interacting actors, each with possibly a different ontology. Mapping in advance, before the interactions, is unfeasible, as the agents may be still unknown. Mapping during the interactions may be computationally difficult, as many interactions with different actors can go on simultaneously.

This paper shows that the interactions between the actors can be used to predict the mappings, making the problems related to dynamic ontology mapping more tractable. The intuitive idea is that interactions follow conventions and patterns, and these patterns are repeated when similar situations arise. The patterns are extracted by analysing the interactions in order to model the relations between the terms that appear in them. If the computed model is representative of a class of interactions, then it can provide the basis for predicting the content

of exchanged messages in future interactions. A prediction is a list of suggestions for the mapping, that any standard ontology matcher can evaluate. If the predictions are accurate, the matchers can avoid evaluating mappings unrelated to the interaction, improving efficiency and decreasing ambiguity. In fact, the context of the interaction provides additional information, that can help in those cases in which the matchers do not have enough static information to distinguish the correct mapping among many possible ones.

This paper shows that, after a reasonably small number of interactions the predictor consistently provides reliable suggestions. The minimal requirement in order to use this system is that it is possible to describe and identify the interaction sequences. In principle, any system based on workflow language can provide this. Workflow systems normally are centralised, but we have recently shown, in the EU funded OpenKnowledge project [\[1\]](#) that it is possible to achieve peer-to-peer based workflow systems, as a means of web service choreography.

In this paper, we first describe, in Section [2](#) the intuitive notions of dialogue and interaction behind our work; then, in Section [3](#), we briefly discuss the alternative approaches for agents communication, introducing the OpenKnowledge peer-to-peer framework for defining and executing interactions. Section [4](#) defines the concepts and terms used in modelling the context of interactions, while Section [5](#) describes what needs to be modelled, and how to model it with an example. Section [6](#) defines what needs to be evaluated, then reports how the testing was structured and next presents and interprets the results.

## 2 Services' Interactions

Many activities require interaction between different actors: for example, in order to book a room for a conference an inquirer needs to contact a travel agency (or more than one) or directly a number of hotels.

In the simplest version, communication between two agents is a message transmitted from a sender to a receiver. According to the speech act theory, a message is a performative act that changes the state of the world [\[14\]](#). For example, a message sent from agents  $i$  to agent  $j$  to *inform* about  $\phi$  will likely change the beliefs of  $j$ , adding the belief about  $\phi$ . In our example, the following message, sent from Mr Smith's agent to the agent representing the hotel Y:

```
inform(booking, 11 Nov 2007, 15 Nov 2007, Mr Smith, single)
```

should make the hotel agent believe that a single room must be reserved for Mr Smith from the 11<sup>th</sup> to the 15<sup>th</sup> of November. Or at least this is what Mr Smith thinks. But, for example, the hotel agent may not know the meaning of `booking` or `single`, or it may use a different format for dates. To overcome this problem, either all agents that contact the hotel service must share the same ontology, which is not feasible in an open environment where agents from different backgrounds may interact, or the agents must have access to the mappings between different ontologies.

---

<sup>1</sup> [www.openk.org](http://www.openk.org)

Unfortunately, it is infeasible in an open system to precompute all mappings, as it is impossible to forecast which agents will contact the hotel service, so some (or perhaps all) mappings must be computed dynamically when the interaction takes place. Many different ontology mapping systems have been developed and tested. However, the use of these systems raises two problems, both connected to the fact that they map full ontologies. Firstly, ontologies can be large, and the process can be lengthy, making it difficult to perform at run time, possibly simultaneously with other interactions that require other mappings. Secondly, ontologies often overlap only partially, and evaluating the result of the mapping process can be hard: one mapping process between two ontologies may yield 15% of coverage, while another one between different ontologies may yield 80% coverage, but the mappings in the first case may be the ones needed for an interaction, and those in the second case may be unrelated to it (this only being knowable at interaction time).

Usually interactions are more complex than single messages. Mr Smith's agent may first check the availability of offers, or it may want to first try single and then double rooms. Moreover, the booking may require a deposit or a credit card number. This increased complexity, consisting in exchanges of messages, follows rules and conventions: as the conversation unfolds, the content of new messages is bound by the previously exchanged messages. A message failing to follow these rules would surprise the hearer as being at best off topic or even incomprehensible.

Dialogue norms and conventions appear at syntactic level: a request is normally followed by an answer, an offer by an acceptance or a rejection. They can also be found at semantic level: the topic of a conversation tends to remain consistent over a number of messages, forming a sort of "local" context to the conversation - for example when the agents are discussing about the purchase of a specific product. The intuitions about syntactic norms has prompted researchers in NLP to study the possibility of *dialogue grammars*, that have often been represented as finite state machines, where the speech acts are the transition states between admissible states of the dialogue. The use of models for dialogues has been used, for example, in dialogue translation [12].

The content of interactions (even the simple ones consisting of single messages) are also influenced by the "real world" context - agents still act on the behalf of real actors - which influences the likelihood of some specific topics (tastes and needs change with geography and time). The travel agency context binds already the possible content of the interaction (you may ask for a flight, for a holiday package, but you will unlikely ask for a laptop)

For example, after entering in a travel agency and asking for an accommodation somewhere, the clerk will likely ask the requested period, and then will try to refine the request proposing different types of accommodations (hotels, B&B, hostels, and so on). Once accommodation has been addressed, the content of the possible messages is further constrained: talks about cruises are still possible, but less likely. However, the list of proposed accommodation may change, depending on the deals that the agency has, or more general on the country and

period (for example, B&B are starting to appear in Italy, but were extremely rare until a few years ago, while they are very common in the UK).

Interactions aimed at performing tasks tend to be repeated fairly similarly every time the same task needs to be done: the structure of the interaction in the example will be repeated similarly in different travel agencies. The repetition of interactions offers the possibility of extracting patterns in the interaction, providing the basis for a representation of context. This representation can be used for focusing the mapping process.

### 3 Dialogues and Protocols

Dialogues between software agents are, at least at the moment, simpler and more restricted than those between humans: they are carried out in order to reach a goal (buying a product, booking a flight, querying a price, ...) and there is no need to care for digressions, unless relevant to the task. Therefore, their grammars can be simpler than those required for human interactions.

Interactions can be hard-coded in the agents involved in them; may be planned dynamically; or defined in workflows that are followed as a script by the agents. These approaches offer different trade-offs between flexibility and efficiency: embedding the interactions in the agents is the most inflexible but possibly very efficient. Planning is based on the the idea that speech acts can be considered as actions that change the world, and have preconditions and postconditions, usually relative to the mental state of the agents involved in the interaction [3]. It offers the maximum flexibility but may require hefty computation at every interaction, and conditions can be difficult to verify. However, as we have argued, interactions tend to be repeated, so planning them every time is a waste of resources: workflows represent a good compromise and are currently the dominant solution.

Workflows can be seen as a sort of explicit, more stringent dialogue grammar, that agents follow. Their speech acts represent the transitions (or moves) between the different state in the dialogue.

Most workflow languages represent interactions between processes, and can be formalised using process calculi (such as  $\pi$ -calculus [11]). The *Lightweight Coordination Calculus (LCC)* [13] is based on  $\pi$ -calculus and can be used as a compact way of representing workflows. It is also executable and it is adapted to peer-to-peer workflows. In the original version, protocols are declarative scripts, circulated with messages. Agents execute the protocols they receive by applying *rewrite rules* to expand the state and find the next move.

It uses roles for agents and constraints on message sending to enforce the social norms. The basic behaviours are to send ( $\Rightarrow$ ) or to receive ( $\Leftarrow$ ) a message. More complex behaviours are expressed using connectives: **then** creates sequences, or creates choices.

The protocol in Figure 1 shows the initial part of a protocol describing an interaction between a customer and a supplier of some product. In the fragment, the customer asks for a product and then the supplier verifies if the request must



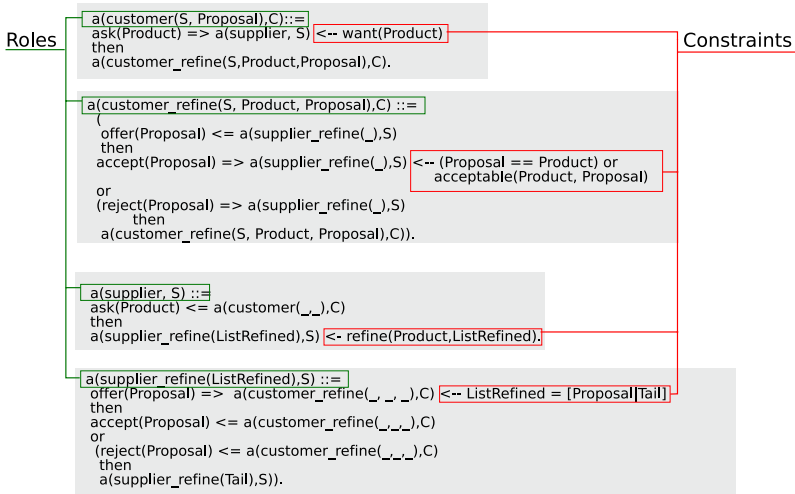


Fig. 1. Request refinement in LCC

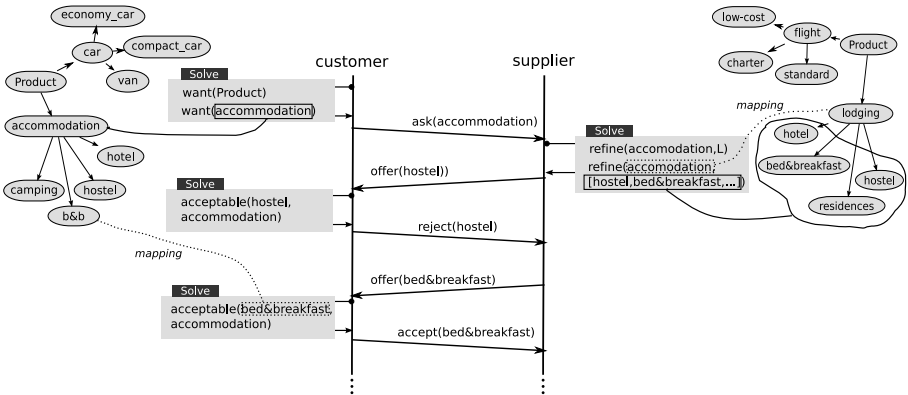


Fig. 2. Run of the protocol for searching an accommodation

be refined. If this is the case, the supplier will propose to the customer another, more specific, product. The customer, in turn, will analyse the proposal and see if it fits its needs. Figure 2 shows a run of the protocol in an interaction between a customer and a travel agency for booking an accommodation.

While the example has been kept simple for explanation purposes, a real interaction could be far more complex, involving many agents: LCC has been used in applications such as business process enactment [8] and e-science service integration [1]. In particular, it has been chosen as the specification language used for defining interaction models in OpenKnowledge, which aims at creating a Peer-to-Peer system that is open in participation, functionality and data and should allow people to easily create, find, invoke, compose and run services in a decentralised and autonomous fashion.



### 3.1 Open Knowledge Kernel

The core concept in OpenKnowledge are the interactions, defined by *interaction models* written in LCC and published by the authors on the *distributed discovery service* with a keyword-based description. The roles in the interaction models are played by peers. The peers that want to perform some tasks, such as booking a room or provide a booking service, use keyword queries to search for published interaction models for the task, and then advertise their intention of interpreting one of its roles to the discovery service. In the running example, a travel agency subscribes to perform role `supplier`, while a peer searching a room subscribes as `customer`.

When all the roles are filled, the discovery service chooses randomly a peer in the network as coordinator for the interaction, and hands over the interaction model together with the list of involved peers in order to execute it.

The coordinator then asks each peer to commit to the interaction. If they all commit, the coordinator executes the interaction instantiating a local proxy for each peer: the peers are contacted to solve constraints in the role they have subscribed. In the example protocol, the coordinator will ask the peer that has subscribed as `customer` to solve `want(Product)`.

## 4 Predicting the Content of Messages

A message in an interaction is a tuple, whose elements convey the content of a single communication act:

$$m_i = \langle s_1, \dots, s_n \rangle$$

A term  $s_i$  is introduced when a constraint in a role is satisfied by one of the actors playing the role (in the example shown in Figure 2, “*accommodation*” is introduced by the customer peer, satisfying the constraint `want(Product)`). The term  $s_i$  is defined in the ontology of that agent, and refers to an entity  $Q_k$ . The other agents, if they need to satisfy a constraint that contains  $s_i$ , will need to find the term  $t_m$  in their ontology that refers to the same, or a similar, entity  $Q_k$  (in the example, in order to satisfy the constraint `refine(Product, ListRefined)`, the supplier must map the term “*accommodation*” to “*lodging*” in its ontology). The mapping is performed by a “mapping oracle”, whose specific implementation is irrelevant for this work: any existing mapping system, such as S-Match [7], would fit smoothly in the framework.

Let us suppose that a peer, with ontology  $L_a$ , needs to satisfy a constraint  $\kappa_r(\dots, w_i, \dots)$  when in a specific state of an interaction, and that  $w_i \notin L_a$  is the foreign term received in some previous message  $m_j$ . The task of the oracle is to find what entity  $Q_k$ , represented in the agent’s ontology by the term  $t_m \in L_a$ , was encoded in  $w_i$ .

**Definition 1.** *The intended entity  $Q_k$  represented in the argument of the constraint by the foreign term  $w_i$  is, from the agent’s perspective, a random variable, whose domain is the whole ontology.*

As said before, an ontology mapping algorithm can be used to interpret the sign  $w_i$  in the message and map it to the corresponding symbol  $t_m$ .

**Definition 2.** *The term  $t_m$  is the matching term: it is, in the agent's ontology, the closest to the intended entity  $Q_k$ . For the current work, the matching term is assumed to exist in  $L_a$ . The assumption is based on the weakness of the relation between  $t_m$  and  $Q_k$ : it is sufficient that the meanings are close enough to perform the interaction.*

However, conventional ontology mapping algorithms do not take into account the context of the interaction, and consider all the terms in the domain as equiprobable:

$$p(Q_k = t_i) = p(Q_k = t_j) \text{ for } \forall t_i, t_j \in L_a$$

As introduced earlier, dialogues follows conventions and rules, made explicit by the protocol, and the content of the messages are influenced by the local and the general context: therefore the terms are not equiprobable - some will be more likely than others.

**Definition 3.** *The random variable  $Q_k$  has a conditional probability distribution, where the evidence is the context of the interaction (we discuss context in Section 5):*

$$P(Q_k | \text{context}) = \langle \dots, p(Q_k = t_i | \text{context}), \dots \rangle \text{ for } t_i \in L_a$$

where  $p(Q_k = t_i | \text{context})$  is the probability that  $t_i$  is the best matching term for  $Q_k$  given the current state of the interaction and the history of previous interactions.

The knowledge of the probability distribution of a variable is used to predict the possible values of  $Q_k$  selecting a subset of likely terms to be verified by the oracle, improving the efficiency and the results of the ontology mapping systems, and making it more feasible to be performed at runtime.

**Definition 4.** *A subset  $\Lambda \subseteq L_a$  is a set of terms containing the most likely terms for a random variable  $Q_k$ . The probability that the correct matching term  $t_m$  belongs to  $\Lambda$  is:*

$$p(t_m \in \Lambda) = \sum_{t_i \in \Lambda} p(Q_k = t_i | \text{context})$$

For  $\Lambda = L_a$ , this probability is 1. If the distribution is uniform, then a set  $\Lambda$  of size  $\gamma |A|$ , with  $0 \leq \gamma \leq 1$ , will contain  $t_m$  with probability  $p(t_m \in \Lambda) = \gamma$ . If the distribution is non-uniform, then even for smaller resizing factor  $\gamma$  it is possible to obtain high probabilities  $p(t_m \in \Lambda)$ : it becomes useful to trade off between the size of the set  $\Lambda$  and the probability of finding the correct mapping. To select the terms to insert in  $\Lambda$ , it is necessary to set a threshold  $\tau < 1$  for  $p(t_m \in \Lambda)$ . If the terms are ordered from the most probable to the least one, then this means solving the equation in  $n$ :

$$\tau \leq \sum_1^n p(t_j)$$

That simply means taking the first  $n$  most likely terms until their cumulative probability is equal or greater than  $\tau$ .

## 5 Modelling the Context

### 5.1 What to Model

An interaction is an exchange of messages, where the content of the messages comes from satisfying constraints. A peer satisfying a constraint is responsible for the introduction of terms related to the interaction: failure to do so disrupts the communication. If the travel agency peer, after being asked for an accommodation, satisfies the constraint `refine(Product, ListRefined)` with a choice of possible types of coffee, then the communication loses meaning. Therefore, what the predictor should model are constraints. Intuitively, constraints fall into three main categories:

- *Purely functional*: given a set of parameters, they always unify with the same values: for example *multiply*( $X, Y, Z$ ) is supposed to always unify the variables with the same numbers.
- *Purely “preference-based”*: they collect requests from users and their possible values can differ every time. In the example, the constraint `want(Product)` is preference-based: each peer will satisfy it according to its tastes and needs. Overall, the variables in preference-based constraints will have an unknown distribution. These distributions may change with time, depending on general shifts of “tastes” and “needs” (fashions, trends, fads, ...) or the heterogeneity in the peer group composition, and can be more or less biased.
- *Mixed*: they can be mainly functional, but the results may change depending on external factors (availability, new products appearing on the market, etc), or can be mainly preference-based, but constrained by some other parameters. In the example, the constraint `refine(Product, ListRefined)` is mainly functional, as it returns the list of possible subclasses of a term if the query can be refined. The list of terms can however change depending on the specific peer and with time.

A purely functional constraint, when the function is ontological (obtain subclasses, or siblings, or properties), can be guessed and the hypotheses can be verified comparing the guesses with the feedback from the ontology matching process. For the purely preference based, it is possible to count the frequencies of the terms and learn their prior probability distribution. For the mixed, it is possible to use a mix of hypotheses and counting the frequencies. Sometimes the ontology of the peer does not allow him to formulate the correct ontological relation (because the ontology is structured differently from the agent that introduced the term): it is still possible to count the conditional frequencies, modelling the relation from a purely statistical point of view.

### 5.2 How to Model

Our solution, suggested but not evaluated in [2], is a model of the interaction in which the properties of entities appearing in the random variable  $Q_k$  in different runs of the same protocol are counted and stored in *assertions*:

**Table 1.** Types of assertions

Frequency of terms:  $Freq(E_i \in \{t_q\})$

Assertions can be about the frequency of the entities in an argument, disregarding the content of other variables in the dialogue, like  $A_{1-5}$  in Table 2.

Conditional frequency of terms:  $Freq(E_i \in \{t_q\} | E_k = t_h)$

More precise assertions can be about the frequency of an entity given the content of previously encountered variables, like  $A_{6-10}$  in Table 2.

Frequency of relations with terms in other variables:  $Freq(E_i \in \{X | rel(X, E_k)\})$

They can regard the relation with an argument of another variable in the protocol, like  $A_{11-12}$  in Table 2.

Frequency of relations with terms in ontology:  $Freq(E_i \in \{X | rel(X, t_k)\})$

They can be about an ontological relation between the entity in the argument and an entity  $t_k$  in the agent’s ontology, like  $A_{13}$  in Table 2.

**Definition 5.** *An assertion about a random variable  $Q_k$  keeps track of the frequency with which the entity has been part of a set  $\Psi$  in the encountered dialogues:*

$$A_j \doteq Freq(Q_k \in \Psi) \quad (1)$$

Assertions can be about frequencies of terms in the variable, or can be about the frequencies of ontological relations between one variable and another, as described in Table 1.

For example, the customer peer, having executed the interaction in Figure 1 a number of times with different types of service providers, will have a table with assertions about the content of the variable  $Proposal_k$  in the form shown in Table 2. As protocols can be recursive, the variables are tagged with their appearance in the run (in the example, the variable  $Proposal$  is used twice, so there will be two random variables named  $Proposal_1$  and  $Proposal_2$ ).

When the content of a variable must be predicted, the assertions relative to it are instantiated with the current state of the the interaction. In the interaction shown in Figure 2, in order to predict the content of  $Proposal_2$  (received in the second offer sent by the travel agency), given that  $Product_1$  was instantiated to “accommodation” and  $Proposal_1$  was instantiated to “hotel”, it is necessary:

1. to drop the conditional assertions whose evidence does not correspond to the current state of the interaction; so assertions  $A_{9-10}$  are dropped because their evidence  $Product_1 = "car"$  is inconsistent with the current interaction,
2. to unify the variables in relations with the current state of the interaction ;  $Product_1$  in  $A_{11}$  is replaced with “accommodation” and  $Proposal_{k-1}$  in  $A_{12}$  is replaced with “hotel”, obtaining:

$A_{11}) Freq(Proposal_2 \in \{Proposal_2 : subclassOf(Proposal_2, "accommodation")\})$

$A_{12}) Freq(Proposal_2 \in \{Proposal_2 : siblingOf(Proposal_2, "hotel")\})$

3. the relations are computed, obtaining sets of terms; so  $A_{11-13}$  becomes:

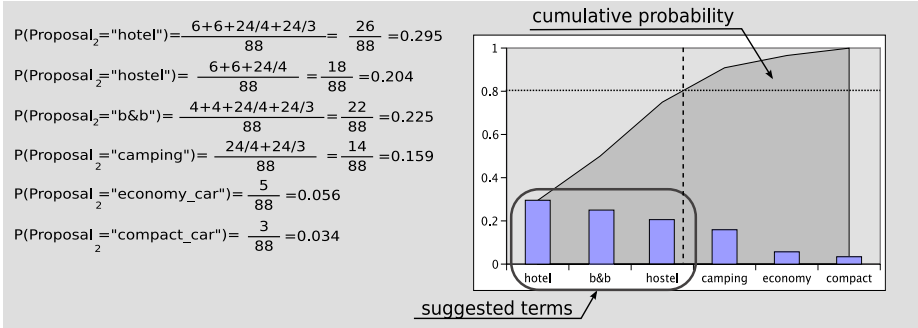
$A_{11}) Freq(Proposal_2 \in \{ "hotel", "hotel", "b&b", "camping" \}) = 24$

$A_{12}) Freq(Proposal_k \in \{ "hotel", "b&b", "camping" \}) = 24$

$A_{13}) Freq(Proposal_k \in \{ "accommodation", "hotel", ..., "car", ..., "van" \}) = 24$

**Table 2.** Statistical model of the context for the customer peer

- $A_1) Freq(\text{Proposal}_k \in \{\text{"hotel"}\}) = 6$   
 $A_2) Freq(\text{Proposal}_k \in \{\text{"hostel"}\}) = 6$   
 $A_3) Freq(\text{Proposal}_k \in \{\text{"b\&b"}\}) = 4$   
 $A_4) Freq(\text{Proposal}_k \in \{\text{"compact\_car"}\}) = 3$   
 $A_5) Freq(\text{Proposal}_k \in \{\text{"economy\_car"}\}) = 5$   
 $A_6) Freq(\text{Proposal}_k \in \{\text{"hotel"}\} | \text{Product}_1 = \text{"accommodation"}) = 6$   
 $A_7) Freq(\text{Proposal}_k \in \{\text{"hostel"}\} | \text{Product}_1 = \text{"accommodation"}) = 6$   
 $A_8) Freq(\text{Proposal}_k \in \{\text{"b\&b"}\} | \text{Product}_1 = \text{"accommodation"}) = 4$   
 $A_9) Freq(\text{Proposal}_k \in \{\text{"compact\_car"}\} | \text{Product}_1 = \text{"car"}) = 3$   
 $A_{10}) Freq(\text{Proposal}_k \in \{\text{"economy\_car"}\} | \text{Product}_1 = \text{"car"}) = 5$   
 $A_{11}) Freq(\text{Proposal}_k \in \{\text{Proposal}_k : \text{subClassOf}(\text{Proposal}_k, \text{Product}_1)\}) = 24$   
 $A_{12}) Freq(\text{Proposal}_k \in \{\text{Proposal}_k : \text{siblingOf}(\text{Proposal}_k, \text{Proposal}_{k-1})\}) = 24$   
 $A_{13}) Freq(\text{Proposal}_k \in \{\text{Proposal}_k : \text{subClassOf}(\text{Proposal}_k, \text{"product"})\}) = 24$

**Fig. 3.** Probability distribution for variable **Proposal**

The result of the third step is that some of the assertions assign probabilities to possibly large and overlapping sets. The frequencies assigned to sets are uniformly distributed among the members: according to the *principle of indifference* the frequency of mutually exclusive elements in a set should be evenly distributed. However, assertions about ontological relations create two main problems. First, some of the relations can be spurious. Second, some relations may refer to large sets, bringing little information (like assertion  $A_{13}$  in the example). To deal with the first issue, only relations found in a significant proportion of the cases are taken into consideration. To deal with the second sets larger than a significant portion of the ontology are discarded. Tests have shown that a threshold for the first issue of 10% and of 20% for the second one minimise the problem.

Finally, the probability that an entity  $t_i$  is used for  $Q_k$  is computed by summing the frequencies in all the instantiated assertions in which  $t_i$  appears, divided by the sum of the frequencies of all the selected assertions:

$$p(t_i) = \frac{\sum A_j(t_i \in \Psi)}{\sum A_k} \quad (2)$$

In the example, to compute the probability that the concept in `Proposal2` is the term “*hotel*”, the numerator contains the assertions  $A_1, A_6, A_{11}, A_{12}$ . The assertions  $A_{11-12}$  contain more than one element, and therefore the frequency assigned to “*hotel*” is computed dividing the frequency assigned to the set by the size of the set to obtain the following:

$$P(\textit{hotel}) = \frac{6+6+24/4+24/3}{6+6+4+3+5+6+6+4+24+24} = \frac{26}{88} = 0.295$$

The complete distribution of variable  $P(\textit{Proposal}_2 = \textit{“hotel”} | \textit{Context})$  is shown in Figure 3

## 6 Evaluation

The predictor is characterised by its average success rate,  $E [P_Q(t_m \in \Lambda)]$ , and the average size of the suggested set  $\Lambda$ ,  $E [|\Lambda|]$ . Let us assume to have the exact probability distribution  $\overline{\mathbf{P}}(Q_k | \textit{context})$  of the terms for a random variable  $Q_k$  given the current context. The correct size  $n$  of  $\Lambda$  in order to obtain the desired probability of finding  $t_m$  is:

$$\tau = \sum_1^n \overline{p}(t_j)$$

If the computed distribution  $\mathbf{P}(Q_k | \textit{context})$  is a good approximation of  $\overline{\mathbf{P}}(Q_k | \textit{context})$ , then the average of  $p(t_m \in \Lambda)$  should converge towards the average of  $\overline{\mathbf{P}}(Q_k | \textit{context})$  and therefore towards the threshold  $\tau$ :

$$\lim_{\textit{iterations} \rightarrow \infty} E [p(t_m \in \Lambda)] = E [\overline{p}(t_m \in \Lambda)] = \tau \quad (3)$$

If the success rate of the predictor remains lower than the threshold  $\tau$ , independently of the number of interactions, then the computed distribution is different from the exact  $\overline{\mathbf{P}}(Q_k | \textit{context})$ .

A key issue to evaluate is the number of repeated interactions needed for the predictor to reach a stable behaviour. This number will be different for every type of interaction, but what is necessary is to find its probability distribution: what is the probability that  $n$  interactions are enough to have a stable behaviour. Once in the stable region, the predictor will go on updating its representation, but the behaviour should change slowly or remain constant.

The size of the suggested set  $\Lambda$  will depend on the existence relations between variables in the interaction and on the unknown distribution of terms in preference-based constraints, as we have seen in Section 5.1. These unknown distributions can change over time - if the phenomena are non-stationary - obviously decreasing the success rate. The lack of relations or flat distributions will cause large suggestions sets  $\Lambda$ .

### 6.1 Testing

One way of testing is through real interaction scenarios, using real ontologies and real workflows for the dialogues, but since these are scarce this would cover only

$$\begin{aligned}
& a(r8a(O), I) ::= \\
& m_1(X, P) \Rightarrow a(r8b, O) \leftarrow \kappa_1(P, X) \\
& \text{then} \left( \begin{array}{l} m_2(Y) \Leftarrow a(r8b, O) \\ \text{or} \\ m_3(M) \Leftarrow a(r8b, O) \end{array} \right) \\
& a(r8b, O) ::= \\
& m_1(X, P) \Leftarrow a(r8a(\_), I) \\
& \text{then} \left( \begin{array}{l} m_2(Y) \Rightarrow a(r8a, O) \leftarrow \kappa_2(P, X, Y) \\ \text{or} \\ m_3(M) \Rightarrow a(r8b, O) \leftarrow \kappa_3(P, X, M) \end{array} \right)
\end{aligned}$$

**Fig. 4.** Protocol template

part of the testing space, without having the possibility of varying parameters to verify the effects.

What is important, however, is to verify the ability of the predictor in statistically modelling the way constraints are satisfied given the state of the interaction. And, as we have seen in Section 5.1, the constraints can be *functional*, *preference-based*, or *mixed*. It is thus possible to simulate different real world scenarios using template protocols executed by dummy peers that can only satisfy constraints according to parametrisable rules and ontologies.

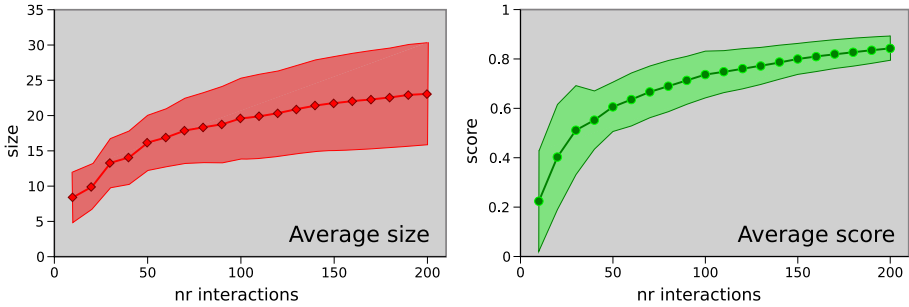
The template protocols must cover the basic patterns in interactions. The functional constraints are ontological rules, the preference-based constraints return terms according to probability distributions that reflect the distribution of “needs” and “tastes” over a community of peers, and mixed constraints are rules with an element of probability.

For example, the protocol in Figure 4 can model many different interactions:  $m_1$  can be a request for information  $X$  about  $P$  (for example, the price of a  $X$ ), with  $m_2$  being the reply and  $m_3$  being the apology for failing to know the answer. Alternatively,  $m_1$  can be an offer (the product  $X$  at price  $P$ ), with  $m_2$  being the acceptance and  $m_3$  the rejection. By viewing interaction protocols abstractly we can set up large scale experiments in which we vary the forms of constraints in a controlled way.

Testing has involved a number of different abstract protocols with different possible relations between the terms in variables. The protocols were run in different batches, with each of them consisting of 200 runs of the protocol. Each batch is characterised by a specific ontology (with a hierarchical depth) and a set of preference distributions. Every 10 runs, the average size and the average score are stored in order to obtain a graph representing the improvement of the results over time.

## 6.2 Results

The results shown in Figure 5 were obtained averaging the results of 12 different batches, generated combining 6 protocols, 3 ontologies (225, 626 and 1850 elements) and different settings for the preference distributions (narrow and wide



**Fig. 5.** Average size and average score

distributions for the preference-based constraints). All the batches were run with a threshold  $\tau = 0.8$ . The figure shows the average value of the size of the suggested set  $A$  and the average value of  $p(t_m \in A)$ , together with a band specifying the standard deviation of the measure. The limit in Formula 3 is verified, as the average score tends to stabilise, logarithmically, around  $\tau$  (the standard deviation, showing fluctuations in success rate, decreases).

The average size remains small, independently of the size of the ontology, but its deviation tends to increase - albeit only logarithmically and remains well below 15% of the smaller ontology. The relatively large deviation reflects the fact that different batches have different relations between variable, and preference-based constraints have different distributions: therefore to obtain the same success rate the size of  $A$  may change meaningfully. However, the use of the filters on the assertions, described in Section 5.2, improved the results substantially: previous tests run on the same batches before the introduction of the filters returned the same average score, but a much higher average size (more than 150 elements instead of about 20).

The learning curve is, as stated, logarithmic: on average, most improvement (from 0 to nearly 70%) is obtained in the first 70-80 interactions, which is a small number of interactions in large peer-to-peer communities as those envisioned in the OpenKnowledge project. In the example scenario, the travel agency peer can be contacted by a thousand peers, all making similar requests, while the customer may need to contact several travel agencies before finding an proper accommodation.

## 7 Conclusion

This paper has shown that it is possible to use the interactions between peers in an open environment to statistically model and then predict the possible content of exchanged messages. The predictions can be forwarded to an ontology matching algorithm that focuses its computational effort on verifying the suggested hypotheses, without wasting time on evaluating mappings not related to the interaction.



The evaluation of the proposed method shows that a relatively small number of interaction is often enough to obtain a good success rate in the suggestions, especially when it is possible to detect ontological relations between terms appearing in the conversation.

The main requirement is to use a framework that allows the description of the interaction sequence: workflow based systems provide the functionality, but are often centralised. With the European project OpenKnowledge we have shown that these results can be obtained in a purely peer-to-peer environment.

## References

1. Barker, A., Mann, B.: Agent-based scientific workflow composition. In: Díaz, J., Orejas, F. (eds.) CAAP 1989 and TAPSOFT 1989. LNCS, vol. 351, pp. 485–488. Springer, Heidelberg (1989)
2. Besana, P., Robertson, D.: Probabilistic dialogue models for dynamic ontology mapping. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, Springer, Heidelberg (2003)
3. Cohen, P.R., Levesque, H.J.: Rational interaction as the basis for communication. *Intentions in Communication*, 221–256 (1990)
4. Do, H.H., Rahm, E.: Coma - a system for flexible combination of schema matching approaches. In: VLDB, pp. 610–621 (2002)
5. Doan, A., Madhavan, J., Dhamankarse, R., Domingos, P., Halevy, A.: Learning to match ontologies on the semantic web. *The VLDB Journal* 12(4), 303–319 (2003)
6. Ehrig, M., Staab, S.: Qom - quick ontology mapping. In: International Semantic Web Conference, pp. 683–697 (2004)
7. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: S-match: an algorithm and an implementation of semantic match. In: Proceeding of the European Semantic Web Symposium, pp. 61–75 (2004)
8. Guo, L., Robertson, D., Chen-Burger, Y.: A novel approach for enacting the distributed business workflows using bpel4ws on the multi-agent platform. In: IEEE Conference on E-Business Engineering, pp. 657–664. IEEE Computer Society Press, Los Alamitos (2005)
9. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. *The Knowledge Engineering Review* 18(1), 1–31 (2003)
10. Melnik, S., H., Garcia-Molina, E.R.: Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In: ICDE 2002. Proceedings of the 18th International Conference on Data Engineering, p. 117 (2002)
11. Puhlmann, F., Weske, M.: Using the pi-calculus for formalizing workflow patterns. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 153–168. Springer, Heidelberg (2005)
12. Reithinger, N., Engel, R., Kipp, M., Klesen, M.: Predicting dialogue acts for a speech-to-speech translation system. In: Proc. ICSLP 1996, vol. 2, pp. 654–657 (1996)
13. Robertson, D.: A lightweight coordination calculus for agent systems. In: Declarative Agent Languages and Technologies, pp. 183–197 (2004)
14. Searle, J.R.: *Speech acts: an essay in the philosophy of language*. Cambridge University Press, Cambridge (1969)
15. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. *Journal on Data Semantics* 4, 146–171 (2005)

# Kernel Methods for Mining Instance Data in Ontologies<sup>★</sup>

Stephan Bloehdorn<sup>1</sup> and York Sure<sup>2</sup>

<sup>1</sup> Institute AIFB, University of Karlsruhe,  
D-76128 Karlsruhe, Germany  
bloehdorn@aifb.uni-karlsruhe.de

<sup>2</sup> SAP AG, Research Center CEC Karlsruhe,  
Vincenz-Prießnitz-Str. 1, D-76131 Karlsruhe, Germany  
york.sure@sap.com

**Abstract.** The amount of ontologies and meta data available on the Web is constantly growing. The successful application of machine learning techniques for learning of ontologies from textual data, i.e. mining *for* the Semantic Web, contributes to this trend. However, no principal approaches exist so far for mining *from* the Semantic Web. We investigate how machine learning algorithms can be made amenable for directly taking advantage of the rich knowledge expressed in ontologies and associated instance data. Kernel methods have been successfully employed in various learning tasks and provide a clean framework for interfacing between non-vectorial data and machine learning algorithms. In this spirit, we express the problem of mining instances in ontologies as the problem of defining valid corresponding kernels. We present a principled framework for designing such kernels by means of decomposing the kernel computation into specialized kernels for selected characteristics of an ontology which can be flexibly assembled and tuned. Initial experiments on real world Semantic Web data enjoy promising results and show the usefulness of our approach.

## 1 Introduction

The standardization of the ontology languages RDF(S) [1] and OWL [2] has led to an ever increasing amount of available semantic annotations. As of August 2007, the statistics of the Semantic Web search engine *Swoogle* [3] count a total of 1,238,295 publicly available “error-free pure Semantic Web Documents”. Research has actively addressed the problem of learning knowledge structures – mostly from text data – *for* the Semantic Web, a field commonly referred to as *Ontology Learning* [3]. Only little work has been directed towards the question on learning *from* Semantic Web data and principled approaches are still missing. We strongly believe that mining Semantic Web data will become a crucial issue to deal with the massively growing amount of Semantic Web data. The need for mining Semantic Web data may arise in the context of two scenarios.

---

<sup>★</sup> This work was funded by the X-Media project ([www.x-media-project.org](http://www.x-media-project.org)) sponsored by the European Commission as part of the Information Society Technologies (IST) programme under EC grant number IST-FP6-026978.

<sup>1</sup> <http://swoogle.umbc.edu/>

On the one hand, we expect that it will be tempting to extend the scope of all kinds of adaptive systems we already find on the current Web (such as personal recommendation systems) from the still dominating document (text) data to Semantic Web data. On the other hand, classifiers that act on semantic web data can complement the crisp reasoning procedures of current Semantic Web type systems possibly leading to hybrid systems where logic-based reasoning procedures interact with machine learning techniques inspired by statistical notions.

In this paper, we investigate the question how machine learning algorithms can be made amenable to work on instances that are described by means of an ontological vocabulary and how we can exploit the rich knowledge encoded in the respective ontologies. We will do so by placing the problem of learning from Semantic Web data in the context of the field of *kernel methods*. Kernel methods (see e.g. [4] for excellent and comprehensive introductions) are one of the most prominent paradigms in modern machine learning research. While Support Vector Machines can safely be regarded as the best known kernelized learning algorithm, many other well-known supervised and unsupervised machine learning algorithms can be kernelized as well. The main idea behind kernel-based learning algorithms is that they express the learned hypothesis by means of linear combinations of a specific type of similarity functions, the so-called kernel functions, where one argument is fixed to some training data item. An intriguing property of kernel-based learning algorithms is that kernel functions need not be restricted on vector-type data as arguments but can be defined directly on data items of arbitrary type as long as some mild restrictions are ensured. This allows to directly work on heterogenous and interconnected data that does not have a natural vector-style representation. Rephrasing the earlier statement we can thus say that this paper investigates how machine learning algorithms can be made amenable for taking advantage of the knowledge expressed in ontologies and associated metadata *by designing kernel functions defined on instances that reference certain ontologies*. We introduce a framework for kernel design on Semantic Web-type data that (i) builds on common notions of similarity, (ii) ensures the validity of the kernel(s) regardless of parameter choices of the user.

This paper is organized as follows: we introduce a number of preliminaries on kernel methods as well as knowledge representation and review related work in Section 2. In Section 3, we introduce our framework for kernel design on ontological instances. We illustrate the instantiation of this framework in Section 4 in the context of two experiments based on the SWRC and GALEN ontologies. We conclude in Section 5.

## 2 Preliminaries

In this section we shortly review the main preliminary notions necessary for the understanding of the main technical contributions of the paper. We give a brief overview of kernel methods and of knowledge representation with ontologies in sections 2.1 and 2.2 respectively. The interested reader is pointed to [4] for comprehensive introductions to the field of kernel methods and to [5,6] for introductions to ontologies and knowledge representation, in particular description logic-type approaches.

## 2.1 Machine Learning with Kernel Methods

Kernel methods are powerful machine learning techniques that have widespread adoption in the machine learning community and have been shown to be powerful learning algorithms in various standard and non-standard learning settings. The major paradigm behind kernel methods is the decoupling of the employed learning algorithms from the representations of the data instances under investigation. Different learning algorithms for various tasks can be “kernelized” such as, for example, Support Vector Machines [7] for classification and regression (i.e. supervised learning tasks) or Kernel-kMeans and Kernel-PCA [8] for clustering and dimensionality reduction (i.e. unsupervised learning tasks). In the “unkernalized” variant, these algorithms operate on simple vectors of real numbers. The hypotheses generated by these algorithms are typically tied to a geometric interpretation within the corresponding vector space such as the notion of a separating hyperplane in the case of classification with Support Vector Machines. By virtue of the design of the algorithms of interest, the input vectors need not be accessible directly by them. Instead, it is sufficient that they are able to access the evaluations of the inner product  $\langle x, y \rangle$  of two vectors  $x, y$  in this space. The resulting hypotheses are then expressed using linear combinations of the input objects. In the context of this paper, we omit too much technical detail on this approach, also referred the *dual* representation of the respective algorithms.

The second component of kernel methods is the so-called *kernel function*. The kernel function computes the similarity of data instances in such a way that it is equivalent to an inner product in some (possibly unknown) vector space.

**Definition 1 (Kernel Function).** *Any function  $\kappa : X \times X \rightarrow \mathbb{R}$  that for all  $x, z \in X$  satisfies  $\kappa(x, z) = \langle \phi(x), \phi(z) \rangle$ , is a valid kernel, whereby  $X$  is some input domain under consideration and  $\phi$  is a mapping from  $X$  to some inner product space  $F$ , called the feature space.*

As such, the computations of the plain inner product in the unkernalized algorithms can be replaced by any valid kernel function. The kernel function, which can be regarded as a function that encodes a particular notion of similarity of data items of the input domain, simultaneously serves three purposes: (i) it provides the interface between the learning algorithm and the data, which is particularly interesting for data items that do not take the traditional form of vectors, (ii) it can leverage the performance of the algorithm by incorporating prior knowledge about the problem domain, and (iii) its evaluation might be computationally advantageous compared to an explicit construction of the feature space in terms of memory and/or computation requirements. The kernel function as such thus becomes an interesting subject of research. However, restrictions apply on the choice of the employed function to make it a valid kernel, namely, that the function needs to be a positive semi-definite function. Constructing appropriate positive semi-definite kernels from arbitrary data is thus not a trivial task. However, several closure properties aid the construction of valid kernels from known valid kernels, which will also be exploited in the construction of kernels in section 3. In particular, kernels are closed under sum, product, multiplication by a positive scalar and combination with well-known kernel modifiers.

Well-known kernel modifiers for a valid kernel  $k(x, y)$  on some input set  $x, y \in \mathcal{X}$  are, among others: (i) the normalisation kernel which scales the kernel results to  $[0, 1]$  (in analogy with the cosine of two vectors):

$$k_{normalised}(x, y) = \frac{k(x, y)}{\sqrt{k(x, x)k(y, y)}};$$

or (ii) the Gaussian kernel:

$$k_{gaussian}(x, y) = \exp\left(-\frac{k(x, x) - 2k(x, y) + k(y, y)}{2\sigma^2}\right), \quad \sigma \in \mathbb{R}^+.$$

As we will often be occupied with dealing with sets of objects we introduce here two well-known kernels defined on sets  $A, B \subseteq \mathcal{X}$  of data items  $x \in \mathcal{X}$ : (i) the intersection kernel

$$k_{\cap}(A, B) = |A \cap B|;$$

and (ii) the crossproduct kernel:

$$k_{\times}(A, B) = \sum_{x_a \in A} \sum_{x_b \in B} k_{base}(x_a, x_b),$$

where  $k_{base}(\cdot, \cdot)$  is any valid kernel defined on  $\mathcal{X}$ . In particular, note that the crossproduct kernel boils down to the intersection kernel when used with the so-called matching kernel:

$$k_{matching}(x_i, x_j) = \delta(x_i, x_j)$$

where  $\delta(x_i, x_j) = 1$  if  $x_i = x_j$  and  $\delta(x_i, x_j) = 0$  otherwise. Proofs for the presented kernel closure properties, kernel modifiers and set kernels are omitted but can for example be found in [4].

## 2.2 Ontologies and Knowledge Representation

Ontologies provide a shared and common understanding of a domain of discourse and provide a vocabulary to describe data instances. A popular knowledge representation formalism is the family of Description Logics (DLs) which allow to model the relevant properties of a domain by means of classes (unary predicates), which denote sets of individuals, object properties (binary predicates), which denote binary relationships between individuals, and datatype properties (binary predicates) which denote relations between individuals and specific datatypes. Terminological axioms in the ontology can combine and relate classes and roles by the use of various concept and role constructors. Assertional axioms in associated knowledge bases make statements about the instances of the domain. The semantics of a certain description logic is typically given as *model theoretic semantics* by relating the syntax of the logic and the models of a domain.

The practical importance of description logics stems from the fact that they form the basis of the Web Ontology Language OWL [2]. In particular, the sublanguage OWL DL is based on the description logic *SHOIN(D)* for which a number of practical reasoning algorithms are known and implemented. As an adopted Semantic Web standard, OWL

is increasingly used to define ontologies and these ontologies – in turn – are increasingly used to describe data instances such as people, publications, services and the like.

The kernels presented in the next section will be based on probing instances for membership in certain classes or properties, during this process, deductions made using terminological axioms may enrich the explicitly asserted facts with deduced facts. During the remainder of this paper it is important to keep in mind that, due to the open world reasoning of DLs, the deduced facts are only the necessary facts. Other facts may or may not hold but they may just not be deduced necessarily.

### 2.3 Related Work

Research on learning algorithms for data that comes in a logic-based representation is mostly rooted in the field of *Inductive Logic Programming (ILP)* [9]. ILP algorithms directly exploit the underlying logic and search for logical concept descriptions that accurately describe the concept(s) hidden in the data. As a positive side-effect, the resulting concept descriptions can thus be directly incorporated in the overall knowledge representation framework. As such they do, however, mostly completely circumvent any statistical notions such that softer hidden concepts are hard to identify.

Research on kernels for structured data, i.e. for data that is expressed in a formalism different from the standard vectorial representation, is a young research field that has recently become a major topic of investigation in the machine learning community. A good and fairly recent overview of basic types of kernels for structured data can be found in [10]. In the following, we will shortly describe the main works related to our research. Most of the work on kernels for structured data is rooted in the idea of the *convolution kernel* [11]. The main argument behind this kernel is that the semantics of composite objects can often be based on the semantics of its parts. The kernel thus aims to first evaluate basic kernels on the combinations of parts and sum the respective comparisons in the overall kernel, a direction that we will also investigate in this paper. Recently, [12] proposed a logic-based kernel on individuals represented as (closed) terms in a typed higher-order logic. Neglecting the technicalities of the approach this logic essentially allows the construction of complex types such as sets or lists out of other types, including standard types as e.g. natural numbers. The work presents a kernel defined on terms in the associated logic. However, the kernel is not capable of incorporating intensional background knowledge. In different spirit, [13] have proposed a kernel on Prolog proof trees. In this setting, the individuals are described in the context of global background knowledge in first-order logic. The idea of this kernel is then to measure the similarity of two individuals by means of the similarity of the proof trees of a special logic program, called the *visitor program*, which probes certain characteristics of the individuals that may be of interest for the domain. In contrast to other approaches, this kernel allows to exploit background knowledge in a principled way. However, as acknowledged by the authors, the design of appropriate visitor programs is not necessarily intuitive and the proof trees may easily contain unnecessary noise that may hurt the overall performance.

### 3 Designing Kernels for Instance Data

In this section, we look at transferring kernel design principles to Semantic Web data. We introduce a principled framework for defining kernel on instances that are described with respect to some ontology. We can view this framework as a toolbox for formulating adequate kernels on Semantic Web data. The framework we describe is generic and we believe that it is capable of capturing most of the interesting future scenarios. As explained earlier, kernels should capture the similarity of the arguments. Along the common lines of interpretation of similarity, we will consider two instances the more similar, the more common (or similar) characteristics they have.

Before digging into the definitions of the kernel components in the next two sections let us fix some notation. We will implicitly define the kernels with respect to some fixed ontology  $\mathcal{O}$ . We will denote the set of named individuals in  $\mathcal{O}$  as  $\mathcal{I}$ , the set of atomic classes  $C \subseteq \mathcal{I}$  as  $\mathcal{C}$ , the set of object properties  $p \subseteq \mathcal{I} \times \mathcal{I}$  as  $\mathcal{P}_{\mathcal{O}}$  and the set of datatype properties  $p \subseteq \mathcal{I} \times \text{dom}(d)$  as  $\mathcal{P}_{\mathcal{D}}$ .

We now introduce four layers of similarity of instances, each of which is defined using a particular (set- or cross product-) kernel. The model is roughly inspired by the layer model of [14] but the layers follow a different breakdown of the overall instance similarity as summarized in the following. The *identity layer* solely considers the identity of two instances. The *class layer* considers similarities of instances based on the classes the arguments can be shown to instantiate. The *property layers* consider similarities of instances based on the data properties and/or object properties the arguments can be shown to instantiate.

#### 3.1 Identity Layer Kernel

On this layer we employ a particularly simple kernel, the identity kernel, which basically performs a binary check on the identity of the two arguments.

**Definition 2 (Identity Kernel).** *Given two instances  $x, z$  and an ontology  $\mathcal{O}$ , we define the identity kernel as:  $k_{\text{identity}}(x, z) := \delta(x, z)$  with  $\delta(x, z) = 1$  if  $\mathcal{O} \models (x \equiv z)$  and  $\delta(x, z) = 0$  otherwise.*

We immediately note the correspondence to the matching kernel such that the kernel will typically evaluate to 0 unless the it is evaluated on an instance with itself or the ontology contains an explicit equivalence axiom for the two argument instances.

#### 3.2 Kernels on the Class Layer

We see the class(es) two individuals instantiate as a basic building block for their comparison. Intuitively, this type of similarity is useful only in those cases where there is some variation in the classes that are instantiated.

**Definition 3 (Common Class Kernel).** *Given two instances  $x, z$  and an ontology  $\mathcal{O}$ , we define the common class kernel as:  $k_{\text{class}}(x, z) := |\{C \in \mathcal{C} \mid \mathcal{O} \models C(x)\} \cap \{C \in \mathcal{C} \mid \mathcal{O} \models C(z)\}|$ .*



The class intersection kernel is a valid kernel, as it is a specific instantiation of the set intersection kernel. It can be interpreted as easily by defining the mapping  $\phi(\cdot)$  as a mapping into a vector space whose dimensions correspond to the atomic classes defined in the ontology. The kernel value is the higher the more atomic classes are instantiated by the argument instances at the same time, if there is no common class the kernel will be zero. We can extend this kernel by means of a weighting scheme  $\mu : C \rightarrow \mathbb{R}^+$  as follows:

**Definition 4 (Weighted Common Class Kernel).** *Given two instances  $x, z$  and an ontology  $O$ , we define the weighted common class kernel as:*

$$k_{class'}(x, z) := \sum_{c \in \{C \in C \mid O \models C(x)\} \cap \{C \in C \mid O \models C(z)\}} \mu(c).$$

### 3.3 Kernels on the Data Property Layer

As a next building block for determining the similarity of two individuals we consider properties. The general structure of this kernel is a sum of kernel evaluations of all compatible pairings of properties. We begin by defining kernels for data properties as follows.

**Definition 5 (Data Property Kernel).** *Given two instances  $x, z$ , a data property  $p \in \mathcal{P}_D$  and an ontology  $O$ , we define the data property kernel as:*

$$k_{DP}^p(x, z) := \sum_{\{d \mid O \models p(x, d)\}} \sum_{\{e \mid O \models p(z, e)\}} k_p(d, e)$$

whereby  $k_p$  is a valid kernel on the datatype referenced by the respective datatype property.

Again, the kernel property can be easily verified as the kernel is an instantiation of the crossproduct kernel. The kernel makes use of an underlying base kernel that is defined on the respective *datatype* to which the computes the similarity of pairs of data. For basic datatypes, such as strings or numeric values, a variety of useful kernels have been defined. For String datatypes, we may for example consider the well-known bag-of-words type kernels or the String Kernel introduced in [15]. For numerical values, the Gaussian Kernel defined on real numbers might be a useful choice. Again, we refer to [4] for information on such kernels.

### 3.4 Kernels on the Object Property Layer

In the same spirit as with the datatype property kernel, we define the object property kernels.

**Definition 6 (Object Property Kernel).** *Given two instances  $x, z$ , an object property  $p \in \mathcal{P}_O$  and an ontology  $O$ , we define the object property kernels as crossproduct kernels:*

$$k_{OP}^p(x, z) := \sum_{\{v \mid O \models p(x, v)\}} \sum_{\{w \mid O \models p(z, w)\}} k_p(v, w)$$



$$k_{O_p}^p(x, z) := \sum_{\{v \mid O \models p(v, x)\}} \sum_{\{w \mid O \models p(w, z)\}} k_p(v, w)$$

whereby  $k_p$  is a valid kernel on instance data.

This definition requires two further remarks. First, note that for each object properties we have to define two separate kernels, depending on whether the argument instances should be considered as the source or the target of the object property. Second, the definition in terms of the crossproduct kernel again makes reference to an underlying base kernel  $k_p$ , this time defined on *instances* just as the overall kernel itself. While being powerful and flexible, this approach can be tricky as we need to avoid cycles during the computation of the kernel. A cycle arises if a kernel defined at a higher level is used again as a kernel it directly or indirectly depend on. In essence we therefore require that the kernel reference graph conforms to a tree structure. As an alternative, the matching kernel can be used as base kernel, such that the object property kernel essentially boils down to an intersection kernel requiring no further dependencies.

### 3.5 Integrated Kernel Calculation

We have so far presented isolated building blocks for kernel calculations on individuals such as kernels comparing the class structure or the property extensions of the instances. All valid kernel combination operators, in particular products or any form of weighted addition would be possible to combine the results. In our view, an weighted additive combination appears to be the most intuitive approach. Similar to the weights  $\mu(C)$  for classes, the weight parameters can be used to tune the contribution of the corresponding kernels. Typically, only a small set of properties will be considered in the combined kernel with all other property kernels having weights equal to 0.

## 4 Illustrations and Evaluations

In this section we aim to show how the kernel framework introduced in this paper can be applied to real world Semantic Web data. We have implemented basic versions of the kernels introduced in the last section based on the ontology management and reasoning infrastructure *KAON2* [4]. Our implementation, the *KAON2Similarity API* [5] provides a number of instantiations of basic kernels together with supporting infrastructure such as kernel modifiers, kernel aggregators and a caching module that can be flexibly plugged together for the purpose at hand. We have further developed an JNI-based extension of the *SVMlight* Support Vector Machine software [16] that allows to deal with the kernels available via *Kaon2Similarity* [4].

As the task of learning from Semantic Web data has not yet been addressed actively, a comparative evaluation is not possible due to the lack of standardised evaluation data sets and baseline approaches. In fact, we aim at building a repository of standardized data sets for this purpose and see our experiments reported in this section as a first step

<sup>2</sup> <http://kaon2.semanticweb.org/>

<sup>3</sup> <http://kaon2similarity.ontoware.org/>

<sup>4</sup> <http://www.aifb.uni-karlsruhe.de/WBS/sbl/software/jnikernel>

**Table 1.** Galen Experiments: Statistics and Results

Number of overall positive Instances			Results on Galen Test Set				
Galen class	$O_{original}$	$O_{weak}$	err	prec	rec	$F_1$	
Biological_entity	550	63	0.03	1.00	0.96	0.98	
Physical_entity	587	11	0.02	0.99	0.96	0.97	
Complex	116	88	0.01	1.00	0.93	0.96	
Continuant_entity	600	96	0.02	1.00	0.97	0.98	
Discrete_entity	433	78	0.01	1.00	0.98	0.99	
Mass_entity	70	16	0.00	1.00	1.00	1.00	
Occurrent_entity	95	0	0.02	1.00	0.75	0.85	

into this direction. In this section, we thus introduce examples and results of two learning problems based on typical Semantic Web data sets. Both problems enjoy promising results and, even though they are comparatively simple, they illustrate the approach very well. In future work, we intend to widen such experiments to larger-scale settings.

#### 4.1 Common Class Kernel for the GALEN Ontology

We first illustrate the use of machine learning classification using only the simple common class kernel. For this purpose, we performed some experiments with the OWL DL version of the GALEN Upper Ontology<sup>5</sup>. The ontology contains atomic 175 classes, together with restrictions arranged in 193 SubClassOf axioms, 51 EquivalentClasses axioms, 127 DisjointClasses axioms and no Nominals. The idea of the experiment was to simply immitate the classification behaviour of the ontology given a semantically weekend ontology. We used 7 subclasses of the Self\_standing\_entity, namely the Biological\_entity, Physical\_entity, Complex, Continuant\_entity, Discrete\_entity, Mass\_entity and Occurrent\_entity classes<sup>6</sup>. As only the TBox of the ontology is available, we randomly populated the ontology with 1000 individuals. We then filtered the ontology axioms and retained only SubClassOf and ClassMember axioms to obtain a weaker and semantically different ontology  $O_{weak}$ . Table 1 summarizes the entailment statistics for the 7 classes for the original and the changed version of the ontology.

We then split the overall instance set in two groups of 500 instances for training and testing with class labels assigned according to the semantics of  $O_{original}$ . We trained (and tested) a Support Vector machine using the common class kernel (with a normalisation modifier) only using the information present in  $O_{weak}$ . The results of the test runs are also reported in table 1. As the results show, the common class kernel is easily able to immitate the reasoning behaviour of the complex ontology on a weaker – semantically different – ontology. These consistent positive results can probably be explained by the fact that the weaker ontology still contains clear and crisp pattern for certain Class (sub-) structures that can be detected easily by the learning algorithm. Nevertheless they show a potential for learning logical patterns in a statistical manner.

<sup>5</sup> <http://www.cs.man.ac.uk/~rector/ontologies/simple-top-bio/>

<sup>6</sup> The Non\_Biological\_entity, Organelle and Non\_Physical\_entity were left out as they are either complements of existing concepts or did not have enough positive examples.

## 4.2 Mixed Object Property Kernels for the SWRC Ontology

We built a second experiment around the SWRC ontology [17] and the metadata available within the Semantic Portal of the Institute AIFB. The SWRC ontology initially grew out of the activities in the KA2 project and is now applied in a number of projects also outside of AIFB. The ontology has been ported to various knowledge representation languages including OWL [8].

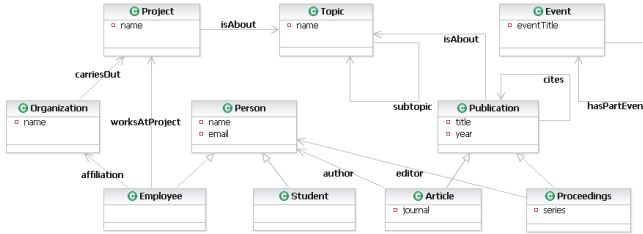


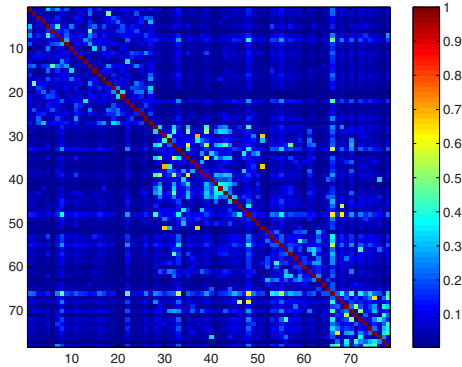
Fig. 1. Main classes and properties of the SWRC Ontology

The SWRC ontology generically models key entities relevant for typical research communities and the relations between them. The current version of the ontology comprises a total of 53 concepts in a taxonomy and 42 object properties, 20 of which are participating in 10 pairs of inverse object properties. All entities are enriched with additional annotation information. SWRC comprises a total of six top level concepts, namely the Person, Publication, Event, Organization, Topic and Project concepts. Figure 1 shows a small portion of the SWRC ontology with its main top-level concepts and relations. Since its initial versions, the SWRC ontology has been used and adapted in a number of different settings, most prominently for providing structured metadata for web portals. These include the web portal of the authors' institute AIFB [8]. The November 2006 version of the AIFB metadata comprises an additional set of 2,547 instances. 1,058 of these can be deduced to belong to the Person class 178 of which have an affiliation to one of the institute's groups (the others are external co-authors) with 78 of these being currently employed research staff. 1232 instances instantiate the Publication class, 146 instances instantiate the ResearchTopic class and 146 instances instantiate the Project class. The instances are connected by a total of 15,883 object property axioms and participate in a total of 8,705 datatype properties.

Given the information in the SWRC ontology, we have designed two classification problems that we have experimentally evaluated, namely the assignment of instances of the Person and Paper classes to one of the four research groups they are affiliated with (or, in the case of papers, where one of the authors is affiliated with.). Note that the information on the affiliations in the AIFB Portal is maintained by the Institute's administration and can thus be considered a very clean learning problem. On the other hand, the data about people's interests and projects or about paper metadata maybe noisy, or

<sup>7</sup> <http://ontoware.org/projects/swrc/>

<sup>8</sup> <http://www.aifb.uni-karlsruhe.de/about.html>



**Fig. 2.** Visualization of the (normalised) kernel matrix of 78 active AIFB researchers using the  $k_{ctp}$ . The researchers are ordered according to their research group affiliations.

**Table 2.** Leave-one-out classification results for the *persons2affiliation* problem on SWRC for different kernels and kernel modifiers. All numbers are percentages.

persons2affiliation, $c = 1$					persons2affiliation, $c = 10$				
kernel config	err	prec	rec	$F_1$	kernel config	err	prec	rec	$F_1$
sim-ctp-p	6.88	85.32	46.07	59.83	sim-ctp-p	7.72	78.10	43.10	55.55
sim-ctp-pc	5.48	93.09	53.64	68.06	sim-ctp-pc	6.46	86.87	49.77	63.29
sim-ctp-pgc	6.32	94.86	43.28	59.45	sim-ctp-pgc	6.18	86.25	51.61	64.58
sim-ctpp-p	6.04	90.70	48.78	63.44	sim-ctpp-p	7.87	72.59	51.65	60.36
sim-ctpp-pc	4.49	95.83	58.13	72.37	sim-ctpp-pc	5.20	90.90	54.22	67.92
sim-ctpp-pgc	5.90	96.43	45.14	61.49	sim-ctpp-pgc	5.34	86.62	57.16	68.87
sim-ctpp-star-pc	4.49	95.87	57.27	71.71	sim-ctpp-star-pc	4.92	89.83	57.11	69.83

inconsistent because this kind of data is maintained autonomously by the researchers. For each of the two tasks, we have employed a couple of kernels. Obviously, the used kernels did not rely directly on any of the target properties as the learning problem would be trivial otherwise.

For the *person2affiliation* task, we designed a set of three kernels. The *sim-ctp* kernels correspond to a kernel combining the common class similarity kernel (with a small weight, 0.1) and the `workedOnBy` and `worksAtProject` object properties, pointing to (or from) associated publications, research topics and projects, each with weight 1. The *sim-ctpp* kernels additionally use the `publication` object property, also with weight 1. In all these cases, the object property kernels are designed as matching kernels. As a variant to this, the *sim-ctpp-star* kernel used an embedded bow cosine kernel on the publications title filed for the `publication` object property. The modifiers *p*, *pc* and *pgc* indicate whether the respective sub-kernels were simply summed up, individually normalised and summed up or summed up and normalised afterwards, respectively. For the *papers2affiliation* task, we also designed a set of three kernels. As a kind of baseline, we report results on the *sim-t* kernel, which corresponds to the bow cosine kernel on the title datatype property. The *sim-cta* kernels combine the common class kernel (again with

**Table 3.** Leave-one-out classification results for the *papers2affiliation* problem on SWRC for different kernels and kernel modifiers. All numbers are percentages.

papers2affiliation, $c = 1$				
kernel config	err	prec	rec	$F_1$
sim-t-p	7.49	86.78	47.83	61.67
sim-cta-p	0.69	99.68	95.10	97.34
sim-cta-pc	0.69	99.79	94.52	97.09
sim-cta-pg1	6.84	96.39	56.97	71.62
sim-cta-pg3	1.38	99.59	90.15	94.63
sim-cta-pgc	0.85	99.44	94.28	96.79
sim-ctap-p	0.77	99.82	94.68	97.18
sim-ctap-pc	0.73	99.45	94.52	96.92
sim-ctap-pg1	7.63	95.93	53.14	68.40
sim-ctap-pg3	1.38	99.59	90.28	94.70
sim-ctap-pgc	0.91	99.33	94.39	96.80
sim-ctat-p	0.61	99.71	95.41	97.51
sim-ctat-pc	0.75	99.65	94.48	97.00

papers2affiliation, $c = 10$				
kernel config	err	prec	rec	$F_1$
sim-t-p	6.21	91.11	63.08	74.55
sim-cta-p	0.63	99.74	95.22	97.43
sim-cta-pc	0.63	99.17	94.95	97.02
sim-cta-pg1	6.09	96.77	61.23	75.01
sim-cta-pg3	0.69	99.79	94.91	97.29
sim-cta-pgc	0.58	99.03	95.68	97.33
sim-ctap-p	0.69	99.71	95.03	97.31
sim-ctap-pc	0.71	98.81	94.92	96.83
sim-ctap-pg1	6.63	96.34	60.11	74.03
sim-ctap-pg3	0.77	99.56	94.58	97.01
sim-ctap-pgc	0.67	98.80	95.48	97.11
sim-ctat-p	0.61	99.74	95.38	97.51
sim-ctat-pc	0.77	99.68	94.46	97.00

weight 0.1), with the *isAbout* and *author* object properties (each with weight 1) pointing to associated topics and authors. The *sim-ctap* kernels additionally consider the *hasProject* property whereas the *sim-ctat* kernels additionally consider the *title* datatype property, again in conjunction with the bow cosine kernel. Again, we have employed different kernel modifiers, this time additionally the *pg1* and *pg3* kernel modifiers which correspond to the gaussian modifiers applied to the plain sum with parameters  $\sigma$  equal to 1 or 3.

As an example, Figure 2 visualizes the kernel matrix (the matrix of all kernel evaluations on all pairs) for the case of the *sim-ctpp-p* kernel in the *person2affiliation* task. Table 2 and table 3 illustrate the macro-averaged results of the classification experiments, estimated via the Leave-One-Out cross-validation strategy, for two different choices of the SVM soft margin parameter  $c$ . Not surprisingly, the *papers2affiliations* task has achieved virtually optimal results that are stable over the different kernel variants. These good results can be traced to the fact that the object properties pointing to associated authors have been included in the kernel computation, inherently bearing a strong correspondence to the research groups.

## 5 Conclusion

In this paper, we have investigated the question how kernel methods can be used to directly apply machine learning algorithms on Semantic Web-type instance data. This problem has to the best of our knowledge only been considered to a minor extend. We believe that this type of learning will become increasingly important in future research both from the machine learning as well as from the Semantic Web communities.

We have introduced a generic and principled framework for designing valid kernels on this type of input data that also allows to implicitly exploit the schema-level knowledge encoded in the ontology. This approach is in line with other work on kernels for

structured data [10][12]. We have presented some initial experiments in which we have exemplified the instantiation of the proposed framework for the specific purpose by classifying researchers to their respective research groups based on information represented by the well-known SWRC ontology and associated instance data and in a scenario using the GALEN upper ontology. While simple and artificially posed, the experiments enjoyed promising results and illustrated the overall approach.

Judged by the aim, namely mining data described by means of logical predicates, this research brings in some connections to Inductive Logic Programming (ILP) and Multi-Relational Data Mining [9]. While the results will always be conceptually similar to results achieved by adequate propositionalization approaches [18], the use of kernels avoid extensive pre-processing efforts and the definition of adequate kernels or the choice of adequate kernel components as a notion of similarity may often be more intuitive to the involved users.

A challenge for future work will, however, be to design means to aid the user in the choice of the various kernels, kernel modifiers and parameters. One such research trail is the automated optimization of kernel parameters such as the weights for kernel components for example by means of kernel alignment techniques [19]. As another line for future work, we aim at augmenting our framework by other kernel types that can exploit the relation structure of an instance space such as diffusion kernels [20]. Such kernels can also help to finding means for a more intuitive approach to avoiding cyclic kernel definitions. As much of the research in Description Logics is occupied with the properties of complex class descriptions, yet a different research trail would be the consideration of DL-type class descriptions as input for kernel-based learning algorithms. Last but not least, these initial conceptual results need to be carried over to the emerging Semantic Web application domains.

## References

1. Brickley, D., Guha, R.: RDF vocabulary description language 1.0: RDF schema. W3C Recommendation, 10 February 2004, Published online (2004), at <http://www.w3.org/TR/2004/REC-rdf-schema-20040210/>
2. McGuinness, D.L., van Harmelen, F.: OWL web ontology language overview. W3C Recommendation, 10 February 2004, Published online (2004), at <http://www.w3.org/TR/2004/REC-owl-features-20040210/>
3. Buitelaar, P., Cimiano, P., Magnini, B.: Trends in Information Processing Systems. *Frontiers in Artificial Intelligence*, vol. 123, p. 180. IOS Press, Amsterdam (2005)
4. Shawe-Taylor, J., Cristianini, N.: *Kernel Methods for Pattern Analysis* (Hardcover). Cambridge University Press, Cambridge (2004)
5. Staab, S., Studer, R.: *Handbook on Ontologies*. Springer, Aix-en-Provence, France (2003)
6. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, Cambridge (2003)
7. Vapnik, V., Golowich, S.E., Smola, A.J.: Support vector method for function approximation, regression estimation and signal processing. In: NIPS, pp. 281–287 (1996)
8. Schölkopf, B., Smola, A., Müller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Technical Report 44, Max Planck Institute for Biological Cybernetics, Tübingen, Germany (1996)

9. Muggleton, S.H., Raedt, L.D.: Inductive logic programming: Theory and methods. *Journal of Logic Programming* 19,20, 629–679 (1994)
10. Gärtner, T.: A survey of kernels for structured data. *SIGKDD Explorations* 5(1), 49–58 (2003)
11. Haussler, D.: Convolution kernels on discrete structures. Technical Report Technical Report UCS-CRL-99-10, UC Santa Cruz (1999)
12. Gärtner, T., Lloyd, J.W., Flach, P.A.: Kernels and distances for structured data. *Machine Learning* 57(3), 205–232 (2004)
13. Raedt, L.D., Passerini, A.: Kernels on prolog proof trees: Statistical learning in the ILP setting. *Journal of Machine Learning Research* 7, 307–342 (2006)
14. Ehrig, M., Haase, P., Stojanovic, N., Hefke, M.: Similarity for ontologies - a comprehensive framework. In: *ECIS 2005. Proceedings of the 13th European Conference on Information Systems*, Regensburg, Germany, May 26-28, 2005 (2005)
15. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text classification using string kernels. *Journal of Machine Learning Research* 2, 419–444 (2002)
16. Joachims, T.: Making large-scale SVM learning practical. In: *Advances in Kernel Methods - Support Vector Learning*, MIT Press, Cambridge (1999)
17. Sure, Y., Bloehdorn, S., Haase, P., Hartmann, J., Oberle, D.: The SWRC ontology - semantic web for research communities. In: *Jajodia, S., Mazumdar, C. (eds.) ICISS 2005. LNCS*, vol. 3803, pp. 218–231. Springer, Heidelberg (2005)
18. Krogel, M.-A., Rawles, S., Zelezny, F., Flach, P., Lavrac, N., Wrobel, S.: Comparative evaluation of approaches to propositionalization. In: *Horváth, T., Yamamoto, A. (eds.) ILP 2003. LNCS (LNAD)*, vol. 2835, pp. 194–217. Springer, Heidelberg (2003)
19. Cristianini, N., Shawe-Taylor, J.: On kernel-target alignment. In: *Advances in Neural Information Processing Systems 14 - Proceedings of NIPS 2001*, Vancouver, Canada, December 3-8, 2001, pp. 367–373 (2001)
20. Kondor, R.I., Lafferty, J.D.: Diffusion kernels on graphs and other discrete input spaces. In: *ICML 2002. Proceedings of the Nineteenth International Conference on Machine Learning*, pp. 315–322. Morgan Kaufmann, San Francisco (2002)

# An Ontology Design Pattern for Representing Relevance in OWL

Fernando Bobillo, Miguel Delgado, and Juan Gómez-Romero\*

Department of Computer Science and Artificial Intelligence  
E.T.S. Ingeniería Informática y Telecomunicaciones, University of Granada  
c/. Periodista Daniel Saucedo Aranda, s/n 18071 Granada Spain  
Tel.: +34 958243194, Fax: +34 958243317  
fbobillo@decsai.ugr.es, mdelgado@ugr.es, jgomez@decsai.ugr.es

**Abstract.** Design patterns are widely-used software engineering abstractions which define guidelines for modeling common application scenarios. Ontology design patterns are the extension of software patterns for knowledge acquisition in the Semantic Web. In this work we present a design pattern for representing relevance depending on context in OWL ontologies, i.e. to assert which knowledge from the domain ought to be considered in a given scenario. Besides the formal semantics and the features of the pattern, we describe a reasoning procedure to extract relevant knowledge in the resulting ontology and a plug-in for Protégé which assists pattern use.

## 1 Introduction

Semantic Web is nowadays more than a promise and ontology-based applications are blooming here and there, permeating the Web environment with RDF and OWL aromas (mixed with microformats, folksonomies and other contributions from the Web 2.0 brand). Nevertheless, despite the efforts that research community has put on providing better tools to manage formal metadata, a classical issue in Artificial Intelligence is still paining the neck of developers: knowledge acquisition bottleneck. Acquiring, reusing, representing and eliciting knowledge to build an ontology becomes frequently an exhausting, time-wasting and frustrating experience, even when collaborative experts, proper tools and sound methodologies are in play.

Consequently simple recipes which support ontologists to apprehend aspects of their application domain are highly appreciated. This is the objective of ontology design patterns: to describe, more or less formally, recurrent modeling scenarios and to provide guidelines for incorporating this knowledge into ontologies correctly. By *correctly* we mean obtaining as a result accurate, transparent and reasonable representations, as pointed out in [1].

One of these common situations is the need of representing relevance of information. It is usual that an intelligent system manages so many information

---

\* Corresponding author.



resources that it is impractical to provide a user with all the available data in response to a query, since it will take too long to filter them manually or simply he will not be able to process it. This issue has been pointed out in the literature with the name of “information overload” [2]. In such case, only relevant information should be delivered and, in order to do so, relevance relations must be represented in the system knowledge base.

As a result of our work in Knowledge Mobilization –the effort to make knowledge available for real-time use, no matter where decision-making processes are taking place–, we strongly believe that what is relevant (or significant) for a user mostly depends on his circumstance<sup>1</sup>. The *circumstance* can be regarded as a mix of environment facts, beliefs, intentions and needs, i.e. (in a wide sense) his context. Accordingly, relevance should be represented in a knowledge base by defining relations among descriptions of usage cases and subsets of the domain knowledge. Having a domain and a scenario connected with such a relation means that this information is important and must be considered in that situation.

In this work we present a design pattern for representing and managing relevance relations in an OWL ontology. Besides the formal semantics of the model, we provide an algorithm to extract context-dependant summaries by reasoning within the ontology. We also introduce a graphical tool which eases the application of the pattern in the ontology development process.

The remaining of this document is structured as follows. Section 2 reviews some related work about ontology design patterns and context representation. Section 3 describes our design pattern; use cases, notation, and formulation are detailed. This section is clarified with the example in Section 4. Section 5 overviews our supporting software tool for applying the pattern. Section 6 includes a discussion of our proposal, as well as an analysis of its computational complexity. Finally, Section 7 points out some conclusions and directions for future work.

## 2 Related Work

Design patterns are concise guidelines which identify common design problems and suggest how to resolve them. Patterns have been recognized as a valuable tool since the very beginning of design sciences, from architecture to software development. Analysis and design patterns are important meta-artefacts which support the design process of software systems, as stressed by [3].

Templates and patterns to build knowledge bases have been proposed in several papers, some specific for a concrete application domain (e.g. [4]), some more general and (even) language-independent (e.g. [5]). Ontologies for the Semantic Web have their own peculiarities, so a task force inside the W3C Semantic Web Best Practices and Deployment Working Group<sup>2</sup> was settled to elaborate best

<sup>1</sup> Although he probably was not considering mobile knowledge-based systems, Spanish philosopher Ortega y Gasset (1883-1955) summarized this idea in his maxim “I am myself and my circumstance” (*Meditaciones del Quijote*, 1914).

<sup>2</sup> <http://www.w3.org/2001/sw/BestPractices/>

practices and patterns for OWL, namely the Ontology Engineering and Patterns Task Force<sup>3</sup>. The work of this task force was partially inspired by [6], a classical ontology-development guide which includes some tips to build them properly.

Ontology development patterns can be considered the extension of software engineering ones. In [7] some differences between both are described from a Semantic Web perspective, remarking that more formality is required in the presentation of the former, which are called CODEPs (Conceptual Ontology Design Patterns). In [1] different design-support artifacts for Semantic Web ontologies are overviewed and some examples of patterns are briefly presented.

Regarding representation of relevance of information depending on context, the aim of this paper, to the best of our knowledge there does not exist any design pattern specifically fitted to OWL particularities. Our proposal is OWL-DL compliant, unlike other approaches about contextualization of knowledge models which concerns non-monotonic formalisms, i.e. models which are satisfiable or not depending on some circumstances. It is interesting however to remark the contribution in [8], which examines some classical works about contexts and microtheories in Artificial Intelligence, and extends some of these ideas to solve context-dependant aggregation problems in the Semantic Web. Similarly, [9] proposes C-OWL, an extension to OWL to define mappings between locally-interpreted and globally-valid ontologies. To end up, we shall mention that the idea underlying our model is quite similar to the multi-viewpoint reasoning in [10], though it concentrates on the conditional interpretation of a model (how to reduce an ontology depending on the viewpoint submodel), whereas we focus on their relevance (in which circumstances a submodel should be considered).

On the other hand, some lessons can be learned from recent works in Pervasive Computing, as they are concerned with context awareness, content filtering and significance representation [11]. Moreover, as the example in Section 3.1 shows, we are especially interested in Ubiquitous Computing and Knowledge Mobilization, so the connection is even clearer. Not surprisingly, ontologies have been proposed to be used for modeling context knowledge in some recent developments in Pervasive Computing, e.g. [12] [13] [14].

### 3 Definition of the Pattern

This section describes the formal semantics of our proposal, the so called Context-Domain Relevance (CDR) pattern. We follow the recommendations sketched by [5] and [7], covering aspects as use cases, notation (in Description Logics language) and syntax (formulation).

#### 3.1 Use Case

Let us suppose a physician who needs to consult a patient's clinical data in order to set a proper treatment for him. If the healthcare act is taking place

<sup>3</sup> <http://www.w3.org/2001/sw/BestPractices/OEP/>

inside the hospital, the doctor will be allowed to access the Hospital Information System (HIS) and to retrieve all the patient's Electronic Health Records (EHRs). Having enough time and knowledge, the specialist will rule out all the useless pieces of information and will get the ones he is interested in. We can consider now another physician in an emergency-assistance unit which is caring at the road for a patient injured in an accident. Knowing some data about his clinical history will be helpful as well in this situation; for instance, some data about patient's adverse drug events (ADEs) may have been recorded. Nevertheless it is not probable that the HIS can be accessed from outside the hospital (even less using a portable device as the one which will be likely used in emergency healthcare) and, if possible, the doctor would not have enough time to review all the stored electronic records.

In the latter situation, a brief report including those pieces of the patient's clinical data which ought to be considered would be very valuable. The clinical procedure which is going to be carried out would determine which data should be part of this summary. For example, if the patient is slightly unconscious and has an hemorrhagic laceration, information about if he has been diagnosed of bad reactions to procaine (an anesthetic drug which reduces bleeding but is also often badly metabolized and triggers allergic reactions) should be taken into account, among others. This would be a prototypical sample of a Knowledge Mobilization application, in contrast to the former example which depicts a typical use case of a classical Information System.

Two different kinds of knowledge are to be managed by such mobile system: (i) domain knowledge about the problem which must be resolved (this is made up by the patients' electronic health records), and (ii) context knowledge about the scenarios where the domain knowledge will be used (for our doctor, this would be a vocabulary to briefly describe the situation of the patient he is going to attend). To state which knowledge from the domain must be considered in each scenario, links between both submodels can be defined. Continuing our example, a link asserting that 'data about previous anesthetic drugs reactions' should be considered when 'the patient has a penetrating wound' should be created. Other links can be similarly included following recommendations of clinical and ADE guidelines. Building these links is the aim of the CDR pattern.

### 3.2 Notation

As it is known, ontologies rely on Description Logics (DL), a family of logics for representing structured knowledge. In this section we overview the basics and the notation of DL. This notation will be used to describe the pattern and is directly translatable to OWL syntax.

In the remaining of this section we will consider the minimal subset proposed in the logic *ALC* (attributive concept description language with complements), since it is expressive enough to encode our pattern. *ALC* is less expressive than *SHOIN(D)* (almost equivalent OWL-DL, the highest descriptive level of OWL which ensures decidability) and therefore its complexity is lower. The reader is referred to [15] for further readings about DL.

Formally, an ontology is a triple  $O = \langle K_R, K_T, K_A \rangle$ , where  $K_R$  (the Role Box or RBox) and  $K_T$  (the Terminological Box or TBox) comprise the intensional knowledge, i.e. general knowledge about the world to be described (statements about roles and concepts, respectively), and  $K_A$  (the Assertional Box or ABox) the extensional knowledge, i.e. particular knowledge about a specific instantiation of this world (statements about individuals in terms of concepts and roles).

In  $\mathcal{ALC}$  there is no RBox, since no axioms involving roles are allowed. In more expressive logics,  $K_R$  consists of a finite set of role axioms stating restrictions as subsumption, transitivity, cardinality, etc.

An  $\mathcal{ALC}$  TBox  $K_T$  consists of a finite set of general concept inclusion (GCI) axioms of the form  $C_1 \sqsubseteq C_2$ , which means that concept  $C_1$  is more specific than  $C_2$ , i.e.  $C_2$  subsumes  $C_1$ . A concept definition  $C_1 \equiv C_2$  ( $C_1$  and  $C_2$  are equivalent) is an abbreviation of the pair of axioms  $C_1 \sqsubseteq C_2$  and  $C_2 \sqsubseteq C_1$ . Concept expressions for  $C_1, C_2$  can be derived inductively starting from atomic primitives. Valid constructs for  $\mathcal{ALC}$  are:  $C_1, C_2 \rightarrow A$  (atomic concept) |  $\top$  (top concept) |  $\perp$  (bottom concept) |  $C_1 \sqcap C_2$  (concept conjunction) |  $C_1 \sqcup C_2$  (concept disjunction) |  $\neg C_1$  (concept negation) |  $\forall R.C_1$  (universal quantification) |  $\exists R.C_1$  (full existential quantification).

An  $\mathcal{ALC}$  ABox consists of a finite set of assertions about individuals (noted  $a$  and  $b$ ). An assertion is either a concept assertion  $a : C$  ( $a$  is an instance of  $C$ ) or a role assertion  $(a, b) : R$  ( $(a, b)$  is an instance of  $R$ ).

A DL ontology not only stores axioms and assertions, but also offers some reasoning services, such as KB satisfiability (or consistency), concept satisfiability, subsumption or instance checking. In  $\mathcal{ALC}$  most inference services are mutually reducible, so only some of them are usually considered.

### 3.3 Formulation

In Sect. [B.1](#) we mentioned two knowledge sub-models that are involved in our design pattern. These correspond to the domain ontology and the context ontology and are the basis over which the relevance ontology is built.

The domain ontology  $O^D = \langle K_R^D, K_T^D, K_A^D \rangle$  contains the knowledge required to solve the concrete problem that the system is facing. As expected, concepts of this ontology represent entities with associated semantics, roles establish connections among them, and instances represent individuals of this world. This ontology can be arbitrarily complex and is closely related to the problem. We will use the notation  $D_j \overset{\circ}{\in} O^D$  to name complex concepts expressions  $D_j$  built using elements in  $O^D$  and ontology constructs. Note that, in principle, these  $D_j$  are not part of the domain ontology.

The context ontology  $O^C = \langle K_R^C, K_T^C, K_A^C \rangle$  contains the knowledge required to express the circumstances or the *surroundings* under which the domain knowledge will be used. The context ontology can be seen as a (formal) vocabulary or *lingo* with which these situations can be described. Being strict, context knowledge is not part of the original problem, though it can be indispensable to solve it; in fact, it would be possible to reuse the same context model in completely different areas. Context knowledge can range from low-level sensor data (like

location, time or humidity) to abstract information (like preferences, desires or mental state). We will use the notation  $C_i \overset{\circ}{\in} O^C$  to name complex concepts expressions  $C_i$  built using elements in  $O^C$  and ontology constructs. Like in the previous case, these  $C_i$  are not necessarily part of the context ontology.

Intuitively, we can guess that a CDR ontology will be made of new classes (the so called profiles) which will relate  $C_i$  context concepts with  $D_j$  domain concepts through quantified roles. We must note that, accordingly, our proposal only considers the intensional component of the knowledge base.

Regarding this we define constructively a CDR ontology as follows:

**Definition 1.** Let  $O^D$  and  $O^C$  be, respectively, the domain ontology and the context ontology,  $C_i \overset{\circ}{\in} O^C$  a context concept built with  $K_T^C$  classes, and  $D_j \overset{\circ}{\in} O^D$  a domain concept built with  $K_T^D$  classes.

The CDR ontology which relates the set of pairs of concepts  $\{(C_i, D_j)\}$  (i.e. states that  $D_j$  is interesting when  $C_i$  happens) is an ontology  $O^P = \langle K_T^P, K_A^P \rangle$  which satisfies:

1.  $K_A^P = \emptyset$
2.  $K_T^P$  include definitions for the concepts  $P_{\top}, C_{\top}, D_{\top}, P_{i,j}, C_i, D_j$ , where:
  - (a)  $P_{\top}, C_{\top}, D_{\top}$  are the super-classes Profile, Context and Domain:
 
$$P_{i,j} \sqsubseteq P_{\top} \wedge P_{\top} \equiv \bigcup_{i,j} P_{i,j}$$

$$C_i \sqsubseteq C_{\top} \wedge C_{\top} \equiv \bigcup_i C_i$$

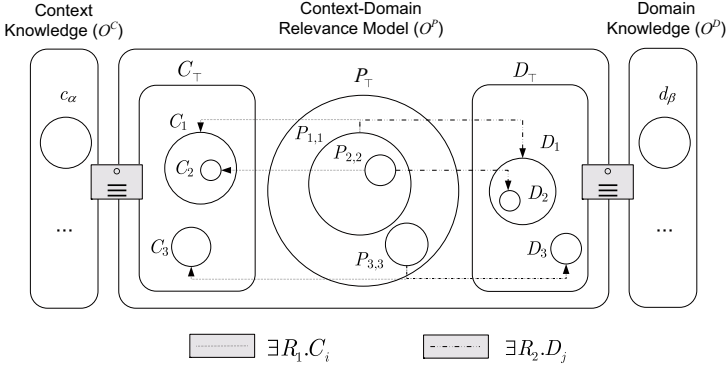
$$D_j \sqsubseteq D_{\top} \wedge D_{\top} \equiv \bigcup_j D_j$$
  - (b)  $R_1$  is the bridge property linking profiles and context concepts:
 
$$P_{\top} \sqsubseteq \forall R_1.C_{\top}$$
  - (c)  $R_2$  is the bridge property linking profiles and domain concepts:
 
$$P_{\top} \sqsubseteq \forall R_2.D_{\top}$$
  - (d)  $P_{i,j}$  is the profile linking named context  $C_i$  and named domain  $D_j$  :
 
$$P_{i,j} \equiv \exists R_1.C_i \sqcap \exists R_2.D_j$$
3.  $O^P$  is consistent.

Figure [1](#) depicts the meaning of this definition. It shows how  $P_{i,j}$  concepts are a reification of the “relevance” relation between context and domain concepts. Representing relevance as a concept and not as a role presents some advantages, e.g. possibility of reusing previously-defined profiles or defining new properties (with associated semantics) for them.

The main reasoning task within the CDR ontology will be to find the domain restricted by a context, that is, to find all the classes of the domain ontology which are associated using profiles (i.e. are relevant) with a given concept built with the context vocabulary. This can be expressed as follows:

**Definition 2.** Given  $O^C$ ,  $O^D$  and  $O^P$ , the restricted domain of the scenario  $S \overset{\circ}{\in} O^C$  (being  $S$  a complex concept expressed in  $K_T^C$  vocabulary) considering  $O^P$  comprises all the classes  $I$  such as:

$$\{I \in K_T^D \mid (S \sqsubseteq C_n) \wedge (P_{n,m} \in K_T^P) \wedge (I \sqsubseteq D_m)\}$$



**Fig. 1.** Schema of the Context-Domain Relevance ontology

**Algorithm 1.** *The restricted domain of a scenario  $S$  considering  $O^P$  can be computed in practice as follows:*

1. Retrieve all the named contexts  $C_n$  which subsume  $S$ :

$$\{C_n \sqsubseteq C_T \mid S \sqsubseteq C_n\}$$

2. Retrieve all the named profiles  $P_{k,l}$  which include  $C_n$  contexts (via  $R_1$ ):

$$\{P_{k,l} \sqsubseteq P_T \mid (P_{k,l} \sqsubseteq \exists R_1.C_k) \wedge (C_n \sqsubseteq C_k)\}$$

3. Retrieve all the named domains  $D_m$  which are related to  $P_{k,l}$  profiles (via  $R_2$ ):

$$\{D_m \sqsubseteq D_T \mid P_{k,l} \sqsubseteq \exists R_2.D_m\}$$

4. Retrieve all the classes  $I$  from  $K_T^D$  which are subsumed by  $D_m$ :

$$\{I \in K_C^D \mid I \sqsubseteq D_m\}$$

## 4 Example

Continuing with the medical case we have sketched in Sect. 3.1, let us suppose the following sample ontologies:

- A domain ontology  $O^D$  abstracting the information units managed by the hospital information system. Among others, it includes concepts as *Patient*, *ElectronicDocument* or *ElectronicRegisterCoagulationDisorder*, and properties as *relatedToPatient* (with domain equals to *Patient* and range equals to *ElectronicDocument*). Instances of this ontology are the concrete values of patient’s electronic health records.

- A context ontology  $O^C$  defining a suitable vocabulary to describe patient situations. It will contain concepts as *Hemorrhage*, *Unconsciousness*, *Trunk* or *High*, and properties like *hasSeriousness* (from *ClinicalFact* to *Seriousness*).

Using the definition of the CDR model, an ontology  $O^P$  can be built to reflect which information from the information system must be considered when facing each clinical case. With *hasClinicalFact* and *hasElectronicRegister* being the bridge properties  $R_1$  and  $R_2$ , respectively, the profiles in Table 1 will be valid. It can be observed that  $C_3 \sqsubseteq C_2 \sqsubseteq C_1$  and  $D_1 \sqsubseteq D_3$ ,  $D_2 \sqsubseteq D_3$ .

Given this profile set, if the doctor is attending to a “hemorrhagic and unconscious patient with a penetration wound”, the system will answer that electronic records about “drug intolerances” should be checked. This is achieved by using the previous algorithm to calculate the restricted domain of a context concept. The process is shown in Table 2.

The final information to be delivered to the doctor’s mobile device will be the instances of the *ElectronicRegister* classes in  $I$  filtered by patient ID, which

**Table 1.** Example of a Context Domain Restriction Ontology

Top concepts
$P_{\top} \sqsubseteq \top$
$C_{\top} \sqsubseteq \top$
$D_{\top} \sqsubseteq \top$
Profile 1,1. When the patient is “unconscious” and “hemorrhagic”, registers about “blood pressure disorders” must be checked
$C_1 \equiv \text{Unconsciousness} \sqcap \text{Hemorrhage}$
$D_1 \equiv \text{ElectronicRegisterBloodPressureDisorder}$
$P_{1,1} \equiv \exists \text{hasClinicalFact}.C_1 \sqcap \exists \text{hasElectronicRegister}.D_1$
Profile 2,2. When the patient is “unconscious”, “hemorrhagic” and has a “penetrating wound”, registers about “drug intolerances” must be checked
$C_2 \equiv \text{Unconsciousness} \sqcap \text{Hemorrhage} \sqcap \text{PenetrationWound}$
$D_2 \equiv \text{ElectronicRegisterDrugIntolerance}$
$P_{2,2} \equiv \exists \text{hasClinicalFact}.C_2 \sqcap \exists \text{hasElectronicRegister}.D_2$
Profile 3,3. When the patient is “unconscious”, with a “highly serious” “hemorrhage” and has a “penetrating wound”, registers about “blood pressure disorders”, “drug intolerances” and “coagulation disorders” must be checked
$C_3 \equiv \sqcap(\text{Unconsciousness} \sqcap \text{Hemorrhage} \sqcap \text{PenetrationWound} \sqcap \exists \text{hasSeriousness}.High)$
$D_3 \equiv \sqcup(\text{ElectronicRegisterBloodPressureDisorder} \sqcup \text{ElectronicRegisterDrugIntolerance} \sqcup \text{ElectronicRegisterCoagulationDisorder})$
$P_{3,3} \equiv \exists \text{hasClinicalFact}.C_3 \sqcap \exists \text{hasElectronicRegister}.D_3$

**Table 2.** Resolution of the example

(1)	$S \equiv Hemorrhage \sqcap Unconsciousness \sqcap PenetrationWound$ $C_n = \{C_1, C_2\}$
(2)	$P_{k,l} = \{P_{1,1}, P_{2,2}\}$
(3)	$D_m = \{D_1, D_2\}$
Let us suppose that <i>ElectronicRegisterBloodPressureDisorder</i> is a leaf concept in $O^D$ and <i>ElectronicRegisterDrugIntolerance</i> has two subclasses: <i>ElectronicRegisterProcaineIntolerance</i> and <i>ElectronicRegisterPenicillinIntolerance</i> . Then:	
(4)	$I = \begin{array}{l} ElectronicRegisterBloodPressureDisorder \\ ElectronicRegisterDrugIntolerance \\ ElectronicRegisterProcaineIntolerance \\ ElectronicRegisterPenicillinIntolerance \end{array}$

must mirror the data stored in the hospital information system. It would be possible to store these instances in a RDF specific repository, avoiding the overload of having them embedded in the ontology.

Note that in Algorithm 1 descendants of  $S$  are not inferred during the reasoning process, since these concepts corresponds to more specific context situations –which will probably drive to more specialized domain information–. However, it may be interesting to calculate the profiles involving these subcontexts and to provide them as feedback information to the user, in order to recommend him to describe further details of the current scenario. For instance, in this example,  $C_3$  in  $P_{3,3}$  is subsumed by  $S$ :

$$Unconsciousness \sqcap (Hemorrhage \sqcap \exists hasSeriousness.High) \sqcap PenetrationWound \sqsubseteq Unconsciousness \sqcap Hemorrhage \sqcap PenetrationWound$$

Consequently the doctor could be advised to carry out other clinical trials to see if the specific part of this restriction ( $\exists hasSeriousness.High$  qualifier of *Hemorrhage*) is present but has not been diagnosed yet. If this knowledge is supplied afterwards, more information about the patient (information unit *ElectronicRegisterCoagulationDisorder*) will be provided.

## 5 CDR Plug-in for Protégé

We have developed a plug-in for the Protégé platform (the ontology development tool from the University of Stanford<sup>4</sup>) which allows to create, edit, test and reason with a CDR ontology. Our plug-in adds a new tab to the Protégé-OWL environment (Protégé enhanced with the OWL plug-in<sup>5</sup>) where a simplified view

<sup>4</sup> <http://protege.stanford.edu>

<sup>5</sup> <http://protege.stanford.edu/plugins/owl/>



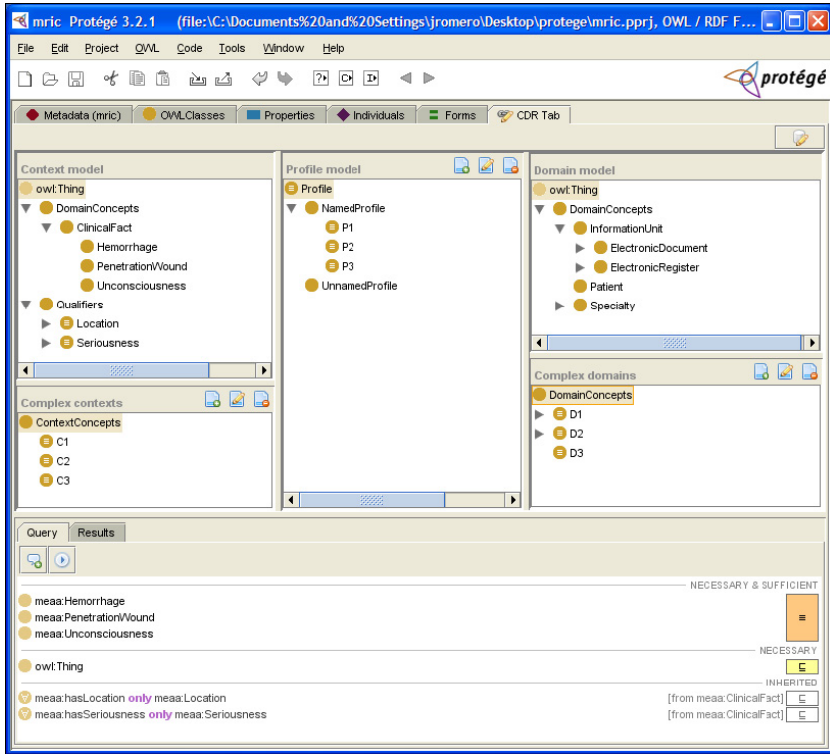


Fig. 2. CDR Tab plug-in in Protégé IDE

of the CDR ontology is displayed and queries can be introduced. A preliminary version can be downloaded in <http://arai.ugr.es/iaso/cdrplugin/>.

We can distinguish four sections in the tab, depicted in Figure 2:

1. *Context side.* The left side of the tab shows the context ontology ( $O^C$ ) and the complex contexts ( $C_i$ ) existing in the profile ontology. The context ontology can be optionally hidden. New complex contexts can be created using the context vocabulary; existential restrictions for the new  $C_i$  are automatically added. It is also possible to edit or delete existing profiles.
2. *Domain side.* The right side mirrors the context side but changing context knowledge by domain knowledge ( $O^D$ ). New complex domains  $D_i$  can also be easily created and editing and deleting are as well allowed.
3. *Profiles.* The central section of the tab shows the profiles in the ontology  $O^P$ . This is probably the most interesting part, since it simplifies the task of creating new profiles. To build a new profile, the user has just to select a complex context in the left box ( $C_i$ ) and a complex domain in the right box ( $D_j$ ), and then push the ‘new profile’ button. The new profile ( $P_{i,j}$ ) will be created as a subclass of the selected profile and the corresponding existential restrictions will be automatically generated.

4. *Reasoning.* The bottom section allows to retrieve the domain relevant to a given context, i.e. it implements Algorithm 1. When a new complex concept for querying is created, its restrictions are shown and the ‘run query’ button is activated. Results are displayed in the ‘Results’ tab of this reasoning section and further information about the obtained classes can be consulted.

From the formal description of the pattern in Section 3.3 it can be deduced that some additional configuration is needed to make the CDR plug-in work correctly. This involves stating the URIs of the external ontologies ( $O^C$ ,  $O^D$ ), the top concepts for the profiles, the contexts and the domains ( $P_\top$ ,  $C_\top$ ,  $D_\top$ ) and the URI of the DIG reasoner which will be used. To assist this procedure, a wizard-like window is presented to the user when pushing the ‘properties’ button on the top toolbar.

The plug-in has been developed with the APIs for Protégé and Protégé-OWL version 3.2.1. It also relies on the CDR-API, our library to manage programmatically models created with the pattern. Installation is easy; as any other Protégé add-on, it just has to be copied to the plug-in directory of the Protégé installation.

## 6 Discussion

Next we summarize some of the highlights of the CDR pattern. Studying computational properties of the resulting ontology deserves its own subsection, where complexity is detailed.

### 6.1 Features

- *Reusability.* By definition, design patterns must be applicable to different problems and domain areas. Our pattern effectively fulfills this objective, since it provides a general guideline for representing relevance without imposing application-dependant restrictions on the domain and the context ontologies.
- *Standardization.* One of the main a priori requirements for our pattern was OWL-DL compliance, that is, the resulting ontology should not include new constructors nor be in OWL-Full. As explained in Section 3.3, the pattern generates a new OWL-DL ontology whose complexity is bounded by context and domain models. Thus, though the reasoning process may seem little straightforward, current tools (e.g. inference engines) can be directly used, without having to extend, modify or re-implement them.
- *Formalization.* We have provided a formal specification of the pattern which goes further than usual text descriptions. This is possible because the target language, OWL, relies on a logic-based formalism, DL.
- *Modularization.* The pattern promotes ontology modularization, as it clearly separates the three involved models. Nevertheless, a limitation is imposed by OWL importing mechanism: the profile ontology  $O^P$  must import completely

$O^C$  and  $O^D$ , as partial including is not allowed. This forces the model to be globally interpreted and valid, which would not be desirable if different (and probably contradictory) relevance criteria and contexts are to be represented.

- *Expressivity*. The pattern allows to represent relevance taking the most of OWL expressivity. For instance, profile hierarchies can be defined to assert inclusion relations between them. In fact, the resulting model is an OWL ontology and can be modified as needed. Further improvements may be considered, e.g. definition of several bridge properties with different semantics to qualify the connections between contexts and domain, or adding properties to profile classes.

## 6.2 Complexity Analysis

Computational complexity of the inference within the CDR model is conditioned by complexity of context and domain expressions ( $C_i \in K_T^C$  and  $D_j \in K_T^D$ ), since  $P_{i,j}$  definitions are included in  $\mathcal{ALC}$  level. In the simplest case, that is  $O^C$ ,  $O^D$  and  $O^P$  ontologies are in  $\mathcal{ALC}$ , reasoning within the CDR ontology is asymptotically bounded by ontology classification complexity, which is EXPTIME for  $\mathcal{ALC}$  with GCIs according to Table 3 [16].

Supposing that  $O^C$  and  $O^D$  do not add further complexity, it is possible to reduce the complexity of the CDR model by restricting the allowed constructors for  $C_i$  and  $D_j$ , moving consequently to a less expressive logic. Restricting negation to atomic concepts and disallowing union concepts would enclose the CDR ontology to  $\mathcal{AL}\mathcal{E}$ , which has PSPACE complexity for general reasoning. Other alternative consists on using only acyclic TBoxes, which would give complexities of PSPACE for  $\mathcal{ALC}$  and CONP for  $\mathcal{AL}\mathcal{E}$ .

Other choices are not appropriate, however. Moving from  $\mathcal{ALC}$  to  $\mathcal{ALU}$  does not reduce the complexity, neither in the general case nor with acyclic TBoxes. Moving to  $\mathcal{AL}$  is not possible, because existential quantification can not be restricted. Similarly, expressivity of  $\mathcal{FL}^-$  is too limited.

According to this formulation, role hierarchies are not necessary in the CDR model. Nevertheless, they may be considered for convenience, in such a way that sub-roles of  $R_1$  and  $R_2$  can be defined with particular semantics and handled consequently. This will increase the complexity to  $\mathcal{ALCH}$ , but with the advantage that reasoning for the general case still remains EXPTIME.

**Table 3.** Complexity of reasoning in basic DLs

DL \ TBox	acyclic	general
$\mathcal{FL}^-$	P <sub>TIME</sub>	P <sub>TIME</sub>
$\mathcal{AL}$	CONP	PSPACE
$\mathcal{AL}\mathcal{E}$	CONP	PSPACE
$\mathcal{ALU}$	PSPACE	EXP <sub>TIME</sub>
$\mathcal{ALC}$	PSPACE	EXP <sub>TIME</sub>

In any case,  $\mathcal{ALCH}$  is less expressive than  $\mathcal{SHIF}(\mathcal{D})$  (equivalent to OWL-Lite), so reasoning in practice with available DL engines (e.g. Pellet<sup>6</sup>) will be quite efficient, as they are highly optimized and worst-case inferences are infrequent. Hence, more complex logics with extended semantics could be as well considered to extend the basic formulation without significant performance impact.

## 7 Conclusions and Future Work

In this work we have presented the CDR formalism, a design pattern for the representation and management of context-relevant knowledge in OWL ontologies. This pattern eases the representation of knowledge when facing the problem of information overload in KBSs, which is critical in Knowledge Mobilization. We also provide a plug-in for the Protégé ontology-development platform which simplifies constructing, editing and consulting the relevance model; currently this software is being tested and new features are being suggested to be implemented. Finally, we have discussed the main features of the pattern, remarking reusability and standardization as the more important, and studied computational complexity of the resulting ontology.

Looking into the future, we strongly believe that describing and promoting best practices for Semantic Web ontologies is not only useful but also necessary to boost semantic applications. More design recommendations and patterns as the produced by the OEP Task Force should be publicly available for use and discussion. It is interesting to note that in turn an eventual pattern repository could be described using an ontology.

Concerning our design pattern, in Sect. 6.1 we have remarked that OWL imports can be problematic when more than one context or profile model is involved. This issue has been pointed out in some current works and some solutions have been proposed [17]. Evolution of the relevance model is also important and temporal and non-monotonic reasoning formalism may be further considered. In fact, representing validity (as in temporal and non-monotonic logics) depending on time or new knowledge (both can be assimilated to context) and relevance (as in our model) might be regarded as similar ideas. It would be interesting to compare both approaches and to study to which extent one can be reduced to the other.

A fuzzy and probabilistic/possibilistic extension to the crisp ontology generated by the CDR pattern is also being considered. Such fuzzy ontology would allow to define weighted relevance relations between context and domains and, which is more interesting, partial matching of similar contexts. For instance, a context could be asserted to be subsumed by another with a certain degree. This would make the CDR ontology no longer compliant with OWL-DL, so we are as well studying procedures to reduce a fuzzy ontology to a crisp one [18].

**Acknowledgement.** This research has been partially supported by the project TIN2006-15041-C04-01 (Ministerio de Educación y Ciencia). F. Bobillo holds

<sup>6</sup> <http://pellet.owldl.com/>

a FPU scholarship from Ministerio de Educación y Ciencia. J. Gómez-Romero holds a FD scholarship from Consejería de Innovación, Ciencia y Empresa (Junta de Andalucía).

## References

1. Svátek, V.: Design Patterns for Semantic Web Ontologies: Motivation and Discussion. In: Proceedings of BIS2004 (2004)
2. Eppler, M., Mengis, J.: The Concept of Information Overload: A Review of Literature from Organization Science, Accounting, Marketing, MIS, and Related Disciplines. *The Information Society* 20(5) (2004)
3. Iivari, J.: Information Systems as a Design Science. In: *Information Systems Development: Advances in Theory, Practice and Education*, Springer, Heidelberg (2005)
4. Reich, J.R.: Ontological Design Patterns: Metadata of Molecular Biological Ontologies, Information and Knowledge. In: Ibrahim, M., Küng, J., Revell, N. (eds.) *DEXA 2000*. LNCS, vol. 1873, Springer, Heidelberg (2000)
5. Staab, S., Erdmann, M., Maedche, A.: Engineering ontologies using semantic patterns. In: *Procs. IJCAI 2001 Workshop on E-Business & the Intelligent Web* (2001)
6. Noy, N., McGuinness, D.: *Ontology Development 101: A Guide to Creating Your First Ontology*. Technical report, Stanford Knowledge Systems Laboratory (2001)
7. Gangemi, A.: Ontology Design Patterns for Semantic Web Content. In: *Proceedings of the 4th International Semantic Web Conference* (2005)
8. Guha, R., McCool, R., Fikes, R.: Contexts for the Semantic Web. In: *Proceedings of the 3rd International Semantic Web Conference* (2004)
9. Bouquet, P., Giunchiglia, F., van Harmelen, F., Serafini, L., Stuckenschmidt, H.: Contextualizing ontologies. *Web Semantics: Science, Services and Agents on the World Wide Web* 1(4) (2004)
10. Stuckenschmidt, H.: Toward Multi-viewpoint Reasoning with OWL Ontologies. In: *The Semantic Web: Research and Applications* (2006)
11. Weiser, M.: The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.* 3(3) (1999)
12. Chen, H., Finin, T., Joshi, A.: The SOUPA Ontology for Pervasive Computing. In: *Ontologies for Agents: Theory and Experiences*, Birkhuser Basel (2005)
13. Gu, T., Pung, H., Zhang, D.: A service-oriented middleware for building context-aware services. *Journal of Network and Computer Applications* 28(1) (2005)
14. Khedr, M., Karmouch, A.: ACAI: agent-based context-aware infrastructure for spontaneous applications. *Journal of Network and Computer Applications* 28(1) (2005)
15. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel -Schneider, P.F.: *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, Cambridge (2003)
16. Calvanese, D.: Reasoning with Inclusion Axioms in Description Logics: Algorithms and Complexity. In: *European Conference in Artificial Intelligence* (1996)
17. Bao, J., Honavar, V.: Divide and Conquer Semantic Web with Modular Ontologies - A Brief Review of Modular Ontology Language Formalisms. In: *WoMo 2006. First International Workshop on Modular Ontologies*, Athens, USA (2006)
18. Bobillo, F., Delgado, M.: A crisp representation for fuzzy *SHOIN* with fuzzy nominals and general concept inclusions. In: *Proceedings of the 2nd Int. Workshop on Uncertainty Reasoning for the Semantic Web*, Athens, Georgia, USA (2006)

# Logical Foundations of (e)RDF(S): Complexity and Reasoning\*

Jos de Bruijn<sup>1</sup> and Stijn Heymans<sup>2</sup>

<sup>1</sup> Faculty of Computer Science, Free University of Bozen-Bolzano, Italy  
debruijn@inf.unibz.it

<sup>2</sup> Digital Enterprise Research Institute (DERI), University of Innsbruck, Austria  
stijn.heymans@deri.org

**Abstract.** An important open question in the semantic Web is the precise relationship between the RDF(S) semantics and the semantics of standard knowledge representation formalisms such as logic programming and description logics. In this paper we address this issue by considering embeddings of RDF and RDFS in logic. Using these embeddings, combined with existing results about various fragments of logic, we establish several novel complexity results. The embeddings we consider show how techniques from deductive databases and description logics can be used for reasoning with RDF(S). Finally, we consider querying RDF graphs and establish the data complexity of conjunctive querying for the various RDF entailment regimes.

## 1 Introduction

The Resource Description Framework RDF [16], together with its vocabulary description language RDFS [3], constitutes the basic language for the semantic Web. More expressive semantic Web languages such as the description logic-based OWL DL [19] and future semantic Web rule languages [1] are supposed to extend it. However, certain properties of the RDF semantics posed layering problems in the definition of OWL DL [12]; it was decided to extend only a part of RDF. This has led to a situation in which the user communities of RDF and OWL DL are increasingly diverging, leading to a fragmentation of the semantic Web. There is a possibility that a similar situation will occur if a possible future rules language for the semantic Web does not adequately account for RDF(S).

Research has been done to uncover some of the formal properties of RDF (e.g. [13, 10, 4]). However, so far little research has been done into the formal relationships between RDF and the logical language paradigms of description logics and logic programming [1]. Therefore, we deem it worthwhile to investigate these relationships, in order to facilitate the RDF-compatibility of a future logic-based rules language for the semantic Web, and to see whether the RDF and description logic worlds can be brought closer

---

\* This work was partially supported by the European Commission under the projects Knowledge Web (IST-2004-507482) and SUPER (FP6-026850).

<sup>1</sup> <http://www.w3.org/2005/rules/>

<sup>2</sup> A notable exception is [4].

together. Additional benefits of such an investigation include the possible use of techniques from logic programming and description logics for reasoning with RDF(S).

The RDF semantics specification [11] defines three increasingly expressive kinds of entailment, namely *simple*, *RDF* and *RDFS entailment*. Furthermore, it describes *extensional RDFS* (eRDFS) entailment as a possible extension of RDFS entailment which is more in line with description logic-like languages. We refer to these kinds of entailment as *entailment regimes*.

To investigate the relationship between RDF and logic we embed the various RDF entailment regimes in F-Logic [15], which is a syntactic extension of first-order logic with object oriented modeling constructs. It turns out that the attribute value construct in F-Logic is exactly equivalent to the triple construct in RDF, and the typing (class membership) construct in F-Logic is very close in spirit to the one in RDF. Additionally, F-Logic, like RDFS, allows to use the same identifier as a class, instance, or property identifier, while still having a standard first-order logic-based semantics.

These embeddings can be used for RDF reasoning using Datalog engines. Furthermore, they lead to several novel complexity results about RDF; see Table 3 on page 96 for an overview of the complexity results for the various entailment regimes.

We then show the embedding of a large subset of *extensional RDFS* in FOL, and we show that it can be embedded in the tractable description logic (contextual) *DL-Lite<sub>R</sub>* [7].

Finally, we define a notion of conjunctive queries over RDF graphs, and establish the data complexity of query answering for the respective entailment regimes.

The structure of the remainder of the paper is as follows. In Section 2 we review F-Logic, *DL-Lite<sub>R</sub>*, and RDF. In Section 3 we define embeddings of RDF in F-Logic and FOL, and demonstrate the relationship with *DL-Lite<sub>R</sub>*. In Section 4 we use these embeddings and correspondences to obtain several novel complexity results for RDF. In Section 5 we define conjunctive query answering in RDF and exhibit its complexity. We discuss some implications of the results in this paper, and compare it with related work, in Section 6. We conclude the paper and outline future work in Section 7.

This paper is an extended version of [5], adding support for literals and treatment of RDF querying. Unlike in [5] we do not consider extensions of the embeddings of logical rules or formulas in this paper; see Section 7 for a discussion about the combination of RDF graphs with logical rules.

Proofs of all results are available in an extended version of this paper.

## 2 Preliminaries

*F-Logic* [3] extends first-order logic with constructs for object-oriented modeling (we use the object typing and attribute value construct), while staying in a first-order semantic framework.

The signature of an F-language  $\mathcal{L}$  is of the form  $\Sigma = \langle \mathcal{F}, \mathcal{P} \rangle$  with  $\mathcal{F}$  and  $\mathcal{P}$  disjoint sets of function and predicate symbols, each with an associated arity  $n \geq 0$ . Let  $\mathcal{V}$  be a set of variable symbols disjoint from  $\mathcal{F}$  and  $\mathcal{P}$ . Terms and atomic formulas are defined

<sup>3</sup> Note that F-Logic is also often used as an extension of nonmonotonic logic programming; however, we follow the original definition which is strictly first-order.



in the usual way;  $\perp$  is an atomic formula. A molecule in F-Logic is one of the following: (i) an *is-a* molecule of the form  $C : D$ , which states that an individual  $C$  is of the type  $D$ , or (ii) a *data molecule* of the form  $C[D \rightarrow E]$  which states that an individual  $C$  has an attribute  $D$  with the value  $E$ , where  $C$ ,  $D$  and  $E$  terms. A molecule is *ground* if it does not contain variable symbols.

Formulas of an F-language  $\mathcal{L}$  are either atomic formulas, molecules, or compound formulas, which are constructed in the usual way from atomic formulas, molecules, the logical connectives  $\neg$ ,  $\wedge$ ,  $\vee$  and  $\supset$ , the quantifiers  $\exists$  and  $\forall$ , and the auxiliary symbols ‘ $\cdot$ ’ and ‘ $(\cdot)$ ’. An F-Logic theory  $\Phi \subseteq \mathcal{L}$  is a set of formulas.

F-Logic Horn formulas are of the form  $(\forall)B_1 \wedge \dots \wedge B_n \supset H$ , with  $B_1, \dots, B_n, H$  atomic formulas or molecules. F-Logic Datalog formulas are F-Logic Horn formulas with no function symbols of arity higher than 0 such that every variable in  $H$  occurs in some equality-free  $B_1, \dots, B_n$ . F-Logic Horn and Datalog theories are sets of F-Logic Horn and Datalog formulas, respectively.

An *F-structure* is a tuple  $\mathbf{I} = \langle U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\rightarrow} \rangle$ , where  $U$  is a non-empty, countable set (the domain) and  $\in_U$  is a binary relation over  $U$ . An  $n$ -ary function symbol  $f \in \mathcal{F}$  is interpreted as a function over the domain  $U$ :  $\mathbf{I}_F(f) : U^n \rightarrow U$ . An  $n$ -ary predicate symbol  $p \in \mathcal{P}$  is interpreted as a relation over the domain  $U$ :  $\mathbf{I}_P(p) \subseteq U^n$ .  $\mathbf{I}_{\rightarrow}$  associates a binary relation over  $U$  with each  $k \in U$ :  $\mathbf{I}_{\rightarrow}(k) \subseteq U \times U$ . Variable assignments are defined as usual.

Given an F-structure  $\mathbf{I}$  of an F-language  $\mathcal{L}$ , a variable assignment  $B$ , and a term  $t$  of  $\mathcal{L}$ ,  $t^{\mathbf{I},B}$  is defined as:  $x^{\mathbf{I},B} = x^B$  for  $x \in \mathcal{V}$  and  $t^{\mathbf{I},B} = \mathbf{I}_F(f)(t_1^{\mathbf{I},B}, \dots, t_n^{\mathbf{I},B})$  for  $t$  of the form  $f(t_1, \dots, t_n)$ , with  $f \in \mathcal{F}$  an  $n$ -ary function symbol and  $t_1, \dots, t_n$  terms.

Satisfaction of atomic formulas and molecules  $\phi$  in  $\mathbf{I}$ , given the variable assignment  $B$ , denoted  $(\mathbf{I}, B) \models_f \phi$ , is defined as:  $(\mathbf{I}, B) \not\models_f \perp$ ,  $(\mathbf{I}, B) \models_f p(t_1, \dots, t_n)$  iff  $(t_1^{\mathbf{I},B}, \dots, t_n^{\mathbf{I},B}) \in \mathbf{I}_P(p)$ ,  $(\mathbf{I}, B) \models_f t_1 : t_2$  iff  $t_1^{\mathbf{I},B} \in_U t_2^{\mathbf{I},B}$ ,  $(\mathbf{I}, B) \models_f t_1[t_2 \rightarrow t_3]$  iff  $(t_1^{\mathbf{I},B}, t_3^{\mathbf{I},B}) \in \mathbf{I}_{\rightarrow}(t_2^{\mathbf{I},B})$ , and  $(\mathbf{I}, B) \models_f t_1 = t_2$  iff  $t_1^{\mathbf{I},B} = t_2^{\mathbf{I},B}$ . This extends to arbitrary formulas in the usual way.

The notion of a model is defined in the usual way. A theory  $\Phi \subseteq \mathcal{L}$  *F-entails* a formula  $\phi \in \mathcal{L}$ , denoted  $\Phi \models_f \phi$ , if for all F-structures  $\mathbf{I}$  such that  $\mathbf{I} \models_f \Phi$ ,  $\mathbf{I} \models_f \phi$ .

*FOL and DL-Lite $\mathcal{R}$* . *Classical first-order logic* (FOL) is F-Logic without molecules. *Contextual first-order logic* (contextual FOL) is classical FOL where  $\mathcal{F}$  and  $\mathcal{P}$  are not required to be disjoint, function and predicate symbols do not have associated arities, and for every structure  $\mathbf{I} = \langle U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\rightarrow} \rangle$  it is the case that  $\mathbf{I}_F$  assigns a function  $\mathbf{I}_F(f) : U^i \rightarrow U$  to every  $f \in \mathcal{F}$  and  $\mathbf{I}_P$  assigns a relation  $\mathbf{I}_P(p) \subseteq U^i$  to every  $p \in \mathcal{P}$ , for every nonnegative integer  $i$ . We denote satisfaction and entailment in classical and contextual FOL using the symbols  $\models$  and  $\models_c$ , respectively.

F-Logic can be straightforwardly embedded in FOL, as shown in the following proposition.

**Proposition 1.** *Let  $\Phi$  and  $\phi$  be an F-Logic theory and formula which do not contain the binary and ternary predicate symbols *isa* and *data*, respectively, and let  $\Phi'$  and  $\phi'$  be the FOL theory and formula obtained from  $\Phi$  and  $\phi$  by replacing every *is-a* molecule  $a : b$  with the atomic formula *isa*( $a, b$ ) and every *data* molecule  $a[b \rightarrow c]$  with the atomic formula *data*( $a, b, c$ ). Then,*



- $\Phi$  has a model iff  $\Phi'$  has a model and
- $\Phi \models_f \phi$  iff  $\Phi' \models \phi'$ .

$DL-Lite_{\mathcal{R}}$  [7] is a description logic (DL) with certain desirable computational properties; most reasoning tasks are polynomial, and query answering has **LogSpace** data complexity.

A classical (resp., contextual)  $DL-Lite_{\mathcal{R}}$  language consists of pairwise disjoint (resp., possibly non-disjoint) sets of concept ( $\mathcal{C}$ ), role ( $\mathcal{R}$ ), and individual ( $\mathcal{F}$ ) identifiers. Concepts and roles in  $DL-Lite_{\mathcal{R}}$  are defined as follows:

$$\begin{aligned} C_l &\longrightarrow A \mid \exists R \\ C_r &\longrightarrow A \mid \exists R \mid \neg A \mid \neg \exists R \\ R, R' &\longrightarrow P \mid P^- \end{aligned}$$

with  $A \in \mathcal{C}$ ,  $P \in \mathcal{R}$ ,  $C_l$  (resp.,  $C_r$ ) a left- (resp., right-)hand side concept, and  $R, R' \in \mathcal{R}$  roles.

A  $DL-Lite_{\mathcal{R}}$  knowledge base  $\mathcal{K}$  consists of a TBox  $\mathcal{T}$  which is a set of inclusion axioms of the forms  $C_l \sqsubseteq C_r$  and  $R \sqsubseteq R'$ , and an ABox  $\mathcal{A}$  which is a set of concept and role membership assertions of the forms  $A(a)$  and  $P(a_1, a_2)$ , with  $a, a_1, a_2 \in \mathcal{F}$ . We define the semantics of  $DL-Lite_{\mathcal{R}}$  through a translation to classical (resp., contextual) FOL, using the mapping function  $\pi$ , which is defined in Table 1;  $\pi$  extends naturally to sets of axioms and assertions.

**Table 1.** Mapping  $DL-Lite_{\mathcal{R}}$  to FOL

$\pi(A, X) = A(X)$	$\pi(C_l \sqsubseteq C_r) = \forall x(\pi(C_l, x) \supset \pi(C_r, x))$
$\pi(P, X, Y) = P(X, Y)$	$\pi(R_1 \sqsubseteq R_2) = \forall x, y(\pi(R_1, x, y) \supset \pi(R_2, x, y))$
$\pi(P^-, X, Y) = P(Y, X)$	
$\pi(\exists R, X) = \exists y(R(X, y))$	$\pi(A(a)) = A(a)$
$\pi(\neg A, X) = \neg \pi(A, X)$	$\pi(P(a_1, a_2)) = P(a_1, a_2)$
$\pi(\neg \exists R, X) = \neg \exists y(R(X, y))$	
$y$ is a new a variable	

Given a classical (resp., contextual)  $DL-Lite_{\mathcal{R}}$  knowledge base  $\mathcal{K}$ , the classical (resp., contextual) *FOL equivalent* of  $\mathcal{K}$  is the theory  $\Phi = \pi(\mathcal{K}) = \pi(\mathcal{T}) \cup \pi(\mathcal{A})$ .

*RDF* is a data language, where the central notion is the *graph*, which is a set of *triples* of the form  $\langle s, p, o \rangle$ ;  $s$  is the *subject*,  $p$  is the *predicate*, and  $o$  is the *object* of the triple.

A *vocabulary*  $V = \langle \mathcal{F}, \mathcal{PL}, \mathcal{TL}, \mathcal{B} \rangle$  consists of a set  $\mathcal{F}$  of URI references, a set  $\mathcal{PL}$  of plain literals (i.e. Unicode character strings with an optional language tag), a set  $\mathcal{TL}$  of typed literals (i.e. pairs  $(s, u)$  of a Unicode string  $s$  and a URI  $u$  denoting a datatype), and a set  $\mathcal{B}$  of blank nodes (i.e. existentially quantified variables); see [16, Sections 6.4, 6.5, 6.6] for more details about the specific form of these symbols. Terms are URI references, plain or typed literals, or blank nodes. A *graph*  $S$  of a vocabulary  $V$  is a set

of triples  $\langle s, p, o \rangle$ , with  $s, p, o \in \mathcal{F} \cup \mathcal{P}\mathcal{L} \cup \mathcal{T}\mathcal{L} \cup \mathcal{B}$ .<sup>4</sup> With  $bl(\langle s, p, o \rangle)$  (resp.,  $bl(S)$ ) we denote the set of blank nodes in  $\langle s, p, o \rangle$  (resp.,  $S$ ). A triple  $\langle s, p, o \rangle$  (resp., graph  $S$ ) is *ground* if  $bl(\langle s, p, o \rangle) = \emptyset$  (resp.,  $bl(S) = \emptyset$ ).

An interpretation of a vocabulary  $V$  is a tuple  $I = \langle U, U_P, U_L, I_{\mathcal{F}}, I_L, I_{ext} \rangle$ , where  $U$  is a countable non-empty set, called the domain,  $U_P$  is a countable set of properties,  $U_L \subseteq U$  is a countable set of literal values with  $\mathcal{P}\mathcal{L} \subseteq U_L$ ,  $I_{\mathcal{F}}$  is an interpretation function for URI references  $I_{\mathcal{F}} : \mathcal{F} \rightarrow U \cup U_P$ ,  $I_L$  is an interpretation function for typed literals  $I_L : \mathcal{T}\mathcal{L} \rightarrow U$ , and  $I_{ext}$  is an *extension function*  $I_{ext} : U_P \rightarrow 2^{(U \times U)}$ .

Given an interpretation  $I$  of a vocabulary  $V$  and a subset of the blank nodes  $\mathcal{B}' \subseteq \mathcal{B}$ , we define a mapping  $A : \mathcal{B}' \rightarrow U$  which is used to interpret blank nodes. For a term  $t$  we define  $t^{I,A}$  as: (a) if  $t \in \mathcal{F}$ , then  $t^{I,A} = I_{\mathcal{F}}(t)$ , (b) if  $t \in \mathcal{P}\mathcal{L}$ , then  $t^{I,A} = t$ , (c) if  $t \in \mathcal{T}\mathcal{L}$ , then  $t^{I,A} = I_L(t)$ , and (d) if  $t \in \mathcal{B}'$ , then  $t^{I,A} = A(t)$ .

An interpretation  $I$  *satisfies* a triple  $\langle s, p, o \rangle$  with respect to a mapping  $A : \mathcal{B}' \rightarrow U$ , with  $bl(\langle s, p, o \rangle) \subseteq \mathcal{B}'$ , denoted  $(I, A) \models \langle s, p, o \rangle$ , if  $p^{I,A} \in U_P$  and  $\langle s^{I,A}, o^{I,A} \rangle \in I_{ext}(p^{I,A})$ .  $I$  satisfies a graph  $S$  with respect to a mapping  $A : \mathcal{B}' \rightarrow U$ , denoted  $(I, A) \models S$ , if  $(I, A) \models \langle s, p, o \rangle$  for every  $\langle s, p, o \rangle \in S$ . An interpretation  $I$  is a *model* of a graph  $S$ , denoted  $I \models S$ , if  $(I, A) \models S$  for some  $A : \mathcal{B}' \rightarrow U$ .

Any interpretation is an *s-interpretation* (simple interpretation). An interpretation  $I$  is an *rdf-* (resp., *rdfs-*, *erdfs-*)interpretation if it interprets the RDF (resp., RDFS) vocabulary, satisfies the RDF (resp., RDFS) axiomatic triples, and satisfies a number of conditions, as specified in [11].

An RDF graph  $S$  *s-(resp., rdf-, rdfs-, or erdfs-)entails* an RDF graph  $E$  if every *s-(resp., rdf-, rdfs-, or erdfs-)interpretation* which is a model of  $S$  is also a model of  $E$ . We refer to these kinds of entailment as *entailment regimes*, and use the symbol  $\models_x$ , with  $x \in \{s, rdf, rdfs, erdfs\}$ , to denote entailment under the respective regimes.

Intuitively, the difference between the RDFS and eRDFS entailment regimes is that for the latter, whenever an ontological relation (e.g. subclass or property domain) implicitly holds in an interpretation, the corresponding RDF statement (`subclassOf`, `domain`) must be true, whereas this is not always the case with the RDFS entailment regime. The following example illustrates this difference.

*Example 1.* Let  $S$  be the following graph:  $S = \{\langle parent, domain, Person \rangle, \langle mother, subPropertyOf, parent \rangle\}$  which says that the domain of *parent* is *Person*, and the property *mother* is a sub-property of *parent*. Using eRDFS entailment we can conclude from  $S$  that the domain of *mother* is also *Person*:

$$S \models_{erdfs} \langle mother, domain, Person \rangle;$$

it is always the case that the subject of any *mother* triple has the type *Person*; thus, *mother* implicitly has the domain *Person*. We cannot draw this conclusion when using RDFS entailment; in RDFS, only explicitly asserted domain constraints can be derived.

<sup>4</sup> Note that we allow literals in subject ( $s$ ), and literals and blank nodes in predicate ( $p$ ) positions, whereas the RDF specification [16] does not. Nonetheless, our results immediately apply to standard RDF graphs as defined in [16].

### 3 RDF(S) Embedding

In this section we first define an embedding of the various entailment regimes in F-Logic. We then consider an embedding of eRDFS entailment in FOL and DL.

#### 3.1 Embedding RDF in F-Logic

We embed a graph as a conjunction of data molecules; URI references and literals are treated as constant symbols, and blank nodes are treated as existentially quantified variables. In the remainder we assume that RDF graphs are finite.

Given a vocabulary  $V = \langle \mathcal{F}, \mathcal{P}\mathcal{L}, \mathcal{T}\mathcal{L}, \mathcal{B} \rangle$ , an F-language  $\mathcal{L}$  corresponds to  $V$  if it has a signature of the form  $\Sigma = \langle \mathcal{F}' \supseteq \mathcal{F} \cup \mathcal{P}\mathcal{L} \cup \mathcal{T}\mathcal{L}, \mathcal{P} \rangle$ <sup>5</sup>

**Definition 1.** Let  $S$  be an RDF graph of a vocabulary  $V$ , let  $\langle s, p, o \rangle \in S$  be a triple in  $S$ , and let  $\mathcal{L}$  be an F-language which corresponds to  $V$ . Then,

$$\begin{aligned} (tr(\langle s, p, o \rangle) \in \mathcal{L}) &= s[p \rightarrow o] \text{ and} \\ (tr(S) \in \mathcal{L}) &= \{ \exists bl(S) (\bigwedge \{ tr(\langle s, p, o \rangle) \mid \langle s, p, o \rangle \in S \}) \}. \end{aligned}$$

If  $\phi$  is an F-Logic formula or theory in prenex normal form with only existential quantifiers, then  $\phi^{sk}$  denotes the *Skolemization* of  $\phi$ , i.e. every existentially quantified variable is replaced with a globally unique new constant symbol.

We use a set of formulas  $\Psi^x \subseteq \mathcal{L}$ , as defined in Table 2 to axiomatize the semantics of an entailment regime  $x \in \{s, rdf, rdfs, erdfs\}$ .

**Proposition 2.** Let  $S$  be an RDF graph of a vocabulary  $V$ . Then,  $tr(S)^{sk} \cup \Psi^x$ , with  $x \in \{s, rdf, rdfs\}$ , can be equivalently rewritten to a set of F-Logic Datalog formulas.

Note that  $\Psi^{erdfs}$  cannot be equivalently rewritten to a set of Datalog formulas, because of the use of universal quantification in the antecedents of some of the implications in  $\Psi^{erdfs}$ .

**Theorem 1.** Let  $S$  and  $E$  be RDF graphs of a vocabulary  $V$  and let  $x \in \{s, rdf, rdfs, erdfs\}$  be an entailment regime. Then,

$$\begin{aligned} S \models_x E &\text{ iff } tr(S) \cup \Psi^x \models_{\mathfrak{f}} tr(E), \text{ and} \\ S \text{ is } x\text{-satisfiable} &\text{ iff } tr(S) \cup \Psi^x \text{ has a model.} \end{aligned}$$

The following corollary follows immediately from Theorem 1 and the classical results about Skolemization (see e.g. [9]). For the case of s-entailment, the result was implicitly stated in [11, Skolemization lemma].

**Corollary 1.** Let  $S$  and  $E$  be RDF graphs and let  $x \in \{s, rdf, rdfs, erdfs\}$  be an entailment regime. Then,

$$S \models_x E \quad \text{iff} \quad tr(S)^{sk} \cup \Psi^x \models_{\mathfrak{f}} tr(E).$$

<sup>5</sup> Even though typed literals are pairs in RDF, we treat them simply as constant symbols in our embedding.

**Table 2.** Axiomatization of the RDF entailment regimes

$\Psi^s = \emptyset$
$\Psi^{rdf} = \Psi^s \cup \{tr(\langle s, p, o \rangle) \mid \langle s, p, o \rangle \text{ is an RDF axiomatic triple}\} \cup$ $\{wellxml(t) \mid t \in \mathcal{TL} \text{ is a well-typed XML literal}\} \cup$ $\{illxml(t) \mid t \in \mathcal{TL} \text{ is an ill-typed XML literal}\} \cup$ $\{\forall x(\exists y, z(y[x \rightarrow z]) \supset x[\text{type} \rightarrow \text{Property}]),$ $\forall x(wellxml(x) \supset x[\text{type} \rightarrow \text{XMLLiteral}]),$ $\forall x(x[\text{type} \rightarrow \text{XMLLiteral}] \wedge illxml(x) \supset \perp)\}$
$\Psi^{rdfs} = \Psi^{rdf} \cup \{tr(\langle s, p, o \rangle) \mid \langle s, p, o \rangle \text{ is an RDFS axiomatic triple}\} \cup$ $\{pl(t) \mid t \in \mathcal{PL}\} \cup$ $\{\forall x, y, z(x[y \rightarrow z] \supset x[\text{type} \rightarrow \text{Resource}] \wedge z[\text{type} \rightarrow \text{Resource}]),$ $\forall u, v, x, y(x[\text{domain} \rightarrow y] \wedge u[x \rightarrow v] \supset u[\text{type} \rightarrow y]),$ $\forall u, v, x, y(x[\text{range} \rightarrow y] \wedge u[x \rightarrow v] \supset v[\text{type} \rightarrow y]),$ $\forall x(x[\text{type} \rightarrow \text{Property}] \supset x[\text{subPropertyOf} \rightarrow x]),$ $\forall x, y, z(x[\text{subPropertyOf} \rightarrow y] \wedge y[\text{subPropertyOf} \rightarrow z] \supset x[\text{subPropertyOf} \rightarrow z]),$ $\forall x, y(x[\text{subPropertyOf} \rightarrow y] \supset \forall z_1, z_2(z_1[x \rightarrow z_2] \supset z_1[y \rightarrow z_2])),$ $\forall x(x[\text{type} \rightarrow \text{Class}] \supset x[\text{subClassOf} \rightarrow \text{Resource}]),$ $\forall x, y(x[\text{subClassOf} \rightarrow y] \supset \forall z(z[\text{type} \rightarrow x] \supset z[\text{type} \rightarrow y])),$ $\forall x(x[\text{type} \rightarrow \text{Class}] \supset x[\text{subClassOf} \rightarrow x]),$ $\forall x, y, z(x[\text{subClassOf} \rightarrow y] \wedge y[\text{subClassOf} \rightarrow z] \supset x[\text{subClassOf} \rightarrow z]),$ $\forall x(x[\text{type} \rightarrow \text{ContainerMembershipProperty}] \supset x[\text{subPropertyOf} \rightarrow \text{member}]),$ $\forall x(x[\text{type} \rightarrow \text{Datatype}] \supset x[\text{subClassOf} \rightarrow \text{Literal}]),$ $\forall x(pl(x) \supset x[\text{type} \rightarrow \text{Literal}]),$ $\forall x(x[\text{type} \rightarrow \text{Literal}] \wedge illxml(x) \supset \perp)\}$
$\Psi^{erdfs} = \Psi^{rdfs} \cup \{\forall x, y(\forall u, v(u[x \rightarrow v] \supset u[\text{type} \rightarrow y]) \supset x[\text{domain} \rightarrow y]),$ $\forall x, y(\forall u, v(u[x \rightarrow v] \supset v[\text{type} \rightarrow y]) \supset x[\text{range} \rightarrow y]),$ $\forall x, y(x[\text{type} \rightarrow \text{Property}] \wedge y[\text{type} \rightarrow \text{Property}] \wedge \forall u, v(u[x \rightarrow v] \supset$ $u[y \rightarrow v]) \supset x[\text{subPropertyOf} \rightarrow y]),$ $\forall x, y(x[\text{type} \rightarrow \text{Class}] \wedge y[\text{type} \rightarrow \text{Class}] \wedge \forall u(u[\text{type} \rightarrow x] \supset u[\text{type} \rightarrow y]) \supset$ $x[\text{subClassOf} \rightarrow y])\}$

Since, by Proposition 2,  $tr(S)^{sk}$ ,  $tr(S)^{sk} \cup \Psi^{rdf}$  and  $tr(S)^{sk} \cup \Psi^{rdfs}$  are equivalent to sets of Datalog formulas, this result implies that simple, RDF, and RDFS entailment can be computed using existing F-Logic rule reasoners<sup>6</sup> such as FLORA-2, and Ontobroker, as well as any rule reasoners which supports Datalog (see Proposition 1). Notice that, in the corollary,  $tr(E)$  can be seen as a boolean conjunctive query (i.e. a yes/no query) in which the existentially quantified variables in  $tr(E)$  are the non-distinguished variables.

We now consider an alternative, direct embedding of the extensional RDFS semantics (erdfs-entailment) which eliminates part of the RDFS vocabulary from the embedded graph, yielding a set of Datalog formulas.

We first define the notion of *nonstandard use* of the RDFS vocabulary, which intuitively corresponds to using the vocabulary in locations where it has not been intended,

<sup>6</sup> Note that a Datalog formula with  $\perp$  in the antecedent corresponds to an integrity constraint, i.e. a query which may not have an answer set.

for example in places where it redefines the semantics of RDF constructs such as in the triple  $\langle \text{type}, \text{subPropertyOf}, a \rangle$ .

We say that a term  $t$  occurs in a *property position* if it occurs as the predicate of a triple, as the subject or object of a `subPropertyOf` triple, as the subject of a domain or range triple, or as the subject of a triple  $\langle t, \text{type}, \text{Property} \rangle$  or  $\langle t, \text{type}, \text{ContainerMembershipProperty} \rangle$ . A term  $t$  occurs in a *class position* if it occurs as the subject or object of a `subClassOf` triple, as the object of a domain, range, or type triple, as the subject of a triple  $\langle t, \text{type}, \text{Class} \rangle$  or  $\langle t, \text{type}, \text{Datatype} \rangle$ . Otherwise, we say that  $t$  occurs in an *individual position*.

**Definition 2.** Let  $S$  be an RDF graph. Then,  $S$  has nonstandard use of the RDFS vocabulary if

- `type`, `subClassOf`, `domain`, `range` or `subPropertyOf` occurs in the subject or object position of a triple in  $S$  or
- `ContainerMembershipProperty`, `Resource`, `Class`, `Datatype` or `Property` occurs in any position other than the object position of a type-triple in  $S$ .

We conjecture that large classes of RDF graphs will not have any nonstandard use of the RDFS vocabulary. We now proceed to define a direct embedding of the extensional RDFS entailment regime in F-Logic.

**Definition 3.** Let  $\langle s, p, o \rangle$  be an RDF triple. Then,

$$\begin{aligned} tr^{erdfs}(\langle s, \text{type}, \text{Datatype} \rangle) &= \forall x(x : s \supset x : \text{Literal}), \\ tr^{erdfs}(\langle s, \text{type}, o \rangle) &= s : o, \\ tr^{erdfs}(\langle s, \text{subClassOf}, o \rangle) &= \forall x(x : s \supset x : o), \\ tr^{erdfs}(\langle s, \text{subPropertyOf}, o \rangle) &= \forall x, y(x[s \rightarrow y] \supset x[o \rightarrow y]), \\ tr^{erdfs}(\langle s, \text{domain}, o \rangle) &= \forall x, y(x[s \rightarrow y] \supset x : o), \\ tr^{erdfs}(\langle s, \text{range}, o \rangle) &= \forall x, y(x[s \rightarrow y] \supset y : o), \text{ and} \\ tr^{erdfs}(\langle s, p, o \rangle) &= s[p \rightarrow o], \text{ otherwise.} \end{aligned}$$

Let  $S$  be an RDF graph of a vocabulary  $V = \langle \mathcal{F}, \mathcal{P}\mathcal{L}, \mathcal{T}\mathcal{L}, \mathcal{B} \rangle$ . Then,

$$\begin{aligned} tr^{erdfs}(S) &= \{\exists bl(S) \wedge \{tr^{erdfs}(\langle s, p, o \rangle) \mid \langle s, p, o \rangle \in S\}\}, \text{ and} \\ \Psi^{erdfs-V} &= \{tr^{erdfs}(\langle s, p, o \rangle) \mid \langle s, p, o \rangle \text{ is an RDF(S) axiomatic triple with no non-} \\ &\quad \text{standard use of the RDF(S) vocabulary}\} \cup \{t : \text{XMLLiteral} \mid t \in \mathcal{T}\mathcal{L} \text{ is a well-} \\ &\quad \text{typed XML literal}\} \cup \{\text{ill:xml}(t) \mid t \in \mathcal{T}\mathcal{L} \text{ is an ill-typed XML literal}\} \cup \\ &\quad \{t : \text{Literal} \mid t \in \mathcal{P}\mathcal{L}\} \cup \{\forall x(x : \text{Literal} \wedge \text{ill:xml}(x) \supset \perp)\} \end{aligned}$$

The *property* (resp., *class*) *vocabulary* of an RDF graph  $S$  consists of all the symbols occurring in property (resp., class) positions in  $S$  and the RDF(S) axiomatic triples with no nonstandard use of the RDF(S) vocabulary.

Given two RDF graphs  $S$  and  $E$ , we write  $E \triangleleft S$  if the property and class vocabularies of  $E$  are subsets of the property and class vocabularies of  $S$  (modulo blank node renaming and instantiation, i.e. replacement of blank nodes with URI references or literals) and `Resource`, `ContainerMembershipProperty`, `Class`, `Property` and `Datatype` do not occur in  $E$ .

**Theorem 2.** Let  $S, E$  be RDF graphs with no nonstandard use of the RDFS vocabulary such that Resource, Class, Property, ContainerMembershipProperty and Datatype do not occur in  $E$ . Then,

- whenever  $E \sqsubseteq S$ ,

$$S \models_{\text{erdfs}} E \text{ iff } \text{tr}^{\text{erdfs}}(S) \cup \Psi^{\text{erdfs}-V} \models_{\text{f}} \text{tr}^{\text{erdfs}}(E);$$

- $(\text{tr}^{\text{erdfs}}(S))^{\text{sk}}$  is a conjunction of F-Logic Datalog formulas, and whenever  $E$  does not contain the terms `subClassOf`, `domain`, `range`, and `subPropertyOf`,  $\text{tr}^{\text{erdfs}}(E)$  is a conjunction of atomic molecules prefixed by an existential quantifier and

$$S \models_{\text{erdfs}} E \text{ iff } (\text{tr}^{\text{erdfs}}(S))^{\text{sk}} \cup \Psi^{\text{erdfs}-V} \models_{\text{f}} \text{tr}^{\text{erdfs}}(E).$$

Since  $(\text{tr}^{\text{erdfs}}(S))^{\text{sk}} \cup \Psi^{\text{erdfs}-V}$  is a set of Datalog formulas we have that, if the RDF graphs fulfill certain conditions, query answering techniques from the area of deductive databases can be used for checking extensional RDFS entailment.

### 3.2 Embedding Extensional RDFS in First-Order Logic

We now consider an embedding of extensional RDFS entailment in FOL, based on the direct embedding of extensional RDFS in F-Logic considered above (Definition 3).

An F-Logic theory or formula is *translatable* to contextual FOL if for molecules of the forms  $t_1[t_2 \rightarrow t_3]$  and  $t_1 : t_2$  holds that  $t_2$  is a constant symbol (i.e. 0-ary function symbol).

Let  $\Phi$  (resp.,  $\phi$ ) be an F-Logic theory (resp., formula) which is translatable to contextual FOL, then  $(\Phi)^{\text{FO}}$  (resp.,  $(\phi)^{\text{FO}}$ ) is the contextual FOL theory obtained from  $\Phi$  (resp.,  $\phi$ ) by:

- replacing every data molecule  $t_1[t_2 \rightarrow t_3]$  with  $t_2(t_1, t_3)$ , and
- replacing every is-a molecule  $t_1 : t_2$  with  $t_2(t_1)$ .

The following proposition follows immediately from a result in [6].

**Proposition 3.** Let  $\Phi, \phi$  be an equality-free F-Logic theory and formula which are translatable to contextual FOL. Then,

$$\Phi \models_{\text{f}} \phi \text{ iff } (\Phi)^{\text{FO}} \models_{\text{c}} (\phi)^{\text{FO}}.$$

An RDF graph  $S$  is a *non-higher-order* RDF graph if  $S$  does not contain blank nodes in class and property positions, and does not contain nonstandard use of the RDFS vocabulary. A non-higher-order RDF graph  $S$  is a *classical* RDF graph if the sets of URIs occurring in individual, class and property positions in  $S$  and its context (e.g. entailing or entailed graph) are mutually disjoint. Notice that every ground RDF graph which does not contain nonstandard use of the RDFS vocabulary is a non-higher-order RDF graph. One can also verify that every OWL DL graph, as defined in [19], is a classical RDF graph, but there are classical RDF graphs which are not OWL DL graphs.

The following theorem identifies subsets of extensional RDFS which have a natural correspondence to contextual and classical FOL. Observe that if  $S$  is a non-higher-order RDF graph, then  $\text{tr}^{\text{erdfs}}(S)$  is translatable to contextual FOL.

**Theorem 3.** *Let  $S$  and  $E$  be non-higher-order RDF graphs such that  $E \trianglelefteq S$ . Then,*

$$S \models_{\text{erdfs}} E \text{ iff } (tr^{\text{erdfs}}(S))^{FO} \models_c (tr^{\text{erdfs}}(E))^{FO}.$$

*If, additionally,  $S, E$  are classical graphs, then  $(tr^{\text{erdfs}}(S))^{FO}$  and  $(tr^{\text{erdfs}}(E))^{FO}$  are theories of classical first-order logic, and*

$$S \models_{\text{erdfs}} E \text{ iff } (tr^{\text{erdfs}}(S))^{FO} \models (tr^{\text{erdfs}}(E))^{FO}.$$

**Proposition 4.** *Let  $S$  be a ground non-higher-order graph<sup>7</sup>. Then,  $(tr^{\text{erdfs}}(S))^{FO}$  can be equivalently rewritten to the FOL equivalent  $\Phi$  of a contextual DL-Lite $_{\mathcal{R}}$  knowledge base  $\mathcal{K}$ .*

*If  $S$  is a classical RDF graph, then  $(tr^{\text{erdfs}}(S))^{FO}$  can be equivalently rewritten to the FOL equivalent  $\Phi$  of a classical DL-Lite $_{\mathcal{R}}$  knowledge base.*

## 4 Complexity

In this section we review the complexity of the various forms of entailment in RDF and present several novel results, based on the embeddings presented in the previous section.

The complexity of simple entailment and RDFS entailment is well known, and the complexity of RDF entailment follows immediately. Note that, although the set of axiomatic triples is infinite, only a finite subset, linear in the size of the graphs, needs to be taken into account when checking entailment.

**Proposition 5** ( [10, 13, 4] ). *Let  $S$  and  $E$  be graphs. Then, the problem of checking  $S \models_s E$ ,  $S \models_{\text{rdf}} E$ , or  $S \models_{\text{rdfs}} E$  is **NP-complete** in the combined size of the graphs, and polynomial in the size of  $S$ . If  $E$  is ground, then the respective problems are polynomial in the combined size of the graphs.*

*Additionally, the problem of checking  $S \models_{\text{erdfs}} E$  is **NP-hard** in the size of the graphs.*

The membership proofs in [10, 13, 4] rely on the fact that the set of all (relevant) entailed triples of a given graph can be computed in polynomial time using the RDFS entailment rules [13]; the problem can then be reduced to subgraph homeomorphism. Using Corollary 1 and the fact that the problem of checking ground entailment in Datalog [8] is polynomial in the size of the data  $(tr(S))$  gives to a novel argument for membership.

NP-hardness can be shown through a reduction from a known NP-hard problem (e.g. 3SAT).

From the embedding in F-Logic, together with the complexity of nonrecursive Datalog [8], we obtain the following novel characterization of the complexity of simple and RDF entailment.

<sup>7</sup> Note that, when considering a variant of DL-Lite $_{\mathcal{R}}$  which allows existentials in the ABox – also allowed in OWL DL – this restriction could be relaxed to  $S$  being a non-higher-order RDF graph with no blank nodes outside of individual positions.



**Theorem 4.** *Let  $S$  and  $E$  be RDF graphs. Then, the problems  $S \models_s E$  and  $S \models_{rdf} E$  are in *LogSpace* in the size of  $S$ , and in the combined size of the graphs if  $E$  is ground.*

Using the correspondence of Proposition 4 the results on the complexity of reasoning in *DL-Lite<sub>R</sub>* [7], and the classical results on skolemization [9] we obtain the following result for extensional RDFS entailment.

**Theorem 5.** *Let  $S$  and  $E$  be RDF graphs with no nonstandard use of the RDFS vocabulary such that  $E \sqsubseteq S$ . Then, the problem of deciding  $S \models_{erdfs} E$  is *NP-complete* in the size of the graphs, and polynomial if  $E$  is ground.*

**Table 3.** Complexity of Entailment  $S \models_x E$  in RDF, measured in the size of  $S, E$

Entailment	Restrictions on $S$	Restrictions on $E$	Complexity
$\models_s, \models_{rdf}, \models_{rdfs}$	none	none	NP-complete
$\models_s, \models_{rdf}$	none	ground	LogSpace
$\models_{rdfs}$	none	ground	P
$\models_{erdfs}$	none	none	NP-hard
$\models_{erdfs}$	no nonst. RDFS	no nonst. RDFS	NP-complete
$\models_{erdfs}$	no nonst. RDFS	ground, no nonst. RDFS	P

Table 3 summarizes the complexity of reasoning with the entailment regimes of RDF; “No nonst. RDFS” stands for “no nonstandard use of the RDFS vocabulary;  $S$  and  $E$  are such that the property and class vocabularies of  $E$  are subsets of the property and class vocabularies of  $S$  (modulo blank node renaming and instantiation); and Resource, Class, Property, ContainerMembershipProperty and Datatype do not occur in  $E$ ”. The results in the first and third line of the table were obtained in [10, 4, 13], and the fourth line follows immediately. To the best of our knowledge, the other results are novel.

## 5 Querying

In this section we consider conjunctive queries over RDF graphs using the RDF entailment regimes we considered throughout this paper.

Given a countable set  $\mathcal{V}$  of variable symbols, disjoint from the symbols in  $V$ , a *generalized RDF triple* is a tuple of the form  $\langle s, p, o \rangle$ , with  $s, p$  and  $o$  terms or variable symbols. A *conjunctive query*  $q(x)$  over an RDF graph  $S$  is a set of generalized RDF triples  $q(x)$  such that  $x$  is a vector of variables occurring in  $q$ , also called the *distinguished variables* of  $q$ ; the blank nodes occurring in  $q$ ,  $bl(q)$ , are the *non-distinguished variables* of  $q$ .

Given an RDF graph  $S$  and conjunctive query  $q(x)$ , then a tuple of terms  $\mathbf{a}$  is an *answer* of a query under  $x$ -entailment if  $S \models_x q(\mathbf{a})$ , with  $x \in \{s, rdf, rdfs, erdfs\}$  an entailment regime. The complexity of query answering is related to the complexity of the corresponding recognition problem: the recognition problem associated with a query  $q(x)$  is the decision problem of checking whether, given an RDF graph  $S$  and



a tuple  $\mathbf{a}$  of terms, the entailment  $S \models_x q(\mathbf{a})$  holds. The *data complexity* of query answering under the  $x$  entailment regime corresponds to the complexity of the corresponding recognition problem, in the size of  $S$ .

From the preceding results on the complexity of the various entailment regimes we obtain the following characterization of the complexity of query answering.

**Theorem 6.** *Let  $S$  be an RDF graph, let  $x \in \{s, rdf, rdfs, erdfs\}$  be an entailment regime, and let  $q(x)$  be a conjunctive query. Then, the data complexity of query answering under the  $x$  entailment regime is*

- in *LogSpace*, if  $x \in \{s, rdf\}$  and
- *polynomial*, if  $x \in \{rdfs\}$ .

## 6 Discussion and Related Work

In this section we discuss implications of the results in this paper, and place it in the context of related work. We distinguish between work done on RDF and on RDF querying.

*RDF.* There have been several investigations [10, 4, 13] into the semantics of RDF. The investigation in [10] reconstructs the semantics from a graph database perspective, and the one in [4] reconstructs the semantics from a logical language perspective. The investigation of the RDF semantics in [13] stays very close to the RDF specification. Additionally, [13] shows that the entailment rules presented in the original specification [11] are not complete with respect to the semantics. These reconstructions have led to a number of complexity results for RDF entailment. In this paper, we built upon these results and complemented them with several novel results for simple, RDF, and extensional RDFS entailment.

The investigation in [4] is close in spirit to our investigation, albeit that [4] bases its logical reconstruction on (contextual) first-order logic, rather than F-Logic.

RDFS(FA) [18] defines a new (extensional) semantics for RDFS which is in line with the semantics of OWL DL, as well as a number of syntactic restrictions to achieve a layered meta-modeling architecture. It is currently not known what the precise relationship is between RDFS(FA) and the RDFS semantics defined in the standard [11].

Finally, we mention [17], in which the authors identify a syntactic subset of RDFS which allows for efficient reasoning ( $O(n \log n)$ ), while still being expressive enough to capture large classes of ontologies.

*RDF Querying.* SPARQL [22] is a query language for RDF, currently under development at W3C. Of all the RDF entailment regimes, SPARQL currently only considers simple entailment. However, the use of other regimes is considered a possible future extension.

The queries we considered in Section 5 are conjunctive queries and correspond to what are called “[SPARQL] graph pattern expressions constructed by using only AND” in [20]. Therefore, not surprisingly, data complexity of conjunctive query answering when considering simple entailment corresponds to the data complexity of evaluating such graph pattern expressions; they are both in *LogSpace* (cf. Theorem 6 and [20, Theorem 4]).

Finally we mention [21], in which a translation from SPARQL queries to Datalog is described. The combination of such a translation with an embedding of the RDF or RDFS semantics, as described in Theorem 1 could be used for evaluating SPARQL queries using the respective entailment regimes.

## 7 Conclusions and Future Work

We have presented embeddings of the different RDF entailment regimes in F-Logic, and we have shown how deductive database and description logic technology can be used for reasoning with RDF. An implementation of answering conjunctive queries over RDF graphs under the RDF and RDFS entailment regimes, and restricted RDF graphs under the eRDFS entailment regime, based on the Datalog reasoner IRIS<sup>8</sup>, can be found at: <http://tools.deri.org/rdfs-reasoner>. It is planned to extend this reasoner with support for more expressive query languages, such as SPARQL, considering the embedding in Datalog presented in [21].

In the course of our investigation we have presented several novel complexity results. To the best of our knowledge, ours is the first comprehensive investigation of the extensional RDFS entailment regime. These results could be used for, for example, rule-based extensions of RDF, or increasing the alignment between RDF and logic-based semantic Web languages (e.g. OWL DL).

Our future work includes the extension of the considered embeddings with more extensive treatment of datatypes, in the form of  $D$ -entailment [11], and  $D^*$ -entailment [13], as well as more expressive query languages such as SPARQL.

Several proposals have been made for rule extensions of RDF graphs (e.g. [2,14,1]), and several rule-based systems which deal with RDF exist (e.g. Jena, CWM). In an earlier version of the present paper [5] we considered extensions of RDF graphs with logical rules and axioms, based on the embeddings we have presented. However, such extensions are not entirely faithful with respect to the model-theoretic semantics of RDF. Therefore, our future work includes an investigation of combinations of logical rules and RDF based on a notion of common models, i.e. an interpretation is a model of a combination if it is a model of both the logical theory and the RDF graph.

Finally, we plan to investigate the precise relationship between eRDFS and OWL DL entailment, taking the subset compatible with  $DL-Lite_{\mathcal{R}}$  (see Proposition 4) as a starting point.

## References

1. Analyti, A., Antoniou, G., Damásio, C.V., Wagner, G.: Stable model theory for extended RDF ontologies. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 21–36. Springer, Heidelberg (2005)
2. Berners-Lee, T., Connolly, D., Kagal, L., Scharfand, Y., Hendler, J.: N3Logic: A logic for the web. Journal of Theory and Practice of Logic Programming (TLP), Special Issue on Logic Programming and the Web (2007)

<sup>8</sup> <http://sourceforge.net/projects/iris-reasoner>

3. Brickley, D., Guha, R.V.: RDF vocabulary description language 1.0: RDF schema. Recommendation 10 February 2004, W3C (2004)
4. Bruijn, J., de Franconi, E., Tessaris, S.: Logical reconstruction of normative RDF. In: OWLED 2005. Proceedings of the Workshop OWL: Experiences and Directions (2005)
5. Bruijn, J., de Heymans, S.: RDF and logic: Reasoning and extension. In: Proceedings of the 6th International Workshop on Web Semantics (WebS 2007), in conjunction with the 18th International Conference on Database and Expert Systems Applications (DEXA 2007) (2007)
6. Bruijn, J., de Heymans, S.: On the relationship between description logic-based and f-logic-based ontologies. In: *Fundamenta Informaticae* (accepted for publication, 2008)
7. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Data complexity of query answering in description logics. In: KR 2006. Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning, pp. 260–270 (2006)
8. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity and expressive power of logic programming. *ACM Computing Surveys* 33(3), 374–425 (2001)
9. Fitting, M.: *First Order Logic and Automated Theorem Proving*, 2nd edn. Springer, Heidelberg (1996)
10. Gutierrez, C., Hurtado, C., Mendelzon, A.O.: Foundations of semantic web databases. In: PODS 2004. Proceedings of the 23rd ACM Symposium on Principles of Database Systems, pp. 95–106. ACM Press, New York (2004)
11. Hayes, P.: RDF semantics. Recommendation 10 February 2004, W3C (2004)
12. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: van: From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics* 1(1), 7–26 (2003)
13. ter Horst, H.J.: Completeness, decidability and complexity of entailment for RDF schema and a semantic extension involving the OWL vocabulary. *Journal of Web Semantics* 3(2–3), 79–115 (2005)
14. ter Horst, H.J.: ter: Combining RDF and part of OWL with rules: Semantics, decidability, complexity. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
15. Kifer, M., Lausen, G., Wu, J.: Logical foundations of object-oriented and frame-based languages. *Journal of the ACM* 42(4), 741–843 (1995)
16. Klyne, G., Carroll, J.J.: Resource description framework (RDF): Concepts and abstract syntax. Recommendation 10 February 2004, W3C (2004)
17. Muñoz, S., Pérez, J., Gutierrez, C.: Minimal deductive systems for RDF. In: ESWC 2007. Proceedings of the 4th European Semantic Web Conference (2007)
18. Pan, J.Z., Horrocks, I.: RDFS(FA): Connecting RDF(S) and OWL DL. 19(2), 192–206 (2007)
19. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL web ontology language semantics and abstract syntax. Recommendation 10 February 2004, W3C (2004)
20. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 30–43. Springer, Heidelberg (2006)
21. Polleres, A.: From SPARQL to rules (and back). In: WWW 2007. Proceedings of the 16th International World Wide Web Conference, pp. 787–796 (2007)
22. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF. Working Draft March 26, 2007, W3C (2007)

# Scalable Cleanup of Information Extraction Data Using Ontologies

Julian Dolby<sup>1</sup>, James Fan<sup>1</sup>, Achille Fokoue<sup>1</sup>, Aditya Kalyanpur<sup>1</sup>,  
Aaron Kershenbaum<sup>1</sup>, Li Ma<sup>2</sup>, William Murdock<sup>1</sup>, Kavitha Srinivas<sup>1</sup>,  
and Christopher Welty<sup>1</sup>

<sup>1</sup> IBM Watson Research Center, P.O. Box 704, Yorktown Heights, NY 10598, USA  
{fanj,dolby,achille,adityakal,aaronk,murdockj,ksrinivs,welty}@us.ibm.com

<sup>2</sup> IBM China Research Lab, Beijing 100094, China  
malli@cn.ibm.com

**Abstract.** The approach of using ontology reasoning to cleanse the output of information extraction tools was first articulated in SemantiClean. A limiting factor in applying this approach has been that ontology reasoning to find inconsistencies does not scale to the size of data produced by information extraction tools. In this paper, we describe techniques to scale inconsistency detection, and illustrate the use of our techniques to produce a consistent subset of a knowledge base with several thousand inconsistencies.

## 1 Introduction

The original vision of the semantic web was concerned with publishing the semantics of, and inter-connecting, the back-end databases that generate the vast majority of HTML content. A common misunderstanding about the semantic web is that the vision somehow hinges on manual markup of natural language text on web pages with semantic labels the way the original web's HTML markup was done.

While this is a misunderstanding and not really a valid criticism of the semantic web vision, the vast majority of knowledge on the web, and in organizations, is in natural language text. Exploiting this knowledge in automated ways is an important scientific and economic problem, and should not be ignored, however it is not always clear whether semantic web technology can really make any difference.

In previous work, we have reported on SemantiClean [1], a system to clean up natural language processing results using an OWL-DL reasoner that has been shown in experiments to improve the precision of relation analysis. The main shortcoming of the SemantiClean work was scalability. In this paper, we report on initial experiments to use a Scalable Highly Expressive Reasoner (SHER) [2] [3], to bring the SemantiClean approach up to the scale of current information extraction technology and to provide explanations for removed assertions.

The paper is organized as follows. After a brief background section, Section 3 presents our approach for scalable cleanup. Section 4 discusses presentation of explanations for removed assertions. The results of our experimental evaluation is presented in Section 5. In Section 6, we describe more advanced cleanup strategies. Section 7 and 8 provide the related work and conclusions.

## 2 Background

### 2.1 SemantiClean

The most problematic kind of extraction produced by natural language components we have experienced is relation extraction - the identification of relationships and their arguments in natural language text. A common type of error we see in extracted relations is the violation of simple domain and range constraints. For example, in the following sentence:

... the decision in September 1991 to withdraw tactical *nuclear bombs*, missiles and torpedoes from US Navy ships ...

our analytics extract an ownership relation in the italicized text between nuclear (annotated as a weapon), and bombs (also a weapon), which maps to a `ownerOf` relation in the ontology. The `ownerOf` relation has a restriction limiting the domain to `Person` or `Organization` or `GPE` and a disjointness constraint between each of these and `Weapon`.

The SemantiClean approach is a simple one. We start with an ontology expressed in OWL-DL that must include negation (i.e. it must be possible to generate a contradiction in the Abox). Relations are extracted from text and stored as an RDF model that instantiates the ontology. As the relations are extracted into RDF, we construct an intermediate model. With each relation added, we run the model through a consistency check using Pellet [4], an in-memory reasoner. If it is not consistent, we drop the triple, if it is consistent, we add the triple to the final RDF model.

This triple-at-a-time technique has two problems. The main problem is scale: our information extraction technology can process, on normal desktop hardware, roughly a million of documents a day, the SemantiClean system could process hundreds of documents a day. Our analytics today produce about 70 entities (RDF nodes) and about 40 relations (RDF triples) per document, but these numbers can easily change. The point is that the size of the RDF graph could be two orders of magnitude larger than the number of documents.

A second problem is an order dependency created by the triple-by-triple approach of dropping the triple that, when added, causes the knowledge-base to change from consistent to inconsistent. When constraint violations arise from multiple triples together, it is possible that the dropped triple is not the incorrect one.

In two separate experiments, we found SemantiClean improved overall precision of relation extraction by 8% and 15%. As we continue to experiment, there

is evidence to indicate that the precision improvement may increase with the size of the graph, however we have reached the resource limits of Pellet on conventional 64-bit hardware. In the next sections, we describe techniques to scale cleanup.

## 2.2 Summarization

The techniques we apply in this paper assume ontologies of SHIN expressiveness. A key feature of our approach is the construction of a summary Abox  $\mathcal{A}'$  corresponding to the Abox  $\mathcal{A}$ . An individual in  $\mathcal{A}'$  represents individuals in  $\mathcal{A}$  which are members of the same concepts. Formally, an Abox  $\mathcal{A}'$  is a summary Abox of a SHIN Abox  $\mathcal{A}$  if there is a mapping function  $\mathbf{f}$  that satisfies the following constraints:

- (1) if  $a : C \in \mathcal{A}$  then  $\mathbf{f}(a) : C \in \mathcal{A}'$
- (2) if  $R(a, b) \in \mathcal{A}$  then  $R(\mathbf{f}(a), \mathbf{f}(b)) \in \mathcal{A}'$
- (3) if  $a \neq b \in \mathcal{A}$  then  $\mathbf{f}(a) \neq \mathbf{f}(b) \in \mathcal{A}'$

The *image* of an individual  $u$  in  $\mathcal{A}'$  is the set, denoted  $Image(u)$ , of individuals  $a$  in  $\mathcal{A}$  such that  $\mathbf{f}(a) = u$ . The *accurate image* of a subset  $L$  of a summary  $\mathcal{A}'$ , denoted  $AccImage(L)$ , is the subset of  $\mathcal{A}$  defined as  $AccImage(L) = \{a : C \in \mathcal{A} \mid \mathbf{f}(a) : C \in L\} \cup \{R(a, b) \in \mathcal{A} \mid R(\mathbf{f}(a), \mathbf{f}(b)) \in L\} \cup \{a \neq b \in \mathcal{A} \mid \mathbf{f}(a) \neq \mathbf{f}(b) \in L\}$ .

If the summary Abox  $\mathcal{A}'$ , obtained by applying the mapping function  $\mathbf{f}$  to  $\mathcal{A}$  is consistent w.r.t. a Tbox  $\mathcal{T}$  and a Rbox  $\mathcal{R}$ , then  $\mathcal{A}$  is consistent w.r.t.  $\mathcal{T}$  and  $\mathcal{R}$  [2]. However, the converse does not hold.

In general, the summary Abox  $\mathcal{A}'$  is dramatically smaller than the original Abox  $\mathcal{A}$ .

## 2.3 Refinement

If the summary Abox is inconsistent, then, in general, we cannot directly conclude anything about the original Abox consistency status.

Our approach [3] for resolving summary Abox inconsistencies is to iteratively *refine* the summary. A refinement step consists of splitting a given summary individual by the sets of role assertions that are present in the original Abox for the Abox individuals mapped to the given summary individual. Refinement increases the size and precision of the summary, and preserves the summary Abox properties (1)-(3) defined in the previous section. Our strategy is to refine only individuals that are part of a summary Abox *justification*, where a justification is a minimal set of assertions which, when taken together, imply a logical contradiction, thus making the entire Abox inconsistent. In some cases, inconsistencies disappear through refinement. Otherwise, when a justification  $\mathcal{J}$  is *precise* (as defined below in Definition 1) we typically know that we have converged on a real inconsistency (see [3] for more details).

**Definition 1.** Let  $\mathcal{A}'$  be a summary Abox of an Abox  $\mathcal{A}$  obtained through the summary mapping  $\mathbf{f}$ . Let  $H$  be a subset of  $\mathcal{A}'$ . We say that an individual  $s \in H$  is *precise* w.r.t.  $H$  iff the following conditions are satisfied:

1. for all individuals  $t \in H$  and for all roles  $R$ ,  $R(s, t) \in H$  (resp.  $R(t, s) \in H$ ) implies that, for all individuals  $a \in \mathcal{A}$  such that  $\mathbf{f}(a) = s$ , there is an individual  $b \in \mathcal{A}$  such that  $\mathbf{f}(b) = t$  and  $R(a, b) \in \mathcal{A}$  (resp.  $R(b, a) \in \mathcal{A}$ ); and
2. for all individuals  $t \in H$ ,  $s \neq t \in H$  (resp.  $t \neq s \in H$ ) implies that, for all individuals  $a \in \mathcal{A}$  such that  $\mathbf{f}(a) = s$ , there is an individual  $b \in \mathcal{A}$  such that  $\mathbf{f}(b) = t$  and  $a \neq b \in \mathcal{A}$  (resp.  $b \neq a \in \mathcal{A}$ ); and
3. There is an individual  $a \in \mathcal{A}$  such that  $\mathbf{f}(a) = s$ ; and
4.  $s : C \in H$  implies that, for all individuals  $a \in \mathcal{A}$  such that  $\mathbf{f}(a) = s$ ,  $a : C \in \mathcal{A}$

We say that  $H$  is precise iff all its individuals are precise w.r.t.  $H$ .

### 3 Summarization and Refinement for Abox Cleanup

In previous work [2] [3], we established that summarization and refinement techniques enable scalable consistency checking and membership query answering over very large and expressive knowledge bases. In this section, we show how these techniques can be adapted to address the issue of detecting and resolving sources of inconsistencies in large knowledge bases such as those generated by text analytic tools.

#### 3.1 The Cleanup Problem

The Abox cleanup problem consists in identifying consistent subsets of an inconsistent Abox. Ideally, it is desirable to identify maximal consistent subsets, i.e. subsets that are consistent but the addition of a single assertion from the inconsistent Abox yields an inconsistency. Unfortunately, computing a single maximal consistency subset is known to be intractable even for realistic small and medium size expressive Aboxes [5].

In our approach, we do not require a cleansed Abox to be a maximal consistent subset. However, each removed assertion must be associated with a unique justification containing it in the original Abox. Since an assertion is removed to avoid the inconsistency created by its associated justification, two distinct assertions must not be associated with the same justification (there is no need to remove more than one assertion from a justification to avoid an inconsistency). Finally, for two justifications  $\mathcal{J}_1$  and  $\mathcal{J}_2$  having assertions  $x$  and  $y$  in common, if the justification associated with  $x$  is  $\mathcal{J}_1$ , then  $\mathcal{J}_2$  cannot be associated with  $y$  (it is clearly not optimal to remove both  $x$  and  $y$  to avoid the inconsistencies due to  $\mathcal{J}_1$  and  $\mathcal{J}_2$ ). When these three conditions are satisfied, we say that the cleansed Abox is a *justification-based consistent subset* as formally defined below:

**Definition 2.** A *justification-based consistent subset*  $CA$  of an inconsistent Abox  $\mathcal{A}$  w.r.t. its Tbox  $\mathcal{T}$  and Rbox  $\mathcal{R}$  is a consistent subset of  $\mathcal{A}$  w.r.t.  $\mathcal{T}$  and  $\mathcal{R}$  such that there is a set  $E$  of justifications of  $\mathcal{A}$  and a bijection  $e$  from  $\mathcal{A} - CA$  to  $E$  such that, for assertions  $x$  and  $y$  in  $\mathcal{A} - CA$ , the following hold: (a)  $x \in e(x)$ , and (b) for  $\mathcal{J}_1$  and  $\mathcal{J}_2$  in  $E$  such that  $\{x, y\} \subseteq \mathcal{J}_1 \cap \mathcal{J}_2$ ,  $e(x) = \mathcal{J}_1$  implies  $e(y) \neq \mathcal{J}_2$ .



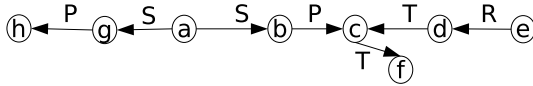


Fig. 1. Simple Abox

A maximal consistent subset is always a justification-based consistent subset, but the converse does not hold if non-disjoint justifications exist. The following example illustrates the differences between maximal consistent subsets and justification-based consistent subsets.

$$\mathcal{R} = \{Range(S) = \forall P.A, Range(P) = \neg A, Range(T) = A, Range(R) = \forall T.\neg A\}$$

Computing a justification-based consistent subset does not require an exhaustive search of justifications. For example, Algorithm 1 provides a straightforward computation of a justification-based consistent subset of an inconsistent Abox. For each inconsistency justification found, one of its assertions is removed from the Abox, and the algorithm continues looking for further justifications and removing axioms from the Abox until it becomes consistent. Likewise, the triple-at-time approach described in our previous work [1] yields a justification-based consistent subset. However, depending on the order in which justifications are found and the choices of removed assertions, a particular execution of these two algorithms might not result in a maximal consistent subset. For instance in the previous example (Figure 1), if an execution of Algorithm 1 first discovers  $J_1$  and  $J_2$  (see below) and chooses to remove  $S(a, b)$  and  $R(e, d)$ , it will at some point find  $J_3$  and be forced to remove either  $P(b, c)$  or  $T(d, c)$ ; i.e. it will not find a maximal consistent subset.

$$\begin{aligned} J_1 &= \{S(a, b), P(b, c)\} & J_2 &= \{R(e, d), T(d, c)\} & J_3 &= \{P(b, c), T(d, c)\} & J_4 &= \\ & \{S(a, g), P(g, h)\} & AllJustifications &= \{J_1, J_2, J_3, J_4\} \\ MA &= \mathcal{A} - \{P(b, c), R(e, d), S(a, g)\} & JCA &= MA - \{S(a, b)\} \\ CA_1 &= JCA - \{T(c, f)\} & CA_2 &= JCA - \{P(g, h)\} \end{aligned}$$

$MA$  is a maximal consistent subset of  $\mathcal{A}$ .  $JCA$  is clearly not maximal (it is a proper subset of  $MA$ , a maximal consistent subset). However,  $JCA$  is a justification-based consistent subset of  $\mathcal{A}$ : the set  $E$  of explanations is the set of all justifications, and  $e$ , defined as  $e(S(a, b)) = J_1$ ,  $e(R(e, d)) = J_2$ ,  $e(P(b, c)) = J_3$ , and  $e(S(a, g)) = J_4$ , satisfies conditions (a) and (b) of Definition 2.  $CA_1$  is not a justification-based consistent subset because  $T(c, f)$  is not present in any justification.  $CA_2$  is not a justification-based consistent subset because  $S(a, g)$  and  $P(g, h)$  have both been removed, but they only appear in a single justification,  $J_4$ .

The most important limitation of the previously described algorithms to compute a justification-based consistent subset is their obvious inability to scale to large and expressive Aboxes containing many inconsistencies, such as those generated from text analytic tools. This is an issue we address in this paper.



```

NaiveJustificationConsistentSubset(Abox  $\mathcal{A}$ , Tbox  $\mathcal{T}$ , Rbox  $\mathcal{R}$ )
begin
   $A \leftarrow \mathcal{A}$ ;
  while inconsistent( $A$ ,  $\mathcal{T}$ ,  $\mathcal{R}$ ) do
    Find a justification  $J$  in  $A$ ;
    select an Abox assertion  $x$  in  $J$ ;
     $A \leftarrow A - \{x\}$ ;
  end
  return  $A$ ;
end

```

**Algorithm 1.** Naive Justification-based Consistent Subset Computation

### 3.2 Scalable Approach to Abox Cleanup

In order to scale the computation of a justification-based consistent subset of an Abox  $\mathcal{A}$ , our approach identifies justifications in the dramatically reduced summary  $\mathcal{A}'$  of  $\mathcal{A}$ . Justifications are then refined until they become *precise* or they disappear from the refined summary. Our Algorithm 2 simulates on  $\mathcal{A}'$  an execution of the naive algorithm 1 applied to  $\mathcal{A}$ .

For an acyclic (considering the undirected graph induced by role and differentFrom assertions) precise justification in the summary  $\mathcal{A}'$ , if it has at least one role assertion, one of its role assertions is removed from the summary; otherwise, one of its concept assertions is removed 1. The accurate image of the summary  $\mathcal{A}'_1$  obtained from removing a role assertion  $R(u, v)$  of a precise acyclic justification is the Abox  $\mathcal{A}_1$  resulting from the removal of all  $R$  role assertions relating images of  $u$  to images of  $v$  from the original Abox  $\mathcal{A}$ . Formally  $\mathcal{A}_1 = \mathcal{A} - \{R(a, b) \in \mathcal{A} | \mathbf{f}(a) = u \text{ and } \mathbf{f}(b) = v\}$ . To show that these assertions removed from  $\mathcal{A}$  can also be removed by an execution of the naive algorithm 1, we need to establish that for each removed assertion  $R(a, b)$  2: there is a justification  $J$  in the Abox such that

- $R(a, b) \in J$ , and
- $J$  does not contain any removed assertion besides  $R(a, b)$ .

**Theorem 1.** *Let  $\mathbf{f}$  be a summary function mapping an Abox  $\mathcal{A}$ , inconsistent w.r.t. to its Tbox  $\mathcal{T}$  and its Rbox  $\mathcal{R}$ , to its summary  $\mathcal{A}'$ . Let  $J'$  be a precise and acyclic justification of  $\mathcal{A}'$ . Let  $R(u, v)$  be a role assertion in  $J'$ . For individuals  $a$  and  $b$  in  $\mathcal{A}$  such that  $\mathbf{f}(a) = u$  and  $\mathbf{f}(b) = v$  and  $R(a, b) \in \mathcal{A}$ , the following conditions hold: there is an inconsistent justification  $J$  of  $\mathcal{A}$  such that:*

1.  $R(a, b) \in J$ , and
2.  $\{R(x, y) \in \mathcal{A} | \mathbf{f}(x) = u \text{ and } \mathbf{f}(y) = v\} \cap J = \{R(a, b)\}$

<sup>1</sup> We remove role assertions instead of concept assertions because concept assertions produced by our text analytic tools have a higher accuracy than role assertions.

<sup>2</sup> We focus on role assertions, but the proof for concept assertion removal is similar.

*Proof.* The main idea behind the proof is that, viewing  $J'$  as a pattern, for each pair of individuals  $(a, b)$  in  $\mathcal{A}$  such that  $\mathbf{f}(a) = u$ ,  $\mathbf{f}(b) = v$ , and  $R(a, b) \in \mathcal{A}$ , there is an instance  $I_{ab}$  of the pattern  $J'$  in  $\mathcal{A}$  such that  $a$  is mapped to  $u$  and  $b$  is mapped to  $v$ . In other words,  $I_{ab}$  is isomorphic<sup>3</sup> to  $J'$ . Since  $J'$  is justification, it follows that  $I_{ab}$  is also a justification. Since  $J'$  is acyclic, it can always be written as a disjoint union of three sets  $J' = J'_u \cup \{R(u, v)\} \cup J'_v$  where  $J'_u$  is the subset of  $J'$  containing the individual  $u$  but not  $v$  whereas  $J'_v$  is the subset of  $J'$  containing  $v$  but not  $u$ . If  $J'_u$  and  $J'_v$  are empty (i.e.  $J'$  consists of the single edge  $R(u, v)$ ), the existence of  $I_{ab}$  is a direct consequence of the summarization process.

We now consider the general case where  $J'_u$  and  $J'_v$  are not empty (if one of them is empty the proof is similar to this general case).  $J'_u$  and  $J'_v$  must be acyclic since there are subsets of an acyclic justification. Lemma [4](#) below establishes the existence of a subset  $I_a$  (resp.  $I_b$ ) of  $\mathcal{A}$  isomorphic to  $J'_u$  (resp.  $J'_v$ ) such that  $a$  (resp.  $b$ ) is mapped to  $u$  (resp.  $v$ ) and each individual  $x$  of  $I_a$  (resp.  $I_b$ ) is mapped to the individual  $\mathbf{f}(x)$  of  $J'_u$  (resp.  $J'_v$ ). It follows that  $I_{ab} \stackrel{def}{=} I_a \cup \{R(a, b)\} \cup I_b$  is a subset of  $\mathcal{A}$  isomorphic to  $J'$  such that, for each individual  $x$  in  $I_{ab}$ ,  $x$  is mapped to the individual  $\mathbf{f}(x)$  of  $J'$ . The isomorphism establishes property (1) of the theorem, namely,  $I_{ab}$  is a justification containing  $R(a, b)$ . Property (2) is a direct consequence of the fact that, for each  $x$  of  $I_{ab}$ ,  $x$  is mapped to the individual  $\mathbf{f}(x)$  of  $J'$ .

**Lemma 1.** *Let  $\mathbf{f}$  be a summary function mapping an Abox  $\mathcal{A}$  to its summary  $\mathcal{A}'$ .*

*If  $L$  is a non-empty acyclic precise subset of  $\mathcal{A}'$ , then, for each individual  $u$  of  $L$  and  $a$  of  $\mathcal{A}$  such that  $\mathbf{f}(a) = u$ , there exists a pair  $(I, \rho)$  such that  $I$  is a subset of  $\mathcal{A}$  and  $\rho$  is a total mapping from  $\text{Indiv}(L)$  to  $\text{Indiv}(I)$ , where  $\text{Indiv}(X)$  denotes the set of individuals in an Abox  $X$ . Furthermore,  $(I, \rho)$  satisfies the following properties for all individuals  $r$  and  $s$  in  $L$ :*

- (a)  $\rho(u) = a$
- (b) if  $R(r, s) \in L$  then  $R(\rho(r), \rho(s)) \in I$
- (c) if  $r \neq s \in L$  then  $\rho(r) \neq \rho(s) \in I$
- (d) if  $s : D \in L$  then  $\rho(s) : D \in I$
- (e)  $\rho(s) = x$  iff.  $\mathbf{f}(x) = s$

*Proof.* See proof of Lemma [4](#) in the technical report [6](#). Note that property (e) is satisfied by the  $\rho$  function presented in the proof.

For a precise cyclic justification  $J'$  in the summary, although, in most cases, it can be directly concluded that it corresponds to a real inconsistency in the Abox [3](#), removing an assertion of  $J'$  from the summary might be too conservative because the accurate image of the removed assertion in  $\mathcal{A}$  might not correspond

<sup>3</sup> Two Aboxes are isomorphic iff. by renaming individuals in one Abox it becomes identical to the other. The renaming must be such that two individuals with different names in the original Abox are not assigned the same name.

to a set of Abox assertions that can be removed by an execution of the naive justification-based consistent subset Algorithm 1.

Consider the following example:

$$\begin{aligned} \mathcal{A} &= \{R(a1, a2), R(a2, a3), R(a3, a1), T(a1, b), a1, a2, a3 : A \sqcap \forall R. \forall R. \neg A\} \\ \mathbf{f}(a1) &= \mathbf{f}(a2) = \mathbf{f}(a3) = u, \mathbf{f}(b) = v \\ \mathcal{A}' &= \{R(u, u), u : A \sqcap \forall R. \forall R. \neg A, T(u, v)\} \quad J' = \mathcal{A}' - \{T(u, v)\} \end{aligned}$$

$J'$  is a precise cyclic justification, and it can directly be shown, based on the application of deterministic tableau rules to the summary (see 3 for more details), that it represents to a real inconsistency in  $\mathcal{A}$ . Removing  $R(u, u)$  creates a consistent summary of a consistent subset of  $\mathcal{A}$ . Unfortunately, this consistent subset is not a justification-based consistent subset. Indeed, Definition 2 is not satisfied since  $R(a1, a2)$  and  $R(a2, a3)$ , have been removed, but they only appear in a single justification,  $J'$ .

To avoid removing more assertions than needed in the Abox, a conclusive precise cyclic justification  $J'$  is further refined until at least one role assertion becomes *super precise*, at which point it is removed.

**Definition 3.** A role assertion  $R(u, v)$  in a summary  $\mathcal{A}'$  of an Abox  $\mathcal{A}$  is *super precise* iff.  $\{R(u, v)\}$  is precise and  $|Image(u)| = |Image(v)| = 1$  (i.e. only one individual in the Abox is mapped to  $u$  or  $v$ ).

Algorithm 2 represents our summarization and refinement based approach to compute a justification-based consistent subset of a large and expressive Abox  $\mathcal{A}$ . It takes as input an inconsistent Abox  $\mathcal{A}$  and its Tbox  $\mathcal{T}$  and its Rbox  $\mathcal{R}$ . It returns a triple consisting of (1) a justification-based consistent subset of  $\mathcal{A}$ , (2) a summary of (1), and (3) a map associating a justification  $J'$  found in a summary to its assertion that was selected for removal.

**Theorem 2.** Algorithm 2 computes a justification-based consistent subset of the input Abox  $\mathcal{A}$ .

*Proof.* After each assertion removal in the summary  $\mathcal{A}'$ ,  $\mathcal{A}'$  is transformed to a summary  $\mathcal{A}''$ . As a direct consequence of Theorem 1 and the definition of super precise role assertion,  $\mathcal{A}''$  is a summary of an Abox which can be created from  $AccImage(\mathcal{A}')$  after several iterations of the naive algorithm 1 applied to  $AccImage(\mathcal{A}')$ . It follows that the final consistent summary  $\mathcal{A}'_f$  returned by Algorithm 2 is such that  $AccImage(\mathcal{A}'_f)$  is a justification-based consistent subset of  $\mathcal{A}$ .

### 3.3 Approximate Cleanup

Computing a justification-based consistent subset of an Abox  $\mathcal{A}$  using Algorithm 2 can be expensive if the summary  $\mathcal{A}'$  has precise cyclic justifications whose accurate images in  $\mathcal{A}$  are very large. The following example illustrates a worst case situation:  $\mathcal{A} = \{R(a_1, a_2), R(a_2, a_3), \dots, R(a_n, a_1), a_1 : A \sqcap \forall R. \neg A\}$

```

JustificationBasedConsistentSubset(Abox  $\mathcal{A}$ , Tbox  $\mathcal{T}$ , Rbox  $\mathcal{R}$ )
begin
   $\mathcal{A}' \leftarrow$  compute the summary Abox of  $\mathcal{A}$ ;
  Results  $\leftarrow \emptyset$ ;
  while inconsistent( $\mathcal{A}'$ ,  $\mathcal{T}$ ,  $\mathcal{R}$ ) do
    Find Justifications in  $\mathcal{A}'$ ;
     $ACJ \leftarrow$  select precise acyclic justifications from Justifications;
     $CJ \leftarrow$  select precise conclusive cyclic justifications from Justifications
    that have at least one super precise role assertion;
    Results  $\leftarrow$  Results  $\cup$  removeAssertion( $\mathcal{A}'$ ,  $ACJ$ ) ;
    Results  $\leftarrow$  Results  $\cup$  removeAssertionInCyclicJ( $\mathcal{A}'$ ,  $CJ$ ) ;
    Justifications  $\leftarrow$  Justifications  $- (ACJ \cup CJ)$  ;
    Execute refinement on  $\mathcal{A}'$  using Justifications ;
  end
  return (AccImage( $\mathcal{A}'$ ),  $\mathcal{A}'$ , Results) ;
end
removeAssertion(SummaryAbox  $\mathcal{A}'$ , SetOfJustifications JS)
begin
  Results  $\leftarrow \emptyset$ ;
  for  $J$  in JS do
    if hasRoleAssertions( $J$ ) then
      Assertion  $\leftarrow$  select a role assertion from  $J$ ;
    else
      Assertion  $\leftarrow$  select a concept assertion from  $J$ ;
    end
    Remove Assertion from  $\mathcal{A}'$ ;
    Results  $\leftarrow$  Results  $\cup (J, \textit{Assertion})$ ;
  end
  return Results;
end
removeAssertionInCyclicJ(SummaryAbox  $\mathcal{A}'$ , SetOfJustifications JS)
begin
  Results  $\leftarrow \emptyset$ ;
  for  $J$  in JS do
    Assertion  $\leftarrow$  select a super precise role assertion from  $J$ ;
    Remove Assertion from  $\mathcal{A}'$ ;
    Results  $\leftarrow$  Results  $\cup (J, \textit{Assertion})$ ;
  end
  return Results;
end

```

**Algorithm 2.** Summarization and Refinement based Justification-based Consistent Subset Computation

$\mathcal{R} = \{Trans(R)\}$  (i.e.  $R$  is transitive)  $\mathcal{A}' = J' = \{R(u, u), u : A \sqcap \forall R. \neg A\}$   
 $J'$  is a conclusive precise cyclic justification (it is conclusive based on the application of deterministic tableau expansion rules on  $\mathcal{A}'$  -see [3] for more details). However, the only precise justification with a super precise role assertion derived from  $J'$  through iterative refinement is the whole Abox  $\mathcal{A}$  (the length  $n$  of the

cycle in  $\mathcal{A}$  could be very significant)! However, in our experimental evaluation, we have not witnessed such an extreme situation. In general, conclusive precise cyclic justifications have fairly small accurate images consisting at most of a few dozen individuals.

An alternative strategy is to not require conclusive precise cyclic justifications in the summary to have a super precise role assertion which can then be removed. We simply remove a selected role assertion from the justification. In doing so, Algorithm 3 always produces a consistent subset of the input Abox  $\mathcal{A}$ , but this subset is no longer guaranteed to be a justification-based consistent subset. However, when an role assertion  $R(u, v)$  is removed from a conclusive precise cyclic justification in the summary, an upper bound of the number of corresponding extraneous Abox assertions removed is given by  $|AccImage(\{R(u, v)\})| - 1$ . As shown in the experimental evaluation section, in practice, the upper bound of the total number of extraneous removed assertions is a small fraction of the total number of removed assertions. Thus, this approximation is quite precise in practice.

```

ApproximateJustificationConsistentSubset(Abox  $\mathcal{A}$ , Tbox  $\mathcal{T}$ , Rbox  $\mathcal{R}$ )
begin
   $\mathcal{A}' \leftarrow$  compute the summary Abox of  $\mathcal{A}$ ;
   $Results \leftarrow \emptyset$ ;
  while inconsistent( $\mathcal{A}', \mathcal{T}, \mathcal{R}$ ) do
    Find Justifications in  $\mathcal{A}'$ ;
     $ACJ \leftarrow$  select precise acyclic justification from Justifications;
     $CJ \leftarrow$  select precise conclusive cyclic justifications from Justifications;
     $Results \leftarrow Results \cup$  removeAssertion( $\mathcal{A}', ACJ \cup CJ$ ) ;
    Justifications  $\leftarrow$  Justifications - ( $ACJ \cup CJ$ ) ;
    Execute refinement on  $\mathcal{A}'$  using Justifications ;
  end
  return (AccImage( $\mathcal{A}'$ ),  $\mathcal{A}'$ ,  $Results$ ) ;
end

```

**Algorithm 3.** Summarization and Refinement based Approximation of Justification-based Consistent Subset Computation

## 4 Explanation

In this section, we discuss the appropriate presentation of information about justifications computed by algorithms presented in the previous section.

As shown in the experimental evaluation section, large Aboxes produced by text analytic tools may have thousands of inconsistencies involving more than one assertion. A justification in the summary can be viewed as a pattern having one or more instances (or isomorphic justifications) in the actual Abox. A summary justification provides a level of abstraction that represents all Abox justifications isomorphic to it.

Furthermore, in our experiments, we have observed that many acyclic justifications in the summary are isomorphic. Therefore, we can represent a group of isomorphic summary justifications with a single justification, called an abstract summary justification. An abstract summary justification abstracts out the names of nodes in the summary, and represents all Abox justifications that have the same pattern of role and concept assertions. Thus, it can be represented as SPARQL query describing its patterns of role and concept assertions. This SPARQL query can then be used to retrieve the set of triples that are involved in the justifications.

For cyclic justifications in the summary such that all role assertions are super precise, we can also group isomorphic justifications and represent them by a single abstract justification, which can be associated with a SPARQL query. However, if a cyclic justification has some edges that are not super precise, the exact pattern represented by the justification is not known (e.g. if the justification consists of just a single role assertion relating an individual to itself, it definitely represents at least one simple cycle in the Abox, but the length of the cycle is unknown). If the user wants more details on a cyclic justification with role assertions that are not super precise, its accurate image in the Abox will have to be retrieved.

## 5 Computational Experience

Our tests were conducted on a 64 bit AMD dual 1GMHz processor 8G RAM Linux machine, and a maximum heap size of 1G. The datasets were stored in a DB2 database. We tested the performance and scalability properties of our approach (Algorithm 3) with 4 datasets generated from text analysis of 100, 500, 1500 and 3683 documents. Table 1 shows the characteristics of these datasets from the perspective of the number of individuals (I), the number of role assertions (R.A.), the number of summary justifications found (J), the number of abstract summary justifications (A.J.), the number of deleted assertions (D.A.), the computed upper bound of the number of extra triples that would have been removed for these datasets if all of these triples were eliminated (Max. E.D.A.), and the time to find these justifications in minutes. The last column represents the number of overlapping abstract justifications of each dataset with the 500 document dataset.

Dataset	I	R.A.	J	A.J.	D.A.	Max. E.D.A.	Time	AJ Overlap with 500
100	8,628	15,521	191	97	299	19	10	84
500	32,787	62,414	625	203	1,150	89	19	203
1500	104,507	195,206	1,570	360	3,910	359	37	169
3683	286,605	513,522	2,744	561	9,574	967	67	168

As shown in Table 1, the time to find justifications is linear. We compare our results for detecting inconsistencies with a technique described in [1]. For the purposes of this comparison, we focus on the 500 document dataset, because

this dataset was used by the triple-at-a-time technique [1] to validate the use of inconsistency checking for cleansing data extracted from text. When a knowledge base has justifications with multiple role assertions, the two techniques are not likely to agree on the set of inconsistent triples because of different choices of role assertions to remove. However, because a vast majority of justifications in the 500 document dataset have a single role assertion, each of the triples found to be inconsistent by the triple-at-a-time technique are also found to be inconsistent using our technique.

The column labeled Max. E.D.A., which represents an upper bound of the number of extraneous assertions removed by our Algorithm 3, shows that the approximate Algorithm 3 is quite precise: the number of extraneous assertions is always less than 10% of the removed assertions and less than 0.1% of the total number of assertions in the original dataset. Moreover, this upperbound is conservatively computed as indicated in section 3.3. In practice, the real number of extraneous assertions might be much smaller. For example, while the computed conservative upperbound of the number of extraneous assertions for the 500 dataset is 89, the actual number was just 7. We removed all assertions removed by the triple-at-a-time technique plus 7 additional assertions.

Finally, the last column of Table 1 indicates substantial overlap between the justifications found in the different datasets. This suggests that patterns of justifications found in small datasets can directly be searched and removed from much larger datasets before starting Algorithm 2 or 3.

## 6 Sophisticated Cleanup Techniques

The fully-automated cleanup strategy based on Algorithm 2 or 3 is inflexible in that it randomly selects and removes assertions present in justifications. An alternative is to make the process more flexible and interactive by allowing the user to specify only a fragment of the error-causing assertions to be removed.

For this purpose, it would be desirable if the system could assist the user in determining likely assertion candidates for removal, by using some sensible metrics for *ranking* assertions. [5] presents a set of strategies for ranking assertions in justifications, including:

- the number of distinct justifications that an assertion appears in – higher the frequency, lower the assertion rank, since it signifies that the assertion is responsible for producing more errors,
- provenance information about the assertion such as its source – in this case, the accuracy of the text analytic tools generating the assertion would be relevant,
- using a history of previous error patterns to identify suspicious assertions – which in our case amounts to storing abstract justifications from previous cleanup sessions on similar Aboxes.

Given a set of ranked assertions, the user could then choose to remove only the low priority assertions from the Abox. However, in doing so, the resultant Abox

is not guaranteed to be consistent (since the earlier approach only guarantees a consistent Abox if the specific randomly selected assertions in the justifications were removed). Thus, we would need to run Algorithm 2 or 3 over the modified Abox again to obtain a new set of justifications, and repeat this process of ranking and removing erroneous assertions iteratively till the entire Abox was clean.

## 7 Related Work

Recently, there has been a lot of work on repairing inconsistencies in OWL-DL Ontologies. Broadly, this work falls into two categories. The first approach, described in [7], [5], involves identifying a single source of the inconsistency (justification) in the ontology by modifying the internals of the DL tableaux reasoner (a technique known as *tableau tracing*), and then using Reiter’s Hitting Set Tree Algorithm [8] to discover all justifications, in order to arrive at a maximal consistent subset of the ontology. We employ the tableau tracing solution in [5] to derive a single justification. However, given the exponential nature of Reiter’s search, using it to fully repair an ontology containing hundreds or thousands of inconsistency-causing justifications, as is in our case, is clearly not feasible.

The second approach is based on phrasing the problem as a belief revision as done in [9], and then revising the knowledge base to get rid of the inconsistency by *rewriting* the axioms to preserve semantics, e.g., introducing disjunctions. On a similar note, [10] proposes tolerating inconsistent theories and using a non-classical form of inference to derive meaningful results from a consistent sub-theory. These solutions do not fit in within the nature of our application given the abundance of inconsistencies, and the fact that most of the errors are actual noise (not partially correct, rewritable axioms) created by the text analytics tools. Instead, we plan to use metrics for ranking erroneous axioms as suggested in [5] to perform sophisticated cleanup (Section 6), based on factors related to the text analytic process.

The other key difference, though, in terms of related work, is that none of the approaches presented above can scale to very large Aboxes containing millions of assertions. Our Abox summarization and refinement techniques have been shown to scale Abox reasoning in a massive way [3].

To conclude, the solution in this paper is the first of its kind that provides a scalable and efficient way to clean very large Aboxes containing numerous inconsistencies.

## 8 Conclusions

Modern Natural Language Processing techniques are extremely scalable but generate noisy data. In previous work we have introduced the problem of cleaning noisy data using semantic web technologies. While showing an overall improvement in precision, the approach had scalability problems, as well as an order dependency. We have been also investigating techniques for summarization and



refinement of Aboxes for scalability. In this paper we put the two together, and introduce a new technique for justification-based consistent subset finding that identifies patterns of inconsistent data in a scalable way.

We are only in the initial phases of putting together scalable inference with scalable natural language processing, but the results presented here are extremely promising; early experiments show a *linear increase* in processing time as the data increases. Due to the challenges of scaling formal evaluations, we have only presented results that scale to an RDF graph of a few hundred thousand nodes, as (random) subsets of this graph have been verified. In other experiments we have managed to process millions of RDF nodes, and we have every reason to believe this technique will scale at the same rate as our information extraction, while improving precision of relation extraction by 8-15% or more.

## References

1. Welty, C.A., Murdock, J.W.: Towards knowledge acquisition from information extraction. In: Proc. of the fifth International Semantic Web Conference, pp. 709–722 (2006)
2. Fokoue, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K.: The summary abox: Cutting ontologies down to size. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 136–145. Springer, Heidelberg (2006)
3. Dolby, J., Fokoue, A., Kalyanpur, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K.: Scalable semantic retrieval through summarization and refinement. In: AAAI 2007. Proc. of the 22nd Conf. on Artificial Intelligence (2007)
4. Sirin, E., Parsia, B.: Pellet: An owl dl reasoner. In: Description Logics (2004)
5. Kalyanpur, A.: Debugging and Repair of OWL-DL Ontologies. PhD thesis, University of Maryland (2006), <https://drum.umd.edu/dspace/bitstream/1903/3820/1/umi-umd-3665.pdf>
6. Dolby, J., Fokoue, A., Kalyanpur, A., Kershenbaum, L., Ma, E., Schonberg, K., Srinivas: Technical report: Scalable semantic retrieval through summarization and refinement (2007), [http://domino.research.ibm.com/comm/research\\_projects.nsf/pages/iaa.index.html/\\$FILE/techReport2007.pdf](http://domino.research.ibm.com/comm/research_projects.nsf/pages/iaa.index.html/$FILE/techReport2007.pdf)
7. Schlobach, S.: Diagnosing terminologies. In: Proceedings of AAAI 2005, pp. 670–675 (2005)
8. Reiter, R.: A theory of diagnosis from first principles. Artificial Intelligence 32, 57–95 (1987)
9. Meyer, T., Lee, K., Booth, R.: Knowledge integration for description logics. In: AAAI, pp. 645–650 (2005)
10. Huang, Z., van Harmelen, F., ten Teije, A.: Reasoning with inconsistent ontologies. In: IJCAI 2005. Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, August 2005 (2005)

# A Cognitive Support Framework for Ontology Mapping

Sean M. Falconer and Margaret-Anne Storey

University of Victoria, Victoria BC V8W 2Y2, Canada  
{seanf,mstorey}@uvic.ca

**Abstract.** Ontology mapping is the key to data interoperability in the semantic web. This problem has received a lot of research attention, however, the research emphasis has been mostly devoted to automating the mapping process, even though the creation of mappings often involve the user. As industry interest in semantic web technologies grows and the number of widely adopted semantic web applications increases, we must begin to support the user. In this paper, we combine data gathered from background literature, theories of cognitive support and decision making, and an observational case study to propose a theoretical framework for cognitive support in ontology mapping tools. We also describe a tool called COGZ that is based on this framework.

## 1 Introduction

Ontologies have seen increasing use in academia and industry, especially as work on the semantic web grows and evolves. A growing reliance on ontologies brings with it many challenges. One challenge is resolving heterogeneity among domain-related ontologies or *ontology mapping*. This is a critical operation for information exchange on the semantic web. Ontology mapping research is receiving increased attention. Mapping contests exist to compare the quality of ontology matchers [9], a mapping API has been proposed [10], and workshops have been organized to discuss this issue. However, the research emphasis has primarily been on the automation of this process, even though most ontology mapping processes require user involvement.

The heterogeneous data mapping problem is not restricted to ontologies and the semantic web. According to Bernstein *et al.* [2], every database research self-assessment has listed interoperability of heterogeneous data as one of the main research problems. Coping with data heterogeneity is still one of the most time-consuming data management problems. Given that this problem is well-known and extensively studied, why is it so difficult to generate mappings between ontologies or any other data source? In addition to different world views and disparate user needs, there are also issues of language and constraints on available data formats. Languages are known to be *locally ambiguous*, meaning that a sentence may contain ambiguous portions unless considered in the context of the whole sentence. Humans use detailed knowledge about the world, connect sentences and fill in missing parts, and infer what someone means (even if he/she did not actually say it) in order to disambiguate [1]. Also, the data format used (e.g., OWL, RDF, XSD) constrains the expressiveness of the data representation. These issues make it unlikely that we will develop fully automated mapping procedures, consequently we concur that “[a] human must be in the loop.” [2]

Research has largely ignored the issue of user intervention and instead has focused on algorithms to compute candidate mappings. Many research tools provide only file dumps of potential mappings [7] or interfaces that quickly become unmanageable [23]. The responsibility of working through the mass of data computed by these algorithms is left to the user. This can be extremely difficult, requiring tremendous patience and an expert understanding of the ontology domain, terminology, and semantics. Contrary to this research trend, we feel that since the human is critical to the success of the mapping procedure, we must address and emphasize user needs. We believe that we must first try to understand the decision making processes used in the mapping task. By understanding these processes, *cognitive support* can be introduced to the tools to reduce the *cognitive load* experienced by users. We believe that user interfaces that offer more effective cognitive support will provide greater productivity gains than improvements to precision and recall in matching algorithms.

The paper is organized as follows. We begin by discussing ontology mapping tools and research trends in ontology mapping. This is followed by a discussion and analysis of cognitive support and decision making theories and how these relate to ontology mapping. Next, we discuss an observational case study where we observed users performing ontology mappings. The data and analysis from this study is combined with our literature review of mapping tools and cognitive support theories to create a theoretical framework for ontology mapping. The framework describes the process and concepts central to human-guided ontology mapping. Following this, we describe how this framework was used to guide the design of the COGZ ontology mapping tool.

## 2 Mapping Tools

Ontology mapping is a prerequisite for many semantic web applications including instance mediation across web sites, agent communication over the Internet, web service integration, and query and answer rewriting. The quality of these applications depends largely on the underlying mapping. Many tools exist to help compute mappings. FOAM (Framework for Ontology Alignment and Mapping) [6] performs fully or semi-automatic alignment of two or more OWL ontologies. The alignment algorithm uses heuristics to compute similarity between terms and individual entities. The user supplies a parameter file that specifies alignment location, an optional file of pre-known mappings, and algorithm specifications. The FOAM tool saves computed mappings along with a score representing the confidence in the mapping. FOAM asks the user to verify certain mappings and the user can specify in the parameter file the maximum number of questions that should be posed.

Chimaera [19] is a tool that supports ontology merging and diagnosis. The system has a web-based interface where the user interacts with web forms to upload ontologies, select algorithm parameters, and merge similar ontology entities. The merge algorithm produces a candidate list of mappings as matching terms, based on term name similarity, term definitions, possible acronyms and expanded forms, and suffix matching.

COMA++ [5], PROMPT [21], AIViz [17], and OLA [11] provide graphical user interfaces. COMA++ automatically generates mappings between source and target schemas (XML or OWL), and draws lines between matching terms. Users can also define their

own term matches by interacting with the schema trees. Hovering over a potential mapping displays a confidence level about the match as a value between zero and one.

PROMPT, developed by the Stanford Medical Informatics group, was designed as a plugin for the popular ontology editor Protégé. The plugin supports managing multiple ontologies including ontology differencing, extraction, merging, and mapping. The user begins the mapping procedure by specifying a source and target ontology. PROMPT then computes an initial set of candidate mappings based largely on lexical similarity between the ontologies. The user works with this list to verify the recommendations or create custom mappings missed by the algorithm. Once a user verifies a mapping, PROMPT's algorithm uses this to perform analysis based on the graph structure of the ontologies. This usually results in further mapping suggestions and the process is repeated until the user deems the mapping complete. Similarly to PROMPT, AlViz is a plugin for Protégé to do ontology mapping. However, the tool is in an early research phase.

OLA (OWL Lite Alignment) provides automated alignment and an environment for manipulating alignments [11]. OLA supports parsing and visualization of ontologies, automated computing of similarities between entities, manual construction of alignments, visualization of alignments, and comparison of alignments. The mapping algorithm finds matches by analyzing the structural similarity between the ontologies using graph-based similarity techniques. This information is combined with label similarity measures to produce mapping correspondences.

Evaluations of these tools have mostly focused on comparing mappings produced with known mappings. PROMPT is an exception in that the authors performed user evaluation experiments [20]. The experiment evaluated tool-generated mapping suggestions by having several users merge two ontologies. The number of steps required, suggestions followed and not followed, and resulting ontologies were all recorded. Precision and recall was used to evaluate the quality of the suggestions. Similarly, Lambrix and Edberg [16] performed a user evaluation of PROMPT and Chimaera [19] for the specific use case of merging ontologies in bioinformatics. The participants were given a number of tasks to perform, a user manual on paper, and the software's help system for support. They were also instructed to "think aloud" during the experiment while an evaluator took notes. Afterwards, the users completed a questionnaire about their experience. The tools were evaluated with the same precision and recall measurements used in the previously described PROMPT experiment, while the user interfaces were evaluated using the REAL (Relevance, Efficiency, Attitude, and Learnability) [18] approach. Under both criteria, PROMPT outperformed Chimaera, but the participants found learning how to merge ontologies in either tool was equally difficult. The participants found it particularly difficult to perform non-automated procedures in PROMPT, such as creating user-defined merges.

Other than these examples, little research has looked at the user side of mapping. We propose that more comprehensive experiments that focus on how people perform mappings will lead to productivity gains in schema matching [2].

### 3 Cognitive Support and Decision Making

Cognitive support refers to the assistance that tools provide to humans in their thinking and problem solving [30]. We often rely on external artifacts (tools) to support

cognition, e.g. a sticky note can be used as an external memory source - a reminder about a task we need to complete. In software tools, software artifacts (e.g. menus, search, term completion) can be introduced to support the human user's cognition.

The relationship between thinking and artifacts is not new. Humans tend to adapt their environment to the activities they wish to complete [30]. For example, Kirlik observed that short-order cooks develop strategies for using their environment to ease their mental work. They “may organize the placement of meats in order of doneness, [and] may lay out dishes or plates to serve as a temporary external memory of orders to be prepared” [15, pp. 84]. The goal of cognitive support within a software system is to offload some of the user's cognitive processes involved in performing a task to the software. This can reduce the number of items that a user must internally track and process, allowing them to concentrate their expertise on other parts of the task.

There is a tendency to support users by automating tasks. Full automation is total cognitive support, relieving the user of all cognitive responsibility. However, some tasks are too difficult to fully automate, and the user is left to deal with the complexity of the task. Automation sometimes introduces complexity or frustration, e.g., the endless menu options in automated phone systems. Brainbridge observed that automation provides the least assistance when we need it most, as generally, we can only automate rudimentary tasks [3]. This is supported by the previously discussed user evaluation of PROMPT and Chimaera. The participants noted that performing non-automated procedures with PROMPT was difficult. This is also true of other mapping tools, which can only automatically discover the simple mappings. It is left to the user to manually create the rest of the mappings with little or no tool support.

In semi-automatic ontology mapping, the automated procedure helps the user heuristically search for mappings by providing suggested or candidate matches. However, for the tool to be effective, it must also support the user by reducing the complexity of analyzing suggested mappings. There has been a growing realization in the sciences that coping with complexity is central to human decision-making [26]. There are several theories of decision making and thinking process that are relevant to solving ontology mapping problems. For example, research has demonstrated that people often solve problems by selective, heuristic search through large problem spaces and large databases [26]. Experts, such as chess masters, use these techniques to solve complex problems. They cannot analyze all possibilities from one chessboard state so they must prune the search space using heuristics. This type of decision-making process is known as the *heuristic-systematic persuasion model* [28].

Related to heuristic search and contextual cues is *filter theory*, which suggests that we make decisions through a series of selection filters [14]. For example, a doctor may begin by asking a patient about their general symptoms and then narrow the focus of the questions based on which diagnoses match the symptoms. In ontology mapping, this decision model can be supported by an overview of the generated mappings with support for user-driven filtering and searching.

*Perceptual contrast effect* describes the effect that humans often make decisions by comparing and contrasting a decision item with a reference item [24]. In mapping, users can compare an unknown mapping to existing mappings, which act as reference items to help reinforce the decision the user is making. Decision makers actively build

a *confirmation bias*, seeking confirmation that they have made a good decision [27]. Tools must provide interfaces to help with the confirmation process (e.g., identifying the local semantic structure of a term in a mapping, the properties of that term, and other possible mappings for the term).

*Multi-attribute choice* describes the decisions we make when comparing situations/objects with multiple attributes [8]. We tend to compare shared attributes or focus on differences in order to come to a decision. For example, when deciding which computer to purchase, we compare the shared features or attributes of the two machines as well as the differences. In ontology mapping, users may compare shared and unique properties of a class to determine if two class labels represent the same concept.

Each of these theories of decision making contributes to our understanding of how users make mapping decisions and how tool support can assist in this process. In the next section we describe an observational case study that further investigates the user decision making process during ontology mapping.

## 4 Observational Case Study

### 4.1 Study Setup

We observed users performing mappings with two different tools, COMA++ and PROMPT, which were selected for several reasons. First, they both support user-interaction and a graphical user-interface. However, the tools support this interaction differently. COMA++ computes a full mapping between the ontologies and the user then interacts with the ontology trees to remove invalid mappings and create missing mappings. PROMPT produces a list of candidate mappings that the user verifies by completing the suggested mapping or removing the operation. This feedback is used by PROMPT to make further suggestions. Moreover, the user-interfaces for both tools are quite different, allowing us to investigate which type of interface better supports a user's mental model.

Four participants,  $P_1$ ,  $P_2$ ,  $P_3$ , and  $P_4$ , were involved in the study.  $P_1$  and  $P_2$  are graduate students in computer science, while  $P_3$  is an ocean sciences graduate student with a physics background, and  $P_4$  has a computer science background and works as a programmer. None of the users had used the tools or performed mappings prior to the study.  $P_1$  and  $P_2$  were placed in the first team,  $T_1$ , while  $P_3$  and  $P_4$  were on the second team,  $T_2$ . The sessions were video recorded, teams were told to “think aloud”, and the generated mappings were saved for later analysis. A team approach was used to encourage discussion. Two university-related ontologies were selected for the experiment, one from the University of Maryland (UMD) and the other from Carnegie Mellon University (CMU). The ontologies cover a domain that should be familiar to all participants and are small enough (UMD has approximately 135 concepts, CMU has approximately 54 concepts) to be explored during the short duration of the experiment.

### 4.2 Analysis

There was a large difference in the users' satisfaction with the tools.  $T_1$  felt that by far, PROMPT was the more useful tool. They had a lot of difficulty making sense of the



mapping lines drawn in COMA++ and *T1* ignored the mapping suggestions after using the tool for seven minutes. Productivity greatly improved once they ignored the suggestions. *T1* started to rely on remembering what they had mapped before in PROMPT and also their knowledge of the ontology's terms. The participants also highlighted context switching issues with COMA++. They found it difficult to tell what had been mapped and what was left to be verified or mapped. *P2* even stated, "How do we know when we're done?" *T1* stated during an interview that they felt two people were necessary to use COMA++ effectively because it forced them to remember so much information: where they were in the ontologies, what had been mapped, etc. For example, they mapped one term twice, first correctly, and then later incorrectly. The teams also tended to revisit mapped terms, having forgotten that they had already inspected them.

Conversely, *T2* primarily felt that COMA++ was the more effective tool. *P3* stated, "COMA++ was easy, was straight-forward, was obvious. The Protégé [PROMPT] tool was irritatingly complex." *P4* agreed that PROMPT had a complex interface, but he did not feel that either tool was necessarily better. He stated that COMA++ was simpler, but difficult to use when there were a lot of candidate mapping lines. He did however feel more confident about the mappings he produced using COMA++. PROMPT gave more information for validating a mapping, but that also complicated the process. There could be several contributing factors for this difference in opinion. The order in which the tools were used may have influenced expectations.

From our analysis, we observed that all participants followed a similar decision making process when judging potential mappings. They relied on concept name similarity from either the suggested candidate mappings or the ontology trees as an indicator of a possible alignment. Next, they used both the internal and external structure of the concepts for validation. If the concepts had similar structure (i.e. context), they felt confident that the mapping was valid. *T2* also highlighted that they relied on their domain knowledge of how a university functions to make decisions. These observations directly correspond to some of the decision making theories previously discussed. Exact matches allow the users to quickly filter the mapping suggestions, as in filter theory. Also, users rely on the internal structure of the mapping terms to compare shared and unshared attributes to infer intended meaning. Domain expertise is used (as in the heuristic-systematic persuasion model) to search for appropriate mappings and also contributes to confirmation bias when inspecting a mapping.

The ability to search and filter mappings and ontology data surfaced during the mapping session and interviews. PROMPT supports searching, but this did not work as the participants expected. *T1* mentioned searching repeatedly, especially while using COMA++, which does not have any search facilities. Advanced searching and filtering (e.g., fuzzy searches) may also be needed, because ontology elements may use abbreviations, prefixes, suffixes, and different word orders. Searching is a way for users to explicitly explore a user-driven mapping or to reduce the ontology's complexity.

In PROMPT, both teams relied on the list of candidate mappings for navigation, while in COMA++, the teams relied on the tree structure of the ontologies. With COMA++, the only navigational device is the ontology trees. When participants mapped two concepts they were often able to quickly perform several additional mappings. We believe this is because once they found a mapping that they were sure about, they inferred other

mappings of parent and child concepts from the ontology trees. However, in PROMPT, they primarily focused on the candidate list, and mostly ignored the ontology trees. Due to this difference in navigation strategy, we believe that COMA++ may better facilitate learning of the ontologies because the user must browse the trees to perform mappings.

Most of the performed mappings were perceived by the participants as simple or “easy” mappings. However, during the study sessions, both teams were forced to ignore some potential mappings when they could not determine if the mapping was correct or could not agree on a decision. This is an interesting result, because both tools do not support a mechanism for returning to a decision point. It is left to the user to remember to come back and inspect a mapping that they initially ignored.

Both teams emphasized the need to determine what has been mapped and what is left to map. *T1* stated that PROMPT supported this better, as it places an “m” icon beside mapped concepts. However, they found it difficult to get a sense for how much they had accomplished and to understand how much was left to complete. Similarly, *T2* felt that COMA++ needed to visualize the difference between unverified and verified mappings.

The teams both liked that PROMPT supplied a reason for suggesting a mapping, although sometimes this reason led to confusion and indecision (e.g. “Meeting” potentially mapping to “Thing”, as a result of the “ing” suffix). *T2* did not feel the confidence value provided by COMA++ (a number between zero and one) was particularly useful. How a tool communicates its candidate mappings relates to how much the user trusts the suggestions.

In summary, the main user concerns seem to stem from the usability of the tool and the cognitive support it offers for manual tasks, rather than the automated mapping generation mechanisms. The main concerns raised were:

- Where should my starting point be for mapping ontologies?
- How do I know when the mapping procedure is complete?
- How can I verify the quality of my mapping?
- How can I identify the most similar areas of the ontologies?
- How can I limit the scope of the mapping?
- How do I flag or indicate a questionable or subjective mapping?
- How can I make temporary decisions and reverse decisions about mappings?

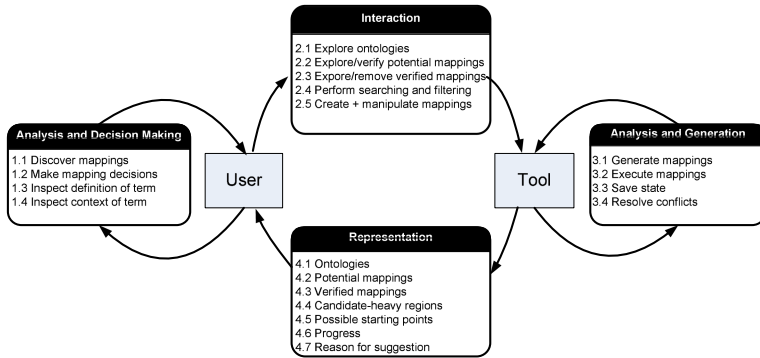
In the following section, the cognitive support framework we propose addresses these user issues within the context of the mapping algorithm support.

## 5 Cognitive Support Framework

In [12], we proposed preliminary cognitive support requirements for ontology mapping tools. Since then, we used data from the observational study and further research into cognitive psychology to develop a theoretical framework describing mapping concepts relating to cognitive support. The framework is shown in Fig. 1 and discussed below.

The framework has four conceptual dimensions: *User Analysis and Decision Making*, *Interaction*, *Analysis and Generation*, and *Representation*, which are based in part on work from [4, pp.7] and [29]. Each dimension represents a concept in the human-guided ontology mapping process. Users internally perform analysis and decision making to understand and validate mappings. Externally, they interact with the tool to acquire information or create mappings. The tool internally performs analysis and generates mappings.





**Fig. 1.** A theoretical framework for cognitive support in ontology mapping

and externally presents these to the user. Distributed cognition between user and artifact (tool) makes the task manageable. The framework dimensions are described below and corresponding software tool requirements (**REQ**) are described for each framework principle (**FP**).

### Analysis and Decision Making

---

#### (#1.1) Discover mappings:

**FP:** Users discover mappings based on their domain knowledge or by exploring the ontologies. This information is often internalized until the user is convinced of the mapping.

**REQ:** Support ontology exploration and manual creation of mappings. Provide tooling for the creation of temporary mappings that the user can address at a later time.

#### (#1.2) Make mapping decisions:

**FP:** Users internally make mapping decisions. The tool aids this by suggesting potential mappings that the user validates.

**REQ:** Provide a method for the user to accept/reject a suggested mapping.

#### (#1.3) Inspect definition of term:

**FP:** The definition of a term comes from the properties that describe the internal structure of the term. The internal structure helps explain the meaning of the term, which facilitates the user's understanding of the ontology.

**REQ:** Provide access to full definitions of ontology terms.

#### (#1.4) Inspect context of term:

**FP:** Context is how a term is used in an ontology. This is derived from the external structure (the *is\_a* hierarchy) and the internal structure (definition of the term). Context of terms in a mapping help the user verify that the intended meaning of terms are the same.

**REQ:** Show the context of a term when a user is inspecting a suggested mapping.

### Interaction Dimension

---

#### (#2.1) Explore ontologies:

**FP:** User-driven navigation of terms, properties, and relationships in the ontologies enforces understanding of the ontology and discovery of mappings.

**REQ:** Provide interactive access to source and target ontologies.

#### (#2.2) Explore/verify potential mappings:

**FP:** Exploring potential mappings aids the user in the verification process.

**REQ:** Support interactive navigation and allow the user to accept/reject potential mappings.

#### (#2.3) Explore/remove verified mappings:

**FP:** Navigation of the verified mappings allows the user to explore what they have completed and what is left to complete.

**REQ:** Support interactive navigation and removal of verified mappings.

**(#2.4) Perform search and filter:**

**FP:** Search and filter facilitates the reduction of information overload for mappings. It also facilitates planning as they allow the user to focus on smaller chunks of the mapping process.

**REQ:** Provide support for searching and filtering the ontologies and mappings (e.g. filters to display terms in the ontologies with/without mappings, or display only the mappings with exact name matches.)

**(#2.5) Direct creation and manipulation of the mappings:**

**FP:** Many mappings are missed by automated procedures, requiring the user to manually create them. Manipulation refers to adding metadata to a verified mapping, such as a reason for the mapping.

**REQ:** Support for adding details on verified mappings and manually create mappings.

## Analysis and Generation

---

**(#3.1) Generate mappings:**

**FP:** Automatic generation of mappings helps users identify simple mappings.

**REQ:** Support the automatic discovery of some mappings.

**(#3.2) Execute mappings:**

**FP:** Executing mappings is the process of transforming instances from one ontology to another based on the available mappings. This can be treated as a *debugging* step in creating a complete mapping: the user can verify if the instances created in the target from the source instances are the ones that (s)he expected.

**REQ:** Allow the user to test mappings by automatically transforming instances from the source to the target ontology.

**(#3.3) Save verification state:**

**FP:** Automatically saving the mapping state and returning to that state with each session relieves the user's working memory from determining where they were, what they were doing, and what their next step is, after an interruption.

**REQ:** The verification process must support potential interruptions by automatically saving and returning users to a given state.

**(#3.4) Conflict resolution and inconsistency detection:**

**FP:** Conflict resolution helps users determine inconsistencies in the created mappings. They can arise from a variety of situations, such as when two concepts are mapped, but some structural elements that are critical for their definition have not been mapped yet.

**REQ:** Support identification and guidance for resolving conflicts.

## Representation Dimension

---

**(#4.1) Source and target ontologies:**

**FP:** Representation of the ontologies facilitates understanding and discovery.

**REQ:** Provide a visual representation of the source and target ontology.

**(#4.2) Potential mappings:**

**FP:** Representation of a potential mapping aids the discovery and decision making process.

**REQ:** Provide a representation of a potential mapping describing why it was suggested, where the terms are in the ontologies, and their context.

**(#4.3) Verified mappings:**

**FP:** Representation of verified mappings frees a user's working memory from remembering what they have already verified.

**REQ:** Provide a representation of the verified mappings that describe why the mapping was accepted, where the terms are in the ontologies, and their context.

**(#4.4) Identify “candidate-heavy” regions:**

**FP:** Identification of candidate-heavy regions aids the planning procedure for performing mappings. It also facilitates understanding of results from the automated procedure.

**REQ:** Identify visually candidate-heavy regions based on the automated mapping procedure.

**(#4.5) Identify possible starting points:**

**FP:** A starting point represents an area of the ontologies or potential mappings where the user may wish to first concentrate their mapping effort.

**REQ:** Indicate possible start points for the user, e.g. flag terms that have exact name matches, as these are generally the most straight-forward mappings to perform.

**(#4.6) Progress feedback:**

**FP:** Progress feedback facilitates planning, as it provides details about where the user is in the overall mapping process. This is also an indicator about the current verification state.

**REQ:** Provide progress feedback on the overall mapping process.

**(#4.7) Reason for suggesting a mapping:**

**FP:** Mappings auto-generated by the tool can support verification and understanding by “explaining” why the algorithm decided the two terms match. An explanation facility helps the user to decide on a mapping and also builds trust between the algorithm and the user.

**REQ:** Provide feedback explaining how the tool determined a potential mapping.

## 6 Using the Framework to Design a Tool

In this section, we use the derived requirements to design a plugin for cognitive support in ontology mapping. Rather than building a tool from scratch, we decided to extend an existing mapping tool with a plugin for cognitive support. We recognized PROMPT as the best match for our cognitive support tool integration. This is because PROMPT already addresses some of the cognitive support requirements we defined and it is available as an open source tool. By working with the PROMPT developers, we created an extensive plugin architecture that allows researchers to easily plug-in their own algorithms, user interface components, and mapping file formats. Using this plugin architecture, researchers can extend many of PROMPT’s user interface components. These extensions to PROMPT were first discussed at the “Ontology Matching Workshop 2006” [12].

We decompose the mapping process into steps: algorithm for comparison, presentation of mappings, fine-tuning and saving mappings, and execution of mappings. These represent plugin extension points in PROMPT. These extensions allow researchers to move their ideas from prototypes to fully implemented mapping tools, without recreating the entire user interface. These extensions to PROMPT provide the ontology engineering community with a consistent interface for mapping and give users access to a suite of tools and algorithms.

We developed a PROMPT plugin called COGZ (Cognitive Support and Visualization for Human-Guided Mapping Systems). COGZ was first introduced in [12] and contained only a neighborhood graph visualization. The latest version contains cognitive aids to address requirements derived from our framework. Because COGZ works as an extension to PROMPT, it can harness the features of PROMPT and enhance or support them with additional visual components. The plugin architecture also allows any algorithm plugin to indirectly benefit from the cognitive support provided by COGZ.

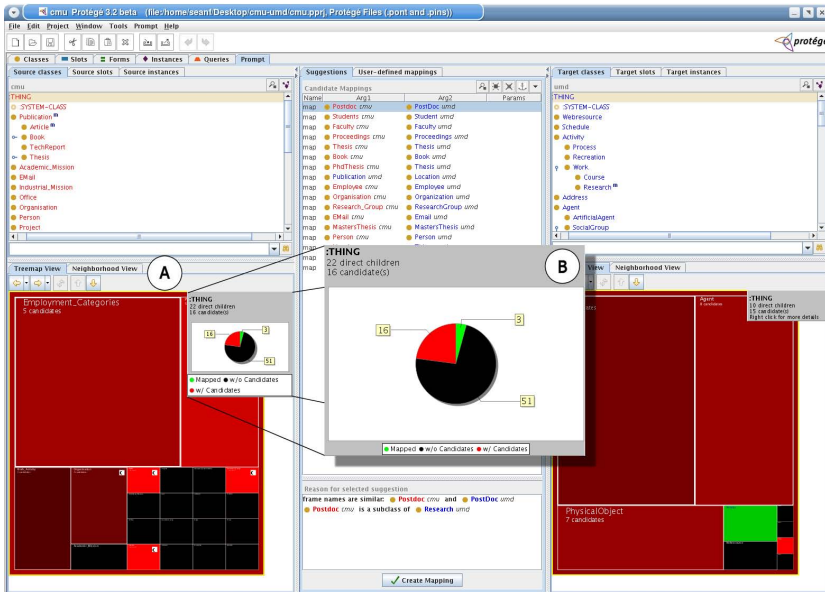


Fig. 2. COGZ TreeMap view (A) with enhanced pie chart view (B)

Figure 2 shows the PROMPT+COGZ tool. TreeMaps [25] are used to provide an overview of the ontology and potential mappings (req. 2.1, 2.2, and 2.3). The TreeMap was chosen for several reasons. The overview needed to fit a small area of the user interface yet display a large amount of data. Since TreeMaps are space-filling, they take up the same amount of screen regardless of ontology size. Also, since ontologies can be very large, we needed a visualization that scales well; TreeMaps can visualize several thousand nodes [13]. Color intensity in the TreeMap helps identify candidate-heavy regions of the ontology and mapped regions (req. 4.4 and 4.3). The pie chart view provides details about the number of candidate mappings, mapped concepts, and concepts without an association within each branch of the ontology. This gives an overview about what has and has not been completed within a branch of the ontology (req. 4.6).

A visualization for comparing term neighborhoods is also available. The neighborhoods represent the “context” of the mapping terms, where the context is defined as the immediate structural relationships of an ontology term (req. 1.4). The generated context provides a visual structural comparison between two candidate terms. The COGZ plugin also provides mapping filters that can be used to reduce the number of mappings shown by PROMPT and allow the user to focus on certain types of mappings (req. 2.4 and 4.5). The filters are based on the categories of potential mappings supported in PROMPT (e.g., exact name matches and synonym matches). Users can also use hierarchical filters to display mappings within certain regions of the ontologies.

To support the user’s working memory, we introduced temporary mappings by extending PROMPT’s candidate mapping list. The user can now flag a mapping as temporary, removing this candidate from the list. Temporary mappings can be viewed within the candidate list either by themselves or with the other candidates. When viewed within

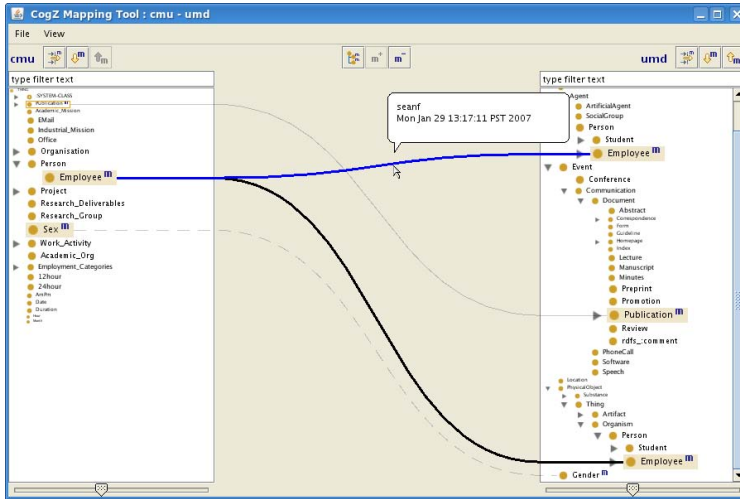


Fig. 3. COGZ view for the current mapping state

the existing candidate list, they are highlighted with a light-blue background. If the user performs a mapping with a concept that already has a temporary mapping, the user is reminded about this and asked if they wish to proceed. If they proceed, the corresponding temporary mappings are removed as possible candidate mappings (req. 1.1 and 1.2).

Figure 3 shows an example of COGZ's complete mapping interface (req. 2.3). Mappings are shown as edges drawn between the ontology trees. The view also displays a mapping annotation that can be used by users to explain why they chose to map two terms (req. 2.5). Temporary mappings are displayed as dashed lines between the source and target terms. The view supports semantic zooming or "fisheye" selection to highlight the current focus. The semantic zooming also effectively displays cases of multiple inheritance, as shown in the figure.

An interactive search is supported where the ontology trees are automatically filtered when a user types in a search query. The text of the query is highlighted in matched nodes. The trees can also be filtered to display only terms with or without mappings. This gives the user a quick overview of the specific mappings they've performed. These advanced filters combined with the semantic zooming help reduce the clutter that can occlude the display when there are a large number of mapping lines (req. 2.4).

While most of the requirements are supported by the PROMPT+COGZ tool suite, there are some limitations. PROMPT's support for executing mappings (req. 3.2), saving verification state (req. 3.3), manual creation of mappings (req. 2.5), and searching (req. 2.4) need to be further extended by COGZ.

## 7 Discussion and Conclusion

The semantic web brings structure and formal semantics to web data. The vision is to create a globally linked database of information, where data can be shared between web pages and local data stores [22]. A prerequisite for information sharing is the mapping

of independent data representations. This procedure is usually carried out offline and relies on the knowledge of domain experts. In this paper, we advocate for cognitive support in ontology mapping tools. Existing research points to a tendency to think of the underlying ontology mapping algorithm as mostly independent from the user. We strongly believe that by embracing a unified view of human and machine, cognitive aids introduced to the mapping process will enhance the quality of mappings.

We introduced a cognitive support framework for mapping tools based on existing literature, theories of cognitive support and decision making, our experience, and an observational case study. The framework describes the relationship between user and tool in the mapping process. From the framework, we see that the automated generation of mappings is a small part of the entire mapping procedure. Moreover, based on our observational study, the problems users experience go beyond the processing of the algorithms. Users have trouble remembering what they have looked at and executed, understanding output from the algorithm, remembering why they performed an operation, reversing their decisions, and gathering evidence to support their decisions. We believe addressing these problems is the key to improving the productivity of the users.

The requirements from this framework were used to develop COGZ, a user-interface plugin for the ontology management suite PROMPT. This tool introduces visualizations to support user cognition, filters to reduce mapping scope, mapping annotations, and the novel cognitive aid of a temporary mapping. To support COGZ as an extension to PROMPT, we enhanced PROMPT by developing a plugin architecture to support algorithm plugins, user-interface plugins, and mapping file extensions. This architecture allows us to harness PROMPT's existing framework and to add mapping algorithm tools to PROMPT. This enables researchers to harness PROMPT's architecture by plugging in their tools and ideas, quickly moving from prototype to full implementation. This approach also makes COGZ completely algorithm independent.

In the future, we plan to carry out a larger usability study to refine the cognitive support framework and enhance the features of COGZ. The PROMPT development team is also enhancing mapping file extension support and other PROMPT features including conflict resolution and mapping verification.

## References

1. Performance, parsing and pragmatics, <http://www.phon.ucl.ac.uk/home/marco/>
2. Bernstein, P.A., Melnik, S.: Model management 2.0: Manipulating richer mappings. In: SIGNMOD 2007 (June 2007)
3. Brainbridge, L.: Ironies of automation. *Automatica* 19, 775–779 (1983)
4. Detienne, F.: *Software Design - Cognitive Aspects*. Springer, Heidelberg
5. H.-H. Do. *Schema Matching and Mapping-based Data Integration*. PhD thesis, Department of Computer Science, Universität Leipzig (2006)
6. Ehrig, M., Sure, Y.: *Ontology mapping - an integrated approach*. In: 1st European Semantic Web Symposium, Heraklion, Greece, Springer, Heidelberg (2004)
7. Ehrig, M., Sure, Y.: FOAM-framework for ontology alignment and mapping; results of the ontology alignment initiative. In: Overmars, M.H. (ed.) *The Design of Dynamic Data Structures*. LNCS, vol. 156, pp. 72–76. Springer, Heidelberg (1983)
8. Einhorn, H.J., Hogarth, R.M.: Behavioral decision theory: Processes of judgement and choice. *Annual Review of Psychology* 32, 53–88 (1981)

9. Euzénat, J.: Eon ontology alignment contest, <http://oaei.inrialpes.fr/2004/Contest/>
10. Euzénat, J.: An API for ontology alignment (version 2.1) (February 2006)
11. Euzénat, J., Loup, D., Touzani, M., Valtchev, P.: Ontology alignment with ola. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, Springer, Heidelberg (2004)
12. Falconer, S.M., Noy, N.F., Storey, M.-A.: Towards understanding the needs of cognitive support for ontology mapping. In: Ontology Matching Workshop (2006)
13. Fekete, J.-D., Plaisant, C.: Interactive information visualization of a million items. In: INFOVIS 2002. Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02), Washington, DC, USA, p. 117. IEEE Computer Society Press, Los Alamitos (2002)
14. Kerckhoff, A.C., Davis, K.E.: Value consensus and need complementarity in mate selection. *American Sociological Review* 27, 295–303 (1962)
15. Kirlik, A.: Global Perspectives on the Ecology of Human-Machine Systems. In: Requirements for psychological models to support design: Toward ecological task analysis, pp. 68–120, ch. 4. Lawrence Erlbaum, Mahwah (1995)
16. Lambrix, P., Edberg, A.: Evaluation of ontology merging tools in bioinformatics. In: Proceedings Pacific Symposium on Biocomputing, Kauai, Hawaii, USA, pp. 589–600 (2003)
17. Lanzenberger, M., Sampson, J.: Alviz - a tool for visual ontology alignment. In: IV 2006: Proceedings of the conference on Information Visualization, Washington, DC, USA, pp. 430–440. IEEE Computer Society Press, Los Alamitos (2006)
18. Löwgren, J.: Human-computer Interaction. What every system developer should know, Studentlitteratur, Lund (1993)
19. McGuinness, D.L., Fikes, R., Rice, J., Wilder, S.: The chimaera ontology environment. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, pp. 1123–1124 (2000)
20. Noy, N.F., Musen, M.A.: Evaluating ontology-mapping tools: Requirements and experience. In: Proceedings of OntoWeb-SIG3 Workshop at the 13th International Conference on Knowledge Engineering and Knowledge Management, Siguenza, Spain, pp. 1–14 (2002)
21. Noy, N.F., Musen, M.A.: The PROMPT suite: Interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies* 59(6), 983–1024 (2003)
22. Palmer, S.B.: The semantic web: An introduction (2001), <http://infomesh.net/2001/swintro/>
23. Robertson, G.G., Czerwinski, M.P., Churchill, J.E.: Visualization of mappings between schemas. In: CHI 2005. Proceedings of the SIGCHI conference on Human factors in computing systems, Portland, Oregon, USA, pp. 431–439. ACM Press, New York, NY, USA (2005)
24. Sherif, M., Taub, D., Hovland, C.: Assimilation and contrast effects of anchoring stimuli on judgements. *Journal of Experimental Psychology* 55, 150–155 (1958)
25. Shneiderman, B.: Tree visualization with tree-maps: 2-d space-filling approach. *ACM Trans. Graph.* 11(1), 92–99 (1992)
26. Simon, H.A., Dantzig, G.B., Hogarth, R., Piott, C.R., Raiffa, H., Schelling, T.C.: Decision making and problem solving. National Academy of Sciences (1986)
27. Snyder, M., Cantor, N.: Testing hypotheses about other people: the use of historical knowledge. *Journal of Experimental Social Psychology* 15, 330–342 (1979)
28. Syque. Theories about decision-making, [http://changingminds.org/explanations/theories/a\\_decision.htm](http://changingminds.org/explanations/theories/a_decision.htm)
29. Victor, B.: Magic ink: Information software and the graphical interface, <http://worrydream.com/MagicInk/>
30. Walenstein, A.: Cognitive support in software engineering tools: A distributed cognition framework. PhD thesis, Simon Fraser University, Vancouver, BC (2002)



# Making More Wikipedians: Facilitating Semantics Reuse for Wikipedia Authoring\*

Linyun Fu, Haofen Wang, Haiping Zhu, Huajie Zhang, Yang Wang, and Yong Yu

Apex Data and Knowledge Management Lab  
Dept. of Computer Science and Engineering, Shanghai Jiao Tong University  
800 Dongchuan Rd, Shanghai, P.R. China, 200240  
{fulinyun, whfcarter, zhu, zhjay, wwwy, yyu}@apex.sjtu.edu.cn

**Abstract.** Wikipedia, a killer application in Web 2.0, has embraced the power of collaborative editing to harness collective intelligence. It can also serve as an ideal Semantic Web data source due to its abundance, influence, high quality and well-structuring. However, the heavy burden of up-building and maintaining such an enormous and ever-growing online encyclopedic knowledge base still rests on a very small group of people. Many casual users may still feel difficulties in writing high quality Wikipedia articles. In this paper, we use RDF graphs to model the key elements in Wikipedia authoring, and propose an integrated solution to make Wikipedia authoring easier based on RDF graph matching, expecting making more Wikipedians. Our solution facilitates semantics reuse and provides users with: 1) a link suggestion module that suggests and auto-completes internal links between Wikipedia articles for the user; 2) a category suggestion module that helps the user place her articles in correct categories. A prototype system is implemented and experimental results show significant improvements over existing solutions to link and category suggestion tasks. The proposed enhancements can be applied to attract more contributors and relieve the burden of professional editors, thus enhancing the current Wikipedia to make it an even better Semantic Web data source.

## 1 Introduction

The past six years have witnessed the tremendously rapid growth of Wikipedia into the largest free encyclopedia that human beings have ever had. Up till now, the advocates have developed 251 languages of Wikipedias<sup>1</sup>, among which the English version<sup>2</sup> is reported to own a prodigious number of more than 1,750,000 articles<sup>3</sup>. The huge impact of Wikipedia has propelled it into the top 20 most popular Web sites on the planet<sup>4</sup>. Moreover, [11] finds that Wikipedia comes close to Encyclopaedia

---

\* This work is funded by IBM China Research Lab.

<sup>1</sup> [http://meta.wikimedia.org/wiki/List\\_of\\_Wikipedias](http://meta.wikimedia.org/wiki/List_of_Wikipedias), accessed on April 27, 2007.

<sup>2</sup> <http://en.wikipedia.org/>

<sup>3</sup> <http://en.wikipedia.org/wiki/Special:Statistics>, accessed on April 27, 2007.

<sup>4</sup> [http://www.alexa.com/data/details/traffic\\_details?y=t&url=Wikipedia.org](http://www.alexa.com/data/details/traffic_details?y=t&url=Wikipedia.org), accessed on April 27, 2007.



Britannica in terms of the accuracy of its science entries, which shows that Wikipedia articles are of high quality and can provide accurate definitions for their topics.

In addition to its abundance, influence and high quality, Wikipedia is well-structured and can serve as an ideal Semantic Web data source. Thus it arouses great interests from the Semantic Web community [2, 9, 1, 4, 10, 3, 7]. All these studies are based on two critical characteristics of Wikipedia, i.e., (*internal*) *links* and *categories*. Links between Wikipedia articles allow the user to access information related to the article she is reading to perform *hyperreading* [15]. These links also point out the associated concepts and can be considered as the carriers of semantic relations. Categories help organize Wikipedia articles in a hierarchical structure. Such a hierarchy, although not so strict [5], is a quite valuable characteristic for a widely accepted Semantic Web data source.

Although links and categories are indispensable for Wikipedia, they may cause much trouble to users who want to make contributions, especially those newcomers. While authoring a Wikipedia article, a casual user often feels at a loss due to lack of knowledge about the existing information accommodated in the system. She may wonder *when it is necessary to provide a link to a related article for readers' reference* and *what categories are proper to characterize an article*. Finding answers to these two kinds of questions distracts the user from authoring the article itself. The user may simply ignore these questions and leave them to professional Wikipedia editors, or even get frustrated and give up making contributions.

In fact, the problem has already been quite serious. [22] shows that 80% articles are contributed by 10% contributors, which means most contributors do not contribute actively. Jimmy Wales, the founder of Wikipedia, even pessimistically declares that the most active 2%, which is 1400 people, did 73.4% of all the edits [12], which implies the heavy burden over professional Wikipedia editors. Properly providing users with information about links and categories at authoring time will greatly relieve the burden of Wikipedia editors and attract many more potential contributors.

In this paper, we propose our solution to the above problem by equipping the current Wikipedia with 1) a **link suggestion** module that seamlessly integrates search and authoring to provide the user with proper inter-article links, and 2) a **category suggestion** module that helps the user to find appropriate categories for her article. The current best practice on link suggestion is prefix matching over titles of Wikipedia articles, and existing document classification approaches are not proper for the category suggestion task due to their poor effectiveness and efficiency when dealing with large-scale category systems [27].

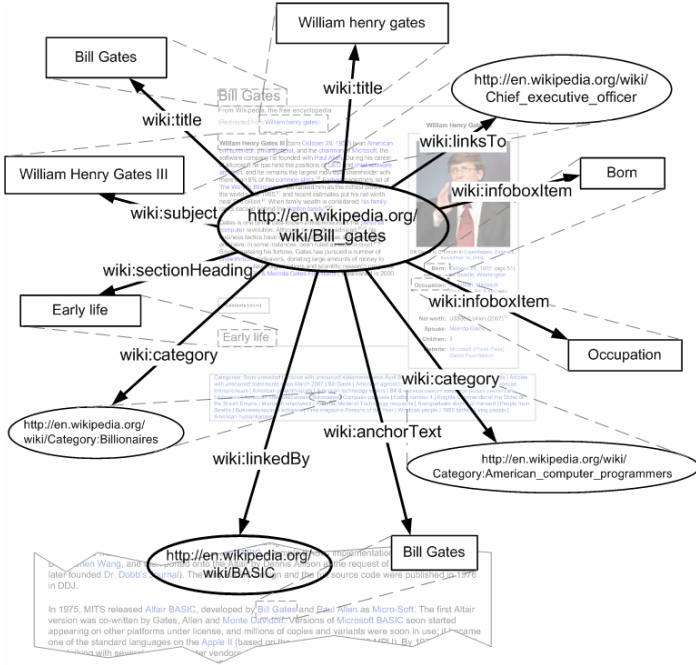
The proposed solution is based on the idea of RDF graph matching since elements in Wikipedia can be represented as resource graphs in RDF [16]. Our method emphasizes making better use of the shallow semantics in Wikipedia. We first use RDF graphs to model resources and queries concerning our tasks, and then show how this model can be applied to find information about links and categories for the user. A prototype system has been implemented and experimental results convincingly validate the effectiveness and efficiency of our solution.

Figure 1 gives two snapshots of our prototype system, namely EachWiki<sup>5</sup>. When a user types some beginning part of a phrase, the link suggestion module will be

---

<sup>5</sup> Accessible at <http://eachwiki.apexlab.org/>





**Fig. 2.** Resource graph for an article in Wikipedia (only part of the property values are displayed)

**Table 1.** Descriptions and usage of Wikipedia article properties

Property Name	Value Description	Usage
title	The title of a Wikipedia article or the title of a redirect page that is redirected to the article	link suggestion
subject	A phrase marked in bold in the first paragraph of the article	link suggestion
anchorText	The displayed text of an link to the article	link suggestion
category	A category of the article	category suggestion
infoboxItem	An item name in the infobox	category suggestion
sectionHeading	The heading for a section of the article	category suggestion
linkedBy	Another Wikipedia article linking to the article	category suggestion
linksTo	Another Wikipedia article to which the article links	category suggestion

We select the properties of an article listed in Table 1 because they carry important semantic features of the topic concerned with the article.

The first three properties, *title*, *subject* and *anchorText*, usually summarize the topic of an article and are used by the link suggestion module. *Titles* usually sum up the topics of articles and most, if not all, of the existing Wikipedia search engines and

authoring assistants, such as LuMriX<sup>6</sup>, WikiWax<sup>7</sup> and Plog4U<sup>8</sup>, support only prefix matching over Wikipedia article titles. In our model, however, titles of redirect pages of an article are treated as its *title* property values as well, and we also select the *subject* and *anchorText* properties to make better use of the shallow semantics in Wikipedia, since it is a Wikipedia convention to mark in bold the names of an article's subject when they are first mentioned in the article, and the displayed text of a link (i.e., the anchor text) usually provides the title or the alternative name of the link target [23]. Expanding the title of an article and taking the other two properties into consideration enable the link suggestion module to find the semantically relevant articles with titles completely different from the query phrase. For example, when the user types “William Henry G”, the article on *Bill Gates* will be found and the user can replace the unfinished phrase with the auto-completed piped link “[*Bill Gates*|William Henry Gates]”, which is beyond the capability of a simple prefix search over article titles.

The next five properties, *category*, *infoboxItem*, *sectionHeading*, *linkedBy* and *linksTo*, outline the structure of an article and are used by the category suggestion module. The basic idea of our category suggestion algorithm is to predict the input article's categories based on the categories of articles structurally similar to it, so *categories* of similar articles are candidate categories for the query article. Articles belonging to the same category usually share many common infobox items and similar section headings, and they often link to and are linked by some common articles. This is why we utilize the *infoboxItem*, *sectionHeading*, *linksTo* and *linkedBy* properties to find similar articles for category suggestion.

The resource graphs of all Wikipedia articles can be connected to form a huge RDF graph for the whole Wikipedia. In this paper, we limit our matching algorithm to the *Concise Bounded Descriptions*<sup>9</sup> (CBD) of articles, as shown in Figure 2, to make our modules efficient while still preserving most essential semantics in Wikipedia articles.

## 2.2 Query Model

In order to find articles most relevant to the input of either link or category suggestion module, we convert each input into an RDF query graph as shown in Figure 3.

A query graph is very much like a resource graph. The differences are as follows:

- The central node of a query graph is always a blank node, indicated by a question mark in Figure 3.
- Some properties in a query graph can be matched for arbitrary times while matching against resource graphs. This kind of properties is defined as *Cloneable Properties* and indicated by an asterisk in Figure 3.

Each of our query graph has only one blank node at the center because the query graph is used to find articles whose CBDs are similar to it; while the introduction of Cloneable Properties makes it possible to convert a phrase query into a query graph and effectively find articles similar to it, as will be shown in Section 3.

<sup>6</sup> <http://wiki.lumrix.net/en/>

<sup>7</sup> <http://www.wikiwax.com/>

<sup>8</sup> [http://www.plog4u.org/index.php/Main\\_Page](http://www.plog4u.org/index.php/Main_Page)

<sup>9</sup> <http://www.w3.org/Submission/CBD/>

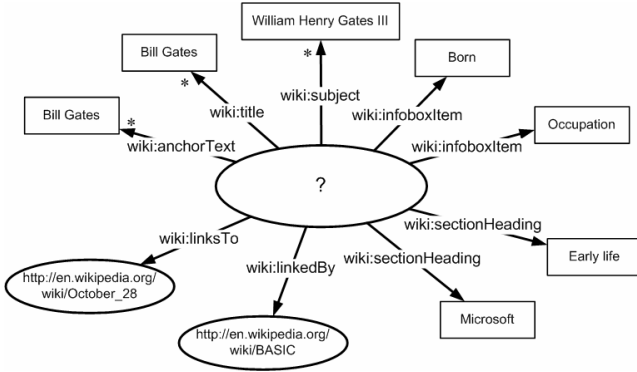


Fig. 3. A sample query graph

### 2.3 Similarity Measurements

Similarity between the query graph and a resource graph is derived from similarity between each query property value and the corresponding resource property value, and similarities between a pair of corresponding property values can be defined according to the need of a specific application. For our tasks, we use the following two kinds of similarities:

- Similarity for Link Suggestion.** In order to let the user get the suggested links as early as possible, we adopt the *Prefix Similarity*, denoted by  $Sim_{pre}(l_q, l_r)$ , for the link suggestion task. Here  $l_q$  and  $l_r$  are literal property values of the query graph and the resource graph, respectively. This similarity presents whether  $l_q$  is a prefix of  $l_r$ , as shown in Equation (1).

$$Sim_{pre}(l_q, l_r) = \begin{cases} 1, & l_q \text{ is prefix of } l_r; \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

The simplicity of Equation (1) makes the link suggestion module efficient enough for authoring time assistance.

- Similarity for Category Suggestion.** In order to reduce noise, each property value indicating structural information of an article is treated as a whole and the similarity for the category suggestion is *Exact Similarity*, denoted by  $Sim_{ex}(v_q, v_r)$ , where  $v_q$  and  $v_r$  are corresponding property values (either literal or resource values) of the query and resource graph, respectively. This similarity is defined in Equation (2).

$$Sim_{ex}(v_q, v_r) = \begin{cases} \log\left(\frac{n}{df+1}\right) + 1, & v_q = v_r; \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

Here  $n$  denotes the total number of articles in the dataset and  $df$  denotes the number of articles having  $v_q$  as the value for the property of concern. This similarity measurement is defined in the same spirit as the Term Frequency/Inverted Document Frequency metric widely accepted by the Information Retrieval (IR) academia and makes the category suggestion module effective.

To specify the importance of each property, it can be assigned a certain *weight*. The similarity between query graph  $q$  and resource graph  $r$ , denoted by *Similarity* ( $q, r$ ), is defined as the weighted sum of the similarities of their corresponding property values, as shown in Equation (3).

$$\text{Similarity}(q, r) = \sum_{p \in P(q) \cap P(r)} w_p \cdot \text{Sim}_p(v_p(q), v_p(r)) \quad (3)$$

Here  $P(g)$  denotes the bag of properties of graph  $g$  and  $v_p(g)$  denotes the value of property  $p$  in  $g$ , and  $w_p$  is the weight for property  $p$  and can be set according to the importance of  $p$  in different applications. The similarity measurement for each property  $p$ ,  $\text{Sim}_p$ , can be set to either  $\text{Sim}_{pre}$  or  $\text{Sim}_{ex}$ . Note that 1) for each Cloneable Property in the query graph, the weighted similarities between the only property value from the query graph and multiple property values from the resource graph are summed up; and 2) if for a given query property that is not Cloneable, there is more than one corresponding resource property value, only the one with the highest similarity is chosen to match the query property value.

## 2.4 Implementation

Despite the simplicity of our model, it is effective enough for the link and category suggestion tasks, as will be validated in Section 4. An additional advantage of our model is that the calculation and top-k ranking of similarity can be accelerated by utilizing an IR engine (Lucene<sup>10</sup> in our experiments) to index the property values for each article in advance. When implementing our prototype system, we index property values in the following way:

- Each article is mapped to a *document*;
- Each value is mapped to a *term*;
- Values of different properties are mapped to terms in different *fields*.

Note that each property value is treated as just one term to support the calculation of the adopted similarity measurements. For example, the article shown in Figure 2 is mapped to a document that has terms “Bill Gates” and “William henry gates” in its “title” field, term “http://en.wikipedia.org/wiki/BASIC” in its “linkedBy” field, etc.

When seeking similar articles with respect to a query, this index can be used to efficiently filter out articles with zero similarity scores, thus accelerating the process.

## 3 Applications of the Model

This section presents how the above model can be applied to the link and category suggestion tasks.

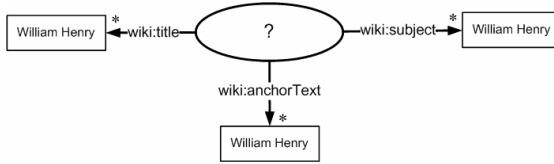
### 3.1 Link Suggestion

The link suggestion module takes the phrase just typed by the user as its input. It first converts the phrase into a query graph; and then it matches all the articles in the

<sup>10</sup> <http://lucene.apache.org/>

dataset against the query graph, getting a similarity score for each article; finally, all the articles are ranked according to their similarity scores and their corresponding wiki links are displayed to the user.

The query graph for a phrase contains three properties: *title*, *subject* and *anchorText*. They are all Cloneable Properties. For each of these properties, its value is set to the input phrase. Figure 4 shows an example of such a query graph.



**Fig. 4.** Query graph for phrase “William Henry”

When matched against resource graphs of articles, all the three properties use the Prefix Similarity measurement. And the weights of these properties are set in reverse proportion to the average numbers of them for a Wikipedia article. The average numbers of values for properties *title*, *subject* and *anchorText* are 1.00, 1.03 and 22.92, respectively, so their weights are set to 23.61, 22.92 and 1.03, respectively.<sup>11</sup>

In order not to let the suggestion list pop up too many times and annoy the user, we set a threshold on similarity score and do not suggest links for stop words.

### 3.2 Category Suggestion

The process of category suggestion is divided into two steps. First, the article that needs categorizing is analyzed, converted into a query graph and matched against by resource graphs of the existing articles to find *k* most similar articles to it; then these *k* articles vote on the categories they belong to and decide the rank of the suggested categories. In our experiment, *k* is set to 200.

The query graph for a newly edited article contains four types of properties: *sectionHeading*, *infoboxItem*, *linkedBy* and *linksTo*, and it may have multiple values for each of them. Note that a newly edited article may have some incoming links because it may be written to fill the target of several broken links<sup>12</sup>. Figure 5 shows a sample query graph for an article.

When matched against resource graphs of existing Wikipedia articles, all the properties in the query graph use the Exact Similarity measurement. And the weights of these properties are set in reverse proportion to the average numbers of them for a Wikipedia article. The average counts of properties *sectionHeading*, *infoboxItem*, *linkedBy* and *linksTo* are 2.40, 1.41, 22.92 and 22.92, respectively, so their weights are set to 740.71, 1260.78, 77.56 and 77.56, respectively.

<sup>11</sup> The average number of values for the *title* property is very close to 1 because only a very small portion of the articles in our dataset have redirect links, and the average number of values for the *subject* property is above 1 because some articles have more than one phrase marked in bold in their first paragraphs.

<sup>12</sup> [http://en.wikipedia.org/wiki/Wikipedia:Most\\_wanted\\_articles](http://en.wikipedia.org/wiki/Wikipedia:Most_wanted_articles).

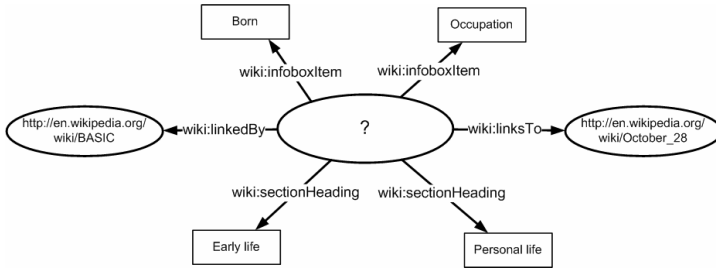


Fig. 5. Query graph for a newly edited article

After getting the similarity score of each existing Wikipedia article, the 200 articles with highest similarity scores are selected to decide the suggested category list by a vote. Each of these 200 articles votes only for the categories it belongs to, with certain amount of votes according to its similarity score. The amount of votes a category gets is shown in Equation (4), where  $Vote(c)$  denotes the amount of votes category  $c$  gets,  $S$  denotes the set of these 200 articles,  $A(c)$  denotes the set of articles that belong to category  $c$ , and  $Similarity(q, a)$  denotes the similarity between query article  $q$  and article  $a$  as defined in Equation (3).

$$Vote(c) = \sum_{a \in S \wedge a \in A(c)} Similarity(q, a) \quad (4)$$

Equation (4) is applied to all the categories that at least one of these 200 articles belongs to. Then these categories are ranked according to their  $Vote$  value and suggested for the user in descending order.

## 4 Experimental Results

### 4.1 Experiment Setup

Our experiments are all done on the WikipediaXML dataset snapshotted in early 2006 [14], which includes 659,353 articles, 113,483 categories and 15,119,272 links. We use the Java programming language to implement the prototype system. The system also utilizes the Dojo javascript toolkit<sup>13</sup> for the user interface and uses the Lucene API to index property values. All the experiments are run on a moderate PC with a 2.4GHz Intel P4 CPU and 2GB memory space.

### 4.2 Link Suggestion Results

To evaluate our link suggestion algorithm, we randomly extracted 576,941 links from the dataset, and use their displayed texts to construct queries. For each of these texts, the first term, the first two terms (if exist) and suchlike parts were fed to the link suggestion module, and the target of this link in the dataset was treated as the standard answer. These same queries were also fed to a prefix search engine over Wikipedia article titles for comparison.

<sup>13</sup> <http://dojotoolkit.org/>



In this way, totally 733,067 queries were constructed and tested. For each query, we recorded the ranks of the standard answer in the suggestion lists generated by the two systems. The results were shown in Table 2.

**Table 2.** Comparison of Link Suggestion and Prefix Matching

#query terms	#queries	Algorithm	ranked 1-5	ranked 6-10	ranked >10
1	576,941	Link Suggestion	<b>83.94%</b>	3.44%	12.62%
		Title Matching	25.30%	9.58%	65.12%
2	126,481	Link Suggestion	<b>90.90%</b>	1.18%	7.92%
		Title Matching	58.93%	4.04%	37.03%
≥3	29,645	Link Suggestion	<b>83.22%</b>	3.20%	13.58%
		Title Matching	51.85%	2.56%	45.59%

From the data shown in Table 2, we could see that for most queries, the correct link targets were ranked among the top 5 suggestions by our link suggestion algorithm, exhibiting significant improvement over the prefix-matching-over-titles algorithm.

The efficiency of our algorithm was also evaluated. The average time cost for a popup was 0.089 seconds.

As mentioned in Section 2.1, one important characteristic of our link suggestion algorithm is that it can find articles semantically related to the query phrase but not containing the phrase as prefix of their titles. Table 3 gives some examples.

**Table 3.** Successful semantic findings by link suggestion

Query	Standard answer	Rank
William Henry Gates	Bill Gates	1
equerry	Master of the Horse	1
cathode	Electrode	4
1977 Election	Manitoba General Election, 1977	1

### 4.3 Category Suggestion Results

To evaluate the category suggestion algorithm, we randomly chose 3,670 existing articles, from the following 3 domains in the WikipediaXML dataset:

- *People* (e.g. Bill Gates, Isaac Newton, Michael Jordan);
- *Location* (e.g. Europe, United Kingdom, London);
- *Jargon* (e.g. Support Vector Machine, Web Ontology Language, PHP, Ontology).

The categories originally associated with these articles served as the ground truth, and the average numbers of categories for an article in the above domains are 3.96, 2.62 and 1.92, respectively. We fed these articles to our category suggestion module as well as a document classification system that treats articles as term weight vectors and measures similarity by the cosine of the angle between document and query vector. The only difference between these two systems is the similarity measurement adopted. The effectiveness of the two systems were measured by using metrics Mean

**Table 4.** Evaluation of category suggestion

Domain	#Query	Algorithm	MAP	P@1	P@2	P@5	R-prec
People	1,030	Category Suggestion	91.1%	88.0%	82.3%	62.3%	90.6%
		Document Classifier	63.6%	60.1%	55.0%	44.1%	61.4%
Location	1,066	Category Suggestion	91.8%	88.3%	78.9%	48.1%	90.3%
		Document Classifier	51.9%	50.1%	35.4%	19.2%	49.9%
Jargon	1,574	Category Suggestion	92.7%	88.3%	72.2%	37.3%	92.3%
		Document Classifier	89.1%	87.0%	63.0%	35.7%	87.3%
Overall	3,670	Category Suggestion	92.0%	88.2%	77.0%	47.5%	91.2%
		Document Classifier	71.1%	68.7%	52.7%	33.3%	69.2%

Average Precision (MAP), Precision at  $n$  ( $P@n$ ) for  $n = 1, 2$  and  $5$ , as well as R-precision (R-prec), which are widely accepted metrics to evaluate retrieval performance. The results are shown in Table 4.

The data shown in Table 4 validate the effectiveness of our algorithm with the improvements over the existing document classification algorithm.

The efficiency of our system and the existing document classifier was also compared. The average time cost of category suggestion for an article was 0.355 seconds, and for the document classifier, the cost was the much longer 129.2 seconds.

Furthermore, scrutiny of the experimental results reveals several interesting findings:

- **Our algorithm is able to suggest *missing categories*.** For example, in the suggestion list for “United Kingdom”, there exists a category named “Category:Island nations”, whose evidences consist of “Republic of Ireland”, “Australia”, etc., but do not contain “United Kingdom” in the WikipediaXML dataset. This article is categorized into “Category:Island nations” (with a synonymous category name “Category:Island countries”) in the current Wikipedia, which implies that our algorithm successfully found an error of *missing category* in the snapshot of Wikipedia in early 2006.
- **The algorithm is capable of discovering improper categorization.** For instance, in the experiment on “Web Ontology Language”, our algorithm failed to suggest the category “Category:XML-based programming languages”, which is a “correct” category for the article according to the ground truth. However, it is not proper to categorize Web Ontology Language (i.e. OWL) as an XML-based programming language. The elimination of this category assignment in the current Wikipedia appears to support our viewpoint.
- **The algorithm can categorize an article to the proper level of abstraction.** Take the article for “Support Vector Machine” as an example, our algorithm suggests both “Category:Machine Learning” and “Category:Artificial Intelligence”, with the former ranked higher than the latter. By consulting the category hierarchy and the standard answer we found that “Category:Machine Learning” is subsumed by “Category:Artificial Intelligence” and the former should be associated with the article, which shows that our algorithm prefers to categorize an article to its immediate categories rather than their ancestors.

## 5 Related Work

As mentioned in Section 1, Wikipedia attracted much attention from the Semantic Web community due to its abundance, influence, high quality and well-structuring. For example, [2] proved that the URIs of Wikipedia articles are surprisingly reliable identifiers for ontology elements; [9] extracted topics and their semantic relations from Wikipedia; [1] presented an experiment to automatically annotate several semantic relationships in Wikipedia; [4] extended Wikipedia with typed links; [10] discovered that links in Wikipedia could provide useful training examples for a named entity disambiguator; [3] presented measures for automatic filtering of strong semantic connections between Wikipedia categories; and [7] provided an extension to be integrated in Wikipedia that allows the typing of links between articles and the specification of typed data inside the articles in an easy-to-use manner.

Our work is mainly motivated by [12] and [13], which pointed out that Wikipedia's authoring interface was not so convenient and smart as had been declared and that Wikipedia only had a relatively regular and small number of community members to take charge of most revision work. We hope to change such a situation.

Research on Wikipedia Modeling and RDF graph matching is closely related to our work. [8] used several features of Wikipedia articles to extract concepts and recognize hierarchical relations between them, and [16] used a specific ontology to integrate Wikipedia into the Semantic Web framework and proposed an RDF graph representation for Wikipedia articles. Our work adapts their models to our tasks by recognizing different properties for Wikipedia articles to better utilize shallow semantics in Wikipedia. [19] introduced a flexible framework for computing semantic similarity between objects represented as RDF graphs. We augment their framework with *Cloneable Properties* for our link suggestion task and define our own similarity measurements between corresponding property values. [21] presented an algorithm for analyzing general logical graphs to compute similarity scores between nodes based on their structural contexts, but the algorithm do not support weighted edges. [20] proposed an approach for semantic search by matching conceptual graphs. We simplify their approach for our real-time tasks. There are other studies related to graph matching in a more general sense. For example, [18] presented a graduated assignment algorithm for graph matching, and [17] presented the Similarity Flooding algorithm for directed labeled graph matching. Their work concentrated on exploring mapping between corresponding nodes of two graphs and do not deal with weighted edges.

Not much work has been done on the task of link suggestion in the Wikipedia environment. [6] addressed the problem of discovering missing links in Wikipedia. Their work concentrated on improving the existing link structure of Wikipedia and was not integrated into the Wikipedia authoring interface. Other related work includes several Wikipedia search engines, which only support prefix search over article titles, as mentioned in Section 2.1.

A straightforward solution to the Wikipedia category suggestion task is to use the vector space model [24, 25] to represent Wikipedia articles as term weight vectors, measure similarity between two articles by the cosine of the angle between their vectors, and suggest categories with the  $k$ -NN method [26]. Our evaluation showed that this solution was far from satisfactory in terms of effectiveness and efficiency.

## 6 Conclusion

In this paper, we use an RDF-based model to represent elements in Wikipedia authoring, and propose two enhancements of the current Wikipedia based on this model to facilitate Wikipedia authoring. Our main contributions can be summarized as follows:

- The proposal of one possible solution for improving Wikipedia authoring interface and reducing the burden of Wikipedia editors, as well as a prototype system based on the proposed enhancements;
- The proposal of a model that extracts semantic information of Wikipedia articles in a succinct way and supports different kinds of queries, along with the recognition of several important semantic properties of Wikipedia articles;
- The proposal of a flexible similarity measurement between query and resource based on the idea of RDF graph matching for effective and efficient accomplishments of the link and category suggestion tasks.

In the future, we will do more user studies and further improve the effectiveness and usability of our system according to feedbacks. We will also suggest other Wikipedia elements such as template for the user and apply our method to Semantic Wikipedia [7] to provide relation and attribute suggestion.

## References

- [1] Ruiz-Casado, M., Alfonseca, E., Castells, P.: From Wikipedia to Semantic Relationships: a semi-automated Annotation Approach. ESWC (2006)
- [2] Hepp, M., Bachlechner, D., Siorpaes, K.: Harvesting Wiki Consensus - Using Wikipedia Entries as Ontology Elements. SemWiki (2006)
- [3] Chernov, S., Iofciu, T., Nejdil, W., Zhou, X.: Extracting Semantic Relationships between Wikipedia Categories (2006)
- [4] Krötzsch, M., Vrandečić, D., Völkel, M.: Wikipedia and the Semantic Web: the Missing Links (2005)
- [5] Voss, J.: Collaborative thesaurus tagging the Wikipedia way. Wikimetrics (2006)
- [6] Adafre, S.F., de Rijke, M.: Discovering Missing Links in Wikipedia. LinkKDD (2005)
- [7] Völkel, M., Krötzsch, M., Vrandečić, D., Haller, H., Studer, R.: Semantic Wikipedia. In: WWW 2006 (2006)
- [8] Kozlova, N.: Automatic Ontology Extraction for document classification. In: Weikum, G. (ed.) A thesis submitted in conformity with the requirements for the degree of Master of Science (2005)
- [9] Kinzler, D.: WikiSense: Mining the Wiki. Wikimania (2005)
- [10] R. Bunescu. Using Encyclopedic Knowledge for Named Entity Disambiguation. EACL 2006 (2006)
- [11] Giles, J.: Internet encyclopaedias go head to head. *Nature* 438(7070), 900–901 (2005)
- [12] Swartz, A.: Raw Thought: Who Writes Wikipedia? (2006), <http://www.aaronsw.com/weblog/whowriteswikipedia>
- [13] Swartz, A.: Raw Thought: Making More Wikipedians (2006), <http://www.aaronsw.com/weblog/morewikipedians>

- [14] Denoyer, L., Gallinari, P.: The Wikipedia XML Corpus. *ACM SIGIR Forum* 40(1) (June 2006)
- [15] Zhang, Y.: Wiki Means More: Hyperreading in Wikipedia. In: *HT 2006* (2006)
- [16] Harth, A., Gassert, H., O'Murchu, I., Breslin, J., Decker, S.: Wikiont: An ontology for describing and exchanging wikipedia articles. *Wikimania 2005* (2005)
- [17] Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching. In: *ICDE 2002* (2002)
- [18] Gold, S., Rangarajan, A.: A graduated assignment algorithm for graph matching. In: *TPAMI 1996* (1996)
- [19] Oldakowski, R., Bizer, C.: SemMF: A Framework for Calculating Semantic Similarity of Objects Represented as RDF Graphs. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005*. LNCS, vol. 3729, Springer, Heidelberg (2005)
- [20] Zhong, J., Zhu, H., Li, J., Yu, Y.: Conceptual Graph Matching for Semantic Search. In: Priss, U., Corbett, D.R., Angelova, G. (eds.) *ICCS 2002*. LNCS (LNAI), vol. 2393, Springer, Heidelberg (2002)
- [21] Jeh, G., Widom, J.: SimRank: a measure of structural-context similarity. In: *KDD 2002* (2002)
- [22] Zlatic, V., Bozicevic, M., Stefancic, H., Domazet, M.: Wikipedias: Collaborative web-based encyclopedias as complex networks. In: *arXiv 2006* (2006), <http://arxiv.org/pdf/physics/0602149>
- [23] Gabrilovich, E., Markovitch, S.: Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. In: *AAAI 2006* (2006)
- [24] Salton, G., Lesk, M.E.: Computer evaluation of indexing and text processing. *Journal of the ACM* 15(1), 8–36
- [25] Salton, G.: *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice-Hall, Englewood Cliffs (1971)
- [26] Mitchell, T.M.: *Machine Learning*. McGraw-Hill, Boston, MA (1997)
- [27] Liu, T-Y., Yang, Y., Wan, H., Zeng, H-J., Chen, Z., Ma, W-Y.: Support Vector Machines Classification with A Very Large-scale Taxonomy. *SIGKDD Explorations* 7(1), 36–43

# CLOnE: Controlled Language for Ontology Editing

Adam Funk<sup>1</sup>, Valentin Tablan<sup>1</sup>, Kalina Bontcheva<sup>1</sup>, Hamish Cunningham<sup>1</sup>,  
Brian Davis<sup>2</sup>, and Siegfried Handschuh<sup>2</sup>

<sup>1</sup> University of Sheffield, UK

<sup>2</sup> Digital Enterprise Research Institute, Galway, Ireland

**Abstract.** This paper presents a controlled language for ontology editing and a software implementation, based partly on standard NLP tools, for processing that language and manipulating an ontology. The input sentences are analysed deterministically and compositionally with respect to a given ontology, which the software consults in order to interpret the input's semantics; this allows the user to learn fewer syntactic structures since some of them can be used to refer to either classes or instances, for example. A repeated-measures, task-based evaluation has been carried out in comparison with a well-known ontology editor; our software received favourable results for basic tasks. The paper also discusses work in progress and future plans for developing this language and tool.

## 1 Introduction

Creating formal data is a high initial barrier for small organisations and individuals wishing to create ontologies and thus benefit from semantic knowledge technologies. Part of the solution comes from ontology authoring tools such as Protégé<sup>1</sup>, but they often require specialist skills in ontology engineering. Therefore defining a controlled language (CL) for formal data description will enable naive users to develop ontologies using a subset of natural language. Building on both controlled language and machine-translation (MT) work, processing input in a controlled language may achieve the high levels of accuracy necessary to make the ontology authoring process more widely feasible. The growing interest in Semantic Web applications and need to translate information into a machine-readable format create many uses for such applications.

The *Controlled Language for Ontology Editing* (CLOnE) allows users to design, create, and manage information spaces without knowledge of complicated standards (such as XML, RDF and OWL) or ontology engineering tools. It is implemented as a simplified natural language processor that allows the specification of logical data for semantic knowledge technology purposes in normal language, but with high accuracy and reliability. The components are based on

---

<sup>1</sup> <http://protege.stanford.edu>

GATE's existing tools for IE (information extraction) and NLP (natural language processing). [1] CLOnE is designed either to accept input as valid (in which case accuracy is generally 100%) or to reject it and warn the user of his errors; because the parsing process is deterministic, the usual IE performance measures (precision and recall) are not relevant.

The prototype was developed and reported in [2] but not evaluated. The CLOnE system has since matured and the main contribution of this work is a *repeated-measures task-based* evaluation in comparison to a standard ontology editor (Protégé). CLOnE has been evaluated favourably by test users and is being further developed and applied in current and future projects. The remainder of this paper is organized as follows: Section 2 briefly discusses existing controlled languages and applications related to the Semantic Web field, Section 3 discusses our design and implementation and Section 4 presents our evaluation and discusses our quantitative findings. Finally, Section 5 and Section 6 offer conclusions and future work.

## 2 Related Work

“Controlled Natural Languages are subsets of natural language whose grammars and dictionaries have been restricted in order to reduce or eliminate both ambiguity and complexity.” [3] The concept was originally proposed in the 1930s by linguists and scholars who sought to establish a “minimal” variety of English, in order to make it accessible and usable internationally for as many people as possible (especially non-native speakers). [3] CLs were later developed specifically for computational treatment. Early CLs include Caterpillar Fundamental English (CFE) [4] and have subsequently evolved into many variations and flavors such as Smart's Plain English Program (PEP) [5], White's International Language for Serving and Maintenance (ILSAM) [5] and Simplified English. [2] They have also found favor in large multi-national corporations such as IBM, Rank, Xerox and Boeing, usually within the context of user-documentation production, machine translation and machine-aided translation. [3,5]

Using CLs for ontology authoring and instance population has already evolved into an active research area. Previous work includes the translation of a CL, PENG-D, into first-order logic (FOL), in order to target the CL to a knowledge representation language such as RDFS or OWL, based on the proposal of FOL as the “semantic underpinning” of the semantic web. [6,7]

A well known implementation of this approach (translating to FOL) is the use of the popular CL, *Attempto Controlled English* [3] (ACE) [8], as an ontology authoring language. It is a subset of standard English designed for knowledge representation and technical specifications, and constrained to be unambiguously machine-readable into discourse representation structures, a form of first-order logic. (It can also be translated into other formal languages.) ACE has

<sup>2</sup> [http://www.simplifiedenglish-aecma.org/Simplified\\_English.htm](http://www.simplifiedenglish-aecma.org/Simplified_English.htm)

<sup>3</sup> <http://www.ifi.unizh.ch/attempto/>

been adopted as the controlled language for the EU FP6 Network of Excellence REVERSE<sup>4</sup> (Reasoning on the Web with Rules and Semantics).<sup>9</sup> The Attempto Parsing Engine (APE) consists principally of a definite clause grammar, augmented with features and inheritance and written in Prolog.<sup>10</sup> This tool can be tested and demonstrated with a web browser through the *APE Webinterface*<sup>5</sup> and clients for the APE web service are also available.<sup>6</sup> REVERSE also proposes ACE OWL, a sublanguage of ACE, as a means of writing formal, simultaneously human- and machine-readable summaries of scientific papers.<sup>11,12</sup>

ACE itself however prohibits certain very natural constructions such as the use of *only* as an adverb. Since ACE OWL also aims to provide reversibility (translating OWL DL into ACE), OWL's *allValuesFrom* must be translated into a construction which can be rather difficult for humans to read. Furthermore, ACE OWL does not currently support enumerations (OWL's *oneOf*) and has limited support for datatype properties; it also imposes several further restrictions on ACE, such as the elimination of plural nouns. ACE itself stipulates a predefined lexicon but unknown words can be used if they are annotated with a POS (part of speech) tag; however this requires the user to be familiar with the lexicon.<sup>13</sup>

*AquaLog*<sup>7</sup> is an ontology-driven, portable question-answering (QA) system designed to provide a natural language query interface to semantic mark-up stored in knowledge base. Like CLONE, it uses JAPE<sup>8</sup> grammars and other tools from the GATE framework<sup>11</sup>. AquaLog uses them to perform shallow parsing on the user's natural-language questions and translate them into a representation called Query-Triples (rather than discourse representation structures or first-order logic).<sup>14</sup> The Relation Similarity Service (RSS) module then maps the questions to ontology-compliant queries and structurally checks the generated query triples against the underlying ontology. If the RSS fails to discover matching relations or concepts within the KB, then it asks the user to disambiguate the relation or concept from a given set of candidates. This module also uses of string similarity metrics, lexical resources such as WordNet<sup>15</sup> and domain-dependent lexica in order to generate query-triples that comply with the underlying ontology.<sup>16</sup> However, existing linguistic rules pose difficulties with respect to complex queries (requiring extension of the NLP component) which the authors plan to remedy in future work.

Another CL implementation for knowledge engineering is the *Cypher*<sup>9</sup> software from Monrai Technologies which translates natural language input into RDF and SeRQL (Sesame RDF Query Language) according to grammars and

<sup>4</sup> <http://reverse.net/>

<sup>5</sup> <http://www.ifi.unizh.ch/attempto/tools/>

<sup>6</sup> [http://www.ifi.unizh.ch/attempto/documentation/appe\\_webservice.html](http://www.ifi.unizh.ch/attempto/documentation/appe_webservice.html)

<sup>7</sup> <http://kmi.open.ac.uk/technologies/aqualog/>

<sup>8</sup> GATE provides the *JAPE* (Java Annotation Pattern Engine) language for matching regular expressions over annotations, adding additional annotations to matched spans, and manipulating the match patterns with Java code.

<sup>9</sup> [http://www.monrai.com/products/cypher/cypher\\_manual](http://www.monrai.com/products/cypher/cypher_manual)



lexica defined by the user in XML. Because Cypher is recently developed and proprietary (but free of charge), no research papers exist to review.

Finally, *GINO* (Guided Input Natural Language Ontology Editor) provides a guided, controlled NLI (natural language interface) for domain-independent ontology editing for the Semantic Web. GINO incrementally parses the input not only to warn the user as soon as possible about errors but also to offer the user (through the GUI) suggested completions of words and sentences—similarly to the “code assist” feature of Eclipse<sup>[10]</sup> and other development environments. GINO translates the completed sentence into triples (for altering the ontology) or SPARQL<sup>[11]</sup> queries and passes them to the Jena Semantic Web framework. (The JENA Eyeball<sup>[12]</sup> model checker verifies the OWL for consistency.) Although the guided interface facilitates input, the sentences are quite verbose and do not allow for aggregation. [17]

### 3 Design and Implementation

Taking into consideration the strengths and weaknesses of other controlled language systems discussed above, we designed the CLOnE software and input language to offer the following advantages.

1. CLOnE requires only one interpreter or runtime environment, the Java 1.5 JRE.
2. CLOnE is a sublanguage of English.
3. As far as possible, CLOnE is grammatically lax; in particular it does not matter whether the input is singular or plural (or even in grammatical agreement).
4. CLOnE can be compact; the user can create any number of classes or instances in one sentence.
5. CLOnE is easy to learn by following examples and a few guiding rules, without having to study formal expressions of syntax; nonetheless, the basic (declarative) implementation of CLOnE uses only 10 syntactic rules.
6. Any valid sentence of CLOnE can be unambiguously parsed.

CLOnE has been favourably evaluated by test users (see Section 4) as part of the SEKT project, and is being further developed for use in other projects (as indicated in Section 6).

Each valid sentence of the controlled language matches exactly one syntactic rule and consists of *keyphrases* (the fixed expressions in the language, which are indexed in a gazetteer, and punctuation marks) and *chunks* (which are similar to noun phrases and are used to name classes, instances, properties and values). A *quoted chunk*, a series of words enclosed in paired quotation marks (“...”), can contain reserved words that would otherwise be detected by the keyphrase

<sup>10</sup> <http://www.eclipse.org/>

<sup>11</sup> <http://www.w3.org/TR/rdf-sparql-query/>

<sup>12</sup> <http://jena.sourceforge.net/Eyeball/full.html>

gazetteer; tokens POS-tagged as determiners and prepositions are also used in some syntactic rules, but otherwise all text that does not match any keyphrases is expected to contribute to a chunk.

Procedurally, CLONe’s analysis consists of the GATE pipeline of processing resources (PRs) shown in Figure 1. This pipeline starts with a series of fairly standard GATE NLP tools which add linguistic annotations and annotation features to the document. These are followed by three PRs developed particularly for CLONe: the gazetteer of keywords and phrases fixed in the controlled language and two JAPE transducers which identify quoted and unquoted chunks. (Names enclosed in pairs of single or double quotation marks can include reserved words, punctuation, prepositions and determiners, which are excluded from unquoted chunks in order to keep the syntax unambiguous.)

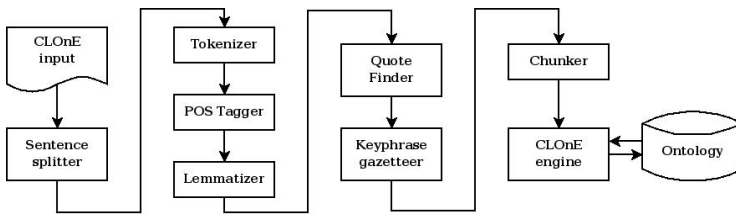


Fig. 1. The CLONe pipeline

The last stage of analysis, the CLONe JAPE transducer, refers to the existing ontology in several ways in order to interpret the input sentences. For example, one syntactic rule has the structure<sup>13</sup>

- (Forget that) CLASSES/INSTANCES are DESCRIPTION PREPOSITION CLASSES/INSTANCES.

which would match the following input sentences.

- Persons are authors of documents.
- Carl Pollard and Ivan Sag are the authors of 'Head-Driven Phrase Structure Grammar'.

This rule can take classes or instances as its arguments. The relevant Java code refers to the ontology and behaves differently according to each argument’s status in the ontology (class, instance, or non-existent). For this rule,

- if the domain and range arguments both refer to classes, the code creates a new property between those classes (e.g. an *Author* property with the domain *Person* and range *Document*);

<sup>13</sup> In the rules, SMALL CAPITALS indicate chunks and chunklists (sequences of chunks separated by commas or the keyword *and*), except that PREPOSITION has the obvious meaning. Most of the rules have an optional *Forget that* prefix that corresponds to “undo” or “delete”.

- if the domain and range arguments both refer to instances and a suitable property definition exists, the code defines a new property value between the instances (e.g. attributing to the instances *Carl\_Pollard* and *Ivan\_Sag* the property of authorship of the document *HSPG*); or
- in other cases (either argument does not exist, or one is a class and the other an instance), the code generates an error message.

Consequently, the same syntactic rule has different semantics according to the context. This means the user only has to learn one syntactic rule and can distinguish its two functions (semantic interpretations) intuitively, as in natural language but with high precision for the software.

The following list explains the other rules of the CLOnE language.

Rule	Example	Action
(Forget that) CLASSES. There is/are CLASSES.	There are agents and documents.	Create new classes (or delete classes).
(Forget that) INSTANCES is a/are CLASS.	"University of Sheffield" is a university. Alice Jones and Bob Smith are persons.	Create (or delete) instances of the class.
(Forget that) CLASSES is/are a type/types of CLASS.	Universities and persons are types of agent. Dogs are a type of mammal. Forget that dogs are a type of cat.	Make subclasses of the last class in the sentence (create new ones if needed). (CLOnE supports multiple inheritance.) The negative form unlinks the the subclass-superclass relationship (but does not delete the subclass).
(Forget that) CLASSES/INSTANCES have CLASSES/INSTANCES.	Journals have articles. "Journal of Knowledge Management" has "Crossing the Chasm".	For lists of classes, create properties of the form <i>Domain_has_Range</i> . For lists of instances, find a suitable property and instantiate it; if there is a class-instance mismatch or a suitable property does not exist, generate an error.
(Forget that) CLASSES have DATATYPE DESCRIPTION.	Projects have string names. Deliverables and conferences have dates as deadlines.	Create datatype properties of the form <i>Domain_has_Description</i> .

Rule		Example	Action
(Forget that) INSTANCE has DESCRIPTION with value VALUE.		SEKT has name with value "Semantically-Enabled Knowledge Technology".	Find a matching datatype property for the instance and instantiate it with the given range value.
(Forget that) INSTANCE is/are also called/known as SYN-ONYMS.	CLASS/	Dogs are also called canines. Bob Smith is also called Bob.	Create synonyms (RDF labels) for the class or instance, which can be used in later CLOnE statements.
Forget everything.		Forget everything.	Clear the whole ontology (and start over).
Forget INSTANCES.	CLASSES/	Forget projects, journals and "Department of Computer Science".	Delete all the classes and instances listed.

## 4 Evaluation

### 4.1 Methodology

We prepared the following documents for the users, which the deliverable [18] includes in full detail.

- The pre-test questionnaire asks each subject how much he thinks he already knows about ontologies, the Semantic Web, Protégé and controlled languages. We scored this questionnaire by assigning each answer a value from 0 to 2 and scaling the total to obtain a score of 0–100.
- The short manual introduces ontologies and provides “quick start” instructions for both pieces of software. Although much simpler, our manual was partly inspired by Protégé’s *Ontology 101* documentation. [19]
- The post-test questionnaire for each tool is the *System Usability Scale* (SUS), which also produces a score of 0–100. [20]
- We devised a comparative questionnaire to measure each user’s preference for one of the two tools. This form is scored similarly to SUS so that 0 would indicate a total preference for Protégé, 100 would indicate a total preference for CLOnE, and 50 would result from marking all the questions *neutral*. On the reverse side and in discussion with the facilitator, we offered each user the opportunity to make comments and suggestions.
- We prepared two similar lists of ontology-editing tasks, each consisting of the following tasks:
  - creating two subclasses of existing classes,
  - creating two instances of different classes, and
  - either (A) creating a property between two classes and defining a property between two instances, or (B) defining two properties between two pairs of instances.

For example, here is task list A:

- Create a subclass *Periodical* of *Document*.
- Create a subclass *Journal* of *Periodical*.
- Create an instance *Crossing the Chasm* of class *Article*.
- Create an instance *Journal of Knowledge Management* of class *Journal*.
- Create a property that agents are publishers of documents.
- Define a property that Hamish Cunningham, Kalina Bontcheva and Yaoyong Li are authors of *Crossing the Chasm*.

We recruited 15 volunteers with varying experience levels and asked each subject to complete the pre-test questionnaire, to read the manual, and to carry out each of the two task lists with one of the two tools. (Approximately half the users carried out task list A with CLOnE and then task list B with Protégé; the others carried out A with Protégé and then B with CLOnE.)

We measured each user’s time for each task list and in most cases (12 of 15) for each sublist. After each task list we asked the user to complete the SUS questionnaire for the specific tool used, and finally we asked him to complete the comparative questionnaire.

## 4.2 Background

Our methodology constitutes a *repeated-measures, task-based* evaluation: each subject carries out a similar list of tasks on both tools being compared.

We chose the SUS questionnaire as our principal measure of software usability because it is a *de facto* standard in this field. Although superficially it seems subjective and its creator called it “quick and dirty”, it was developed properly as a Likert scale. [20] Furthermore, researchers at Fidelity Investments carried out a comparative study of SUS and three other published usability questionnaires as well as an internal questionnaire used at that firm, over a population of 123 subjects, to determine the sample sizes required to obtain consistent, accurate results. They found that SUS produced the most reliable results across all sample sizes; they noted a jump in accuracy to 75% at a sample size of 8, but recommended a sample of at least 12–14 subjects. [21]

## 4.3 Quantitative Findings

Table 2 summarizes the main measures obtained from our evaluation. [14] In particular, CLOnE’s usability scores are generally above the baseline of 65–70% [22] and are generally distributed higher than Protégé’s, and scores on the comparative questionnaire are generally favourable to CLOnE. Table 3 shows confidence intervals for the SUS scores by tool and task list (A, B or combined); these indicate that for each task list and for the combined results, the larger population which our sample represents will also produce mean SUS scores for CLOnE that are both higher than those for Protégé and above the SUS baseline. [15]

<sup>14</sup> The deliverable [18] provides full details of the measurements discussed here.

<sup>15</sup> A data sample’s *95% confidence interval* is a range 95% likely to contain the mean of the whole population that the sample represents. [23]

**Table 2.** Summary of the questionnaire scores

Measure	min	mean	median	max
Pre-test scores	25	55	58	83
CLIE SUS rating	65	78	78	93
Protégé SUS rating	20	47	48	78
CLIE/Protégé preference	43	72	70	93

**Table 3.** Confidence intervals (95%) for the SUS scores

Tool	Confidence intervals		
	Task list A	Task list B	Combined
Protégé	33–65	30–58	37–56
CLIE	75–93	67–79	73–84

We also generated Pearson’s and Spearman’s correlations coefficients<sup>16</sup> for a wide range of data from the experiments; Table 4 shows the highlights of these calculations. In particular, we note the following points.

- The pre-test score has no correlation with task times or SUS results.
- The task times for both tools are moderately correlated with each other but there are no significant correlations between task times and SUS scores, so both tools are technically suitable for carrying out both task lists.
- As expected, the C/P preference score correlates moderately strongly with the CLOnE SUS score and moderately negatively with the Protégé SUS score. (The SUS scores for the two tools also show a weak negative correlation with each other.) These figures confirm the coherence of the questionnaires as a whole.

#### 4.4 Sample Quality

Although our sample size ( $n = 15$ ) satisfies the requirements for reliable SUS evaluations (as discussed in Section 4.2), it is also worthwhile to establish the consistency of the two partitions of our sample, as enumerated in Table 5.

<sup>16</sup> A *correlation coefficient* over a set of pairs of numbers is analogous to a scatter plot: +1 signifies a perfect correlation and corresponds to a graph in which all points lie on a straight line with a positive slope; -1 signifies a perfect inverse correlation (the points lie on a straight line with a negative slope); 0 indicates a complete lack of correlation (random distribution of the points). Values above +0.7 and below -0.7 are generally considered to indicate strong correlations.

Pearson’s formula assumes that the two variables are linearly meaningful (e.g. physical measurements such as temperature), whereas Spearman’s formula stipulates only ordinal significance (ranking) and is often considered more suitable for subjective measurements (such as many in the social sciences). [23][24]

**Table 4.** Correlation coefficients

Measure	Measure	Pearson's	Spearman's	Correlation
Pre-test	CLIE time	-0.06	-0.15	none
Pre-test	Protégé time	-0.13	-0.27	none
Pre-test	CLIE SUS	-0.17	-0.17	none
Pre-test	Protégé SUS	-0.16	-0.15	none
CLIE time	Protégé time	0.78	0.51	+
CLIE time	CLIE SUS	0.26	0.15	none
Protégé time	Protégé SUS	-0.17	-0.24	none
CLIE time	Protégé SUS	0.19	-0.01	none
Protégé time	CLIE SUS	0.42	0.44	weak +
CLIE SUS	Protégé SUS	-0.31	-0.20	none
CLIE SUS	C/P Preference	0.68	0.63	+
Protégé SUS	C/P Preference	-0.62	-0.63	-

**by tool order and task-tool assignment:** subjects who carried out task list A on CLOnE and then B on Protégé, in comparison with those who carried out A on Protégé then B on CLOnE; and

**by sample source:** subjects drawn from the GATE development team, in comparison with others.

Tool order was divided almost evenly among our sample. Although the SUS scores differed slightly according to tool order (as indicated in Table 3), the similar task times suggest that task lists A and B required similar effort. We note that the SUS scores for each tool tend to be slightly lower for task list B than for A, and we suspect this may have resulted from the subjects' waning interest as the evaluation progressed. (To eliminate the possibility of this effect completely with the same reliability, we would need twice as many subjects, each carrying out one task list with one tool—a *between-subjects* experiment, in contrast to our *within-subject* experiment.) But because Table 3 in particular shows consistent results between CLOnE and Protégé for each task list, we conclude that our study was fair.

We must also consider the possibility of biased subjects drawn from colleagues of the developers and facilitator. As Table 5 shows, colleagues in the GATE team constituted 60% of the user sample. The measures summarized in Table 6, however, show in particular that although members of group G generally rated their own expertise higher (in the pre-test questionnaire) than those in group NG, both groups produced very similar ranges of SUS scores for each tool and of C/P preferences scores. These measures allay concerns about biased subjects: we conclude that groups G and NG were equally reliable so the sample as a whole is satisfactory for the SUS evaluation (according to the research [21] explained in Section 4.2).

#### 4.5 Subjects' Suggestions and Comments

The test users made several suggestions about the controlled language and the user interface.

**Table 5.** Groups of subjects by source and tool order

Source	Tool order		Total
	PC	CP	
G GATE team	4	5	9
NG others	4	2	6
Total	8	7	15

**Table 6.** Comparison of the two sources of subjects

Measure	Group	min mean median max			
		Pre-test	G	25	62
	NG	33	44	42	58
CLIE SUS	G	65	78	78	90
	NG	65	78	78	93
Protégé SUS	G	25	46	48	70
	NG	20	47	46	78
C/P Preference	G	50	71	68	90
	NG	43	74	79	93

- Several subjects complained that they needed to spell and type correctly the exact names of the classes and instances, such as “Journal of Knowledge Management” and that CLOnE is intolerant of typos and spelling mistakes. They suggested spell-checking and hinting (as provided in the Eclipse UI) to alleviate this cognitive load.
- A related suggestion is to highlight the input text with different colours for classes, instances and properties, perhaps after the user clicks a *Check* button, which would be provided in addition to the *Run* button. The *Check* button could also suggest corrections and give error messages without affecting the ontology.
- Users complained that it was difficult to tell why some input sentences failed to have any effect, because CLOnE does not explicitly indicate unparseable sentences. (This has already been corrected in subsequent work.)
- Some suggested that CLOnE should automatically clear the input text box after each run, but they also realized this would make it more difficult to correct errors, because it is currently easy to prefix the keyword **forget** to incorrect sentences from the previous input without having to retype them.
- Some suggested making the *Run* button easier to find and putting the ontology viewer and the input box on the screen at the same time instead of in alternative panes.
- A few users said it would be useful to have an *Undo* button that would simply reverse the last input text, instead of having to **forget** all the sentences individually.



## 5 Conclusion

Our user evaluation consistently indicated that our subjects found CLOnE significantly more usable and preferable than Protégé for the straightforward tasks that we assigned. Of course we make no claims about the more complicated knowledge engineering work for which Protégé but not CLOnE is designed and intended. The users considered CLOnE picky about spelling and typing, but they found the language basically easy to learn and use.

Our subjects made several interesting and useful suggestions for improvements to CLOnE, many of which we already envisage developing in future work on this software beyond SEKT, as Section 6 will indicate. In particular, we will embed CLOnE in wikis and perhaps other user interfaces which will eliminate some of the constraints imposed by running it in the GATE GUI, which is really intended for developing language engineering applications (such as CLOnE itself) rather than for interactively editing language resources (such as documents and ontologies). The use of CLOnE in NEPOMUK will especially address these issues.

## 6 Continuing and Future Work

We have discussed using CLOnE to generate ontologies from input text. The reverse of the process involves the generation of CLOnE from an existing ontology by shallow NLG (natural language generation). A prototype component for generating CLOnE has already been implemented as a GATE resource. The textual output of the generator is configured using an XML file, which contains text templates that are instantiated and filled in with ontological values. The NL generator and the authoring process both combine to form a Round-Trip Ontology Authoring environment: one can start with an existing or empty ontology, produce CLOnE using the NL generator, modify or edit the text as requirement and subsequently parse the text back into the ontology using the CLOnE environment. The process can be repeated as necessary until the required result is obtained. Current developments in relation to the NL generator involve extending modifying the XML templates with respect to the generator's linguistic output so that it complies with the grammar and subsequent extensions such as alternatives for expressing the same message. This is essential in order to ensure that the generation component does not interfere with ontological data created or modified by CLOnE. This work is currently being carried out in collaboration with DERI Galway<sup>17</sup> on the integration of CLOnE as a web service for Round Trip Ontology Authoring as a part the Nepomuk<sup>18</sup> Solution (The Social Semantic Desktop). We are also investigating the provision of CLOnE as a local service within one of the promised semantic wikis in addition to the possible use of CLOnE for Semi-Automatic Semantic Annotation. Shallow, template-based

<sup>17</sup> <http://www.deri.ie/>

<sup>18</sup> <http://nepomuk.semanticdesktop.org/xwiki/>

NLG of CLOnE will also be exploited in KnowledgeWeb<sup>19</sup> for the verbalization suggestions for semi-automatic ontology integration.

In the TAO project, CLOnE is currently being extended and embedded in applications to allow controlled-language querying of knowledge bases. This will allow non-expert users (who are not familiar with SeRQL and other sophisticated semantic search tools) to ask questions in an extended version of CLOnE; they are translated by the application into SeRQL or SPARQL and evaluated by OWLIM.

We are also assisting the EPSRC-funded Easy project to use CLOnE to provide a controlled natural language interface for editing IT authorization policies (access to network resources such as directories and printers) stored as ontologies. This project will probably involve specialized extensions to the controlled language as well as a *virtuous circle* or round-trip information flow, to be completed by generation of CLOnE, similar to the work in NEPOMUK. [25](#)

## Acknowledgements

This research has been partially supported by the following grants: KnowledgeWeb (EU Network of Excellence IST-2004-507482), TAO (EU FP6 project IST-2004-026460), SEKT (EU FP6 project IST-IP-2003-506826, L on (Science Foundation Ireland project SFI/02/CE1/1131) and NEPOMUK (EU project FP6-027705).

## References

1. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: ACL 2002. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (2002)
2. Tablan, V., Polajnar, T., Cunningham, H., Bontcheva, K.: User-friendly ontology authoring using a controlled language. In: 5th Language Resources and Evaluation Conference (2006)
3. Schwitter, R.: Controlled natural languages. Technical report, Centre for Language Technology, Macquarie University (2007)
4. Caterpillar Corporation: Dictionary for Caterpillar Fundamental English. Caterpillar Corporation (1974)
5. Adriaens, G., Schreurs, D.: From COGRAM to ALCOGRAM: Toward a controlled English grammar checker. In: COLING 1992. Conference on Computational Linguistics, Nantes, France, pp. 595–601 (1992)
6. Schwitter, R., Tilbrook, M.: Controlled natural language meets the semantic web (2004)
7. Horrocks, I., Patel-Schneider, P.: Three theses of representation in the semantic web (2003)
8. Fuchs, N., Schwitter, R.: Attempto Controlled English (ACE). In: CLAW 1996. Proceedings of the First International Workshop on Controlled Language Applications, Leuven, Belgium (1996)

<sup>19</sup> <http://knowledgeweb.semanticweb.org/>

9. Fuchs, N.E., Kaljurand, K., Kuhn, T., Schneider, G., Royer, L., Schröder, M.: Attempto Controlled English and the semantic web. Deliverable I2D7, REWERSE Project (2006)
10. Hoefler, S.: The syntax of Attempto Controlled English: An abstract grammar for ACE 4.0. Technical Report ifi-2004.03, Department of Informatics, University of Zurich (2004)
11. Kaljurand, K., Fuchs, N.E.: Bidirectional mapping between OWL DL and Attempto Controlled English. In: Fourth Workshop on Principles and Practice of Semantic Web Reasoning, Budva, Montenegro (2006)
12. Kuhn, T.: Attempto Controlled English as ontology language. In: Bry, F., Schwerdtel, U. (eds.) REWERSE Annual Meeting 2006 (2006)
13. Kaljurand, K.: Writing OWL ontologies in ACE. Technical report, University of Zurich (2006)
14. Bernstein, A., Kaufmann, E., Göring, A., Kiefer, C.: Querying Ontologies: A Controlled English Interface for End-users. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 112–126. Springer, Heidelberg (2005)
15. Fellbaum, C. (ed.): WordNet - An Electronic Lexical Database. MIT Press, Cambridge (1998)
16. Lopez, V., Motta, E.: Ontology driven question answering in AquaLog. In: Meziane, F., Métails, E. (eds.) NLDB 2004. LNCS, vol. 3136, Springer, Heidelberg (2004)
17. Bernstein, A., Kaufmann, E.: GINO—a guided input natural language ontology editor. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
18. Funk, A., Davis, B., Tablan, V., Bontcheva, K., Cunningham, H.: Controlled language IE components version 2. Deliverable D2.2.2, SEKT (2006)
19. Noy, N.F., McGuinness, D.L.: Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05, Stanford Knowledge Systems Laboratory (2001)
20. Brooke, J.: SUS: a “quick and dirty” usability scale. In: Jordan, P., Thomas, B., Weerdmeester, B., McClelland, A. (eds.) Usability Evaluation in Industry, Taylor and Francis, London (1996)
21. Tullis, T.S., Stetson, J.N.: A comparison of questionnaires for assessing website usability. In: Usability Professionals’ Association Conference, Minneapolis, Minnesota (2004)
22. Bailey, B.: Getting the complete picture with usability testing. Usability updates newsletter, U.S. Department of Health and Human Services (2006)
23. John, L., Phillips, J.: How to Think about Statistics. W. H. Freeman and Company, New York (1996)
24. Connolly, T.G., Sluckin, W.: An Introduction to Statistics for the Social Sciences, 3rd edn. Macmillan, NYC (1971)
25. Chadwick, D., Sasse, A.: The virtuous circle of expressing authorization policies. In: Semantic Web Policy Workshop, Athens, Georgia (2006)

# Creating a Dead Poets Society: Extracting a Social Network of Historical Persons from the Web

Gijs Geleijnse and Jan Korst

Philips Research  
High Tech Campus 34, 5656 AE Eindhoven, The Netherlands  
{gijs.geleijnse, jan.korst}@philips.com

**Abstract.** We present a simple method to extract information from search engine snippets. Although the techniques presented are domain independent, this work focuses on extracting biographical information of historical persons from multiple unstructured sources on the Web. We first similarly find a list of persons and their periods of life by querying the periods and scanning the retrieved snippets for person names. Subsequently, we find biographical information for the persons extracted. In order to get insight in the mutual relations among the persons identified, we create a social network using co-occurrences on the Web. Although we use uncontrolled and unstructured Web sources, the information extracted is reliable. Moreover we show that Web Information Extraction can be used to create both informative and enjoyable applications.

## 1 Introduction

Information extraction (IE) is the task of identifying instances and relations between those instances in a text corpus. Thanks to popular search engines, the Web is a well-indexed tera-size corpus, where precise queries often lead to highly relevant text fragments. When we assume that all knowledge available on a domain can be found on the Web, this corpus can be used for *ontology-driven information extraction* [1], i.e. we can assume that all required information is available within the web. Due to the redundancy of information on the web, the question is rather how to identify *at least one* formulation of every fact of interest (determined by the ontology), than how to extract *every* relevant formulation within the corpus (i.e. *corpus-driven* information extraction).

Pattern-based approaches have shown to be effective for ontology-driven information extraction. Such approaches can not only be used to identify lexicographical relations (e.g. hyponyms, part-of) [2,3] but also various other relations. Patterns expressing relations (e.g. ‘*is the president of*’, [2]) or combined instance-pattern terms (e.g. *is the president of the United States*, [4,1]) are used as queries to a search engine. Such queries give access to highly relevant text fragments and instances (e.g. *George W. Bush*) and the expressed relations can be extracted simultaneously.

In this paper we show how web information extraction can indeed be a valuable addition to the current collective knowledge on the web. We present a method to populate an ontology using information extraction from search engine query results. Using

the redundancy of information on the web, we can apply simple methods to extract term and relations. We illustrate this approach by populating an ontology with historical persons, their main biographical data and their ‘degree of fame’. We also create a social network among the persons extracted. By combining and structuring information from a large collection of web pages, questions like *Who are the most famous novelists from Ireland?*, *Which people were born in 1648?*, *Who are popular female composers?* and *Who are considered to be most related to Vincent van Gogh?* can now be easily answered. This work illustrates that using simple techniques we can create a reliable ontology that gives insight in the domain and can be a true addition to the information available.

## 2 Related Work

KnowItAll is a hybrid named-entity extraction system [2] that finds lists of instances (e.g. ‘Busan’, ‘Karlsruhe’) of a given class (e.g. ‘City’) from the web using a search engine. It combines hyponym patterns [5] and class-specific, learned patterns to identify and extract named-entities. Moreover, it uses adaptive wrapper algorithms [6] to extract information from HTML markup such as tables. Contrary to the method used in this paper, the instances found are not used to create new queries. In [7] the information extracted by KnowItAll is evaluated using a combinatorial model based on the redundancy of information on the web.

Recently, the KnowItAll project has addressed the identification of complex named entities (such as book titles), using a statistical *n-gram* approach [8]. In [9,4] such complex entities are recognized using a set of simple rules. For example, a movie title is recognized when it is placed between quotation marks.

Cimiano and Staab [10] describe a method to use a search engine to verify a hypothesis relation. For example, the number of hits to “*rivers such as the Nile*” and “*cities such as the Nile*”) are compared to determine the class of *Nile*. Per instance, the number of queries is linear in the number of classes considered.

Where Cimiano and Staab assume the relation patterns to be given, in [11] a technique is introduced to find precise patterns using a training set of related instances. In [4] a method is presented to extract information from the web using effective patterns, as precision is not the only criterion for a pattern to provide useful results.

The number of search engine *hits* for pairs of instances can be used to compute a semantic distance between the instances [12]. The nature of the relation is not identified, but the technique can for example be used to cluster related instances. In [13] a similar method is used to cluster artists using search engine counts. However, the total number of hits provided by the search engine is an estimate and not always reliable [14]. In [15] an approach is presented where one instance is queried and the resulting texts are mined for occurrences of other instances. Such an approach is not only more efficient in the number of queries, but also gives better results.

The extraction of social networks using web data is a frequently addressed topic. For example, Mori et al. [16] use *tf-idf* to identify relations between politicians and locations and [17] use inner-sentence co-occurrences of company names to identify a network of related companies.

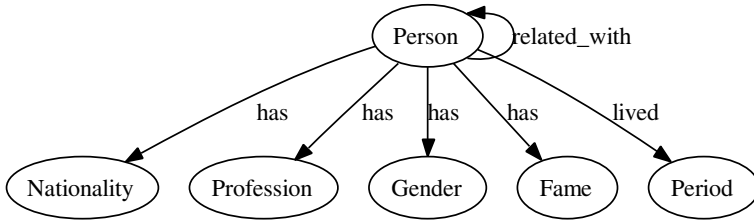


Fig. 1. The ontology on historical persons to be populated

### 3 Problem Description

Suppose that we are given an ontology  $O$  with  $n$  classes  $c_1, \dots, c_n$  and a set  $\mathcal{R}$  of relation classes  $R_{i,j}$  on classes  $c_i$  and  $c_j$ . For example, if ‘Person’ and ‘City’ are classes, then ‘is born in(Person, City)’ is a relation class on these classes. The tuple ‘(Napoleon Bonaparte, Ajaccio)’ may be an instance of ‘is born in(Person, City)’ if ‘Napoleon Bonaparte’ is an instance of ‘Person’ and ‘Ajaccio’ is an instance of ‘City’. We call a class *complete* if all instances are assumed to be given beforehand. For *empty* classes no instances are given.

**Problem.** Given the ontology  $O$ ,

- (1.) populate the incomplete classes with instances extracted from the web, and
- (2.) populate the relation classes in  $\mathcal{R}$  by identifying formulations of all relations  $R_{i,j}$  between the instances of  $c_i$  and  $c_j$ .  $\square$

In this work we focus on the population of an ontology on historical persons. In the given ontology (cf. Figure 1) all classes but *Person* are complete, while *Person* is empty. The class *Period* contains all combinations of years that are likely to denote the periods of life of a historical person. For example ‘1345 - 1397’ is an instance of *Period*. The class *Nationality* contains all names of countries in the world. We identify derivatives of country names as well and use them as synonyms (e.g. *American* for *United States*, *English* for *United Kingdom* and *Flemish* for *Belgium*). A hierarchy among the instances of *Nationality* is defined using the names of the continent, such that we can for example select a list of historical persons from Europe. Likewise, the instances of *Profession* reflect 88 professions. For the instances *male* and *female* we have added a list of derivatives to be used as synonyms, namely the terms *he*, *his*, *son of*, *brother of*, *father of*, *man* and *men* for *male* and the analogous words for *female*. We use the class *Fame* to rank the retrieved instances of *Person* according to their presence on the web. Hence the task is to identify a collection of biographies of historical persons and to identify a social network between the persons found. As persons may have more than one profession and can be related to multiple other people, we are interested in a ranked list of professions and related persons for each historical person found.

For efficiency reasons, we only extract information from the snippets returned by the search engine. Hence, we do not download full documents. As only a limited amount of

automated queries per day are allowed<sup>1</sup>, the approach should be efficient in the number of queries.

## 4 Finding Biographies

We use the given instances in the ontology  $O$  to populate the class *Person*, i.e. to find a list of names of historical persons. We use the instances available in the given ontology to formulate queries and hence create a corpus to extract information from.

Using a pattern-based approach, we combine natural language formulations of relations in  $O$  with known instances into queries. Such queries lead to highly relevant search results, as the snippets are expected to contain related instances. As search engines return only a limited amount of results and the same names may occur in multiple search results, we have to identify effective queries [4].

Suppose we use instances in the class *Profession* to extract the persons. When querying for the instance '*composer*', it is likely that few well-known composers dominate the search results. As we are interested in a rich ontology of historical persons, this is thus a less-suited approach.

The class *Period* contains all combinations of years that are likely to denote the periods of life of a historical person. Hence, the number of instances known for the class *Period* is by far the largest for all complete classes in  $O$ . As it is unlikely that many important historical persons share both a date of birth and a date of death, the use of this class is best suited to obtain a long and diverse list of persons. The names of historical persons are often preceded in texts with a period in years (e.g. '*Vincent van Gogh (1853 - 1890)*'). As this period is likely to denote the period he or she lived in, we choose the pattern "(year of birth – year of death)" to collect snippets to identify the names of historical persons.

### 4.1 Identifying Person Names in Snippets

Having obtained a collection of snippets, the next problem is to extract instances from the texts, in this case persons names. We choose to identify the names within the snippets using a rule-based approach. Although the design of rules is laborious, we do not opt for an approach based on machine learning (e.g. [18,19,8]) for the following reasons.

1. No representative training set is available. The corpus – the collection of snippets – contains broken sentences and improper formulations. Moreover, the corpus is multi-lingual.
2. The Named Entity Recognition (NER) task is simplified by the use of the patterns. Since we expect person names preceding the period queried, we know the placeholder of the named entity (i.e. preceding the queried expression). In a general NER task this information is not available.
3. Since the corpus consists of uncontrolled texts, there is need for post-processing. The snippets may contain typos, missing first or last names, and errors in dates and names.

---

<sup>1</sup> Yahoo! accepts 5,000 queries per day.



We hence use a rule-based approach to identify person names in the snippets found with the periods.

First we extract all terms directly preceding the queried expressing that match a regular expression. That is, we extract terms of two or three capitalized words and compensate for initials, inversions (e.g. '*Bach, Johann Sebastian*'), middle names, Scottish names (e.g. *McCloud*) and the like.

Subsequently, we remove extracted terms that contain a word in a tabu list (e.g. '*Biography*') and names that only occur once within the snippets. Having filtered out a set of potential names of persons, we use a string matching among the extracted names to remove typos and names extracted for the wrong period.

Using the 80,665 periods identified, we obtain a list of 28,862 terms to be added as instance to the class *Person*. Simultaneously, we extract the relations between the periods queried and the extracted instances.

In the evaluation section we analyze the quality of the extracted instances and compare the rule-based approach with a state-of-the-art named entity recognizer based on machine learning.

## 4.2 Using Mined Names to Find Additional Biographical Information

Having found a list of instances of the class *Person*, we first determine a ranking of the instances extracted.

**Finding a Rank.** To present the extracted information in an entertaining manner, we determined the number of hits for each identified person. As names are not always unique descriptors, we queried for the combination of the last name and period (e.g. '*Rubens (1577 - 1640)*'). Although the number of hits returned a search engine is an estimate and irregularities may occur [14], we consider this simple and efficient technique to be well suited for this purpose.

Now we use the names of these instances in a similar fashion to acquire biographical information for the 10,000 best ranked persons. To limit the number of queries per instance, we select the pattern 'was' to reflect the relation between *Person* on the one hand and *Nationality*, *Gender* and *Profession* on the other hand. By querying phrases such as '*Napoleon Bonaparte was*' we thus expect to acquire sentences containing the biographical information. Table 1 contains examples of the sentences used to determine biographical information. We scan these sentences for occurrences of the instances (and their synonyms) of the related classes.

**Relating persons to a gender.** We simply counted instances and their synonyms within the snippets that refer to the gender '*male*' the opposite words that refer '*female*'. We simply related each instance of *Person* to the gender with the highest count.

**Relating persons to a nationality.** We assigned the nationality with the highest count.

**Relating persons to professions.** For each person, we assigned the profession  $p$  that most frequently occurred within the snippets retrieved. Moreover, as persons may have multiple professions, all other professions with a count at least half of the count of  $p$  were added.



**Table 1.** Example search results for the query 'Napoleon Bonaparte was'

**Napoleon Bonaparte was** the greatest military genius of the 19th century  
**Napoleon Bonaparte was** born of lower noble status in Ajaccio, Corsica on August 15, 1769  
**Napoleon Bonaparte was** effectively dictator of France beginning in 1799 and  
**Napoleon Bonaparte was** the emperor of France in the early 1800s  
**Napoleon Bonaparte was** a bully, rude and insulting  
**Napoleon Bonaparte was** in Egypt and was not enjoying his tour  
**Napoleon Bonaparte was** a great warrior and a notable conqueror  
**Napoleon Bonaparte was** born on August 15, 1769 to Carlo and Letizia Bonaparte  
**Napoleon Bonaparte was** defeated at Waterloo

Hence, using one query per instance of *Person*, we identify basic biographical information.

## 5 Identifying a Social Network

Having gathered a list of historical persons with biographical information, we are interested how the persons in the list are perceived to be related. Obviously such information can be extracted from the biographies, e.g. persons can be considered related when they share a profession, have the same nationality or lived in the same period.

However, we are interested in the way people nowadays relate the historical people extracted. For example, we are interested to identify the person who is considered to be most related to Winston Churchill. We therefore mine the Web for a social network of people extracted using the method in the previous section.

We assume that two persons are related when they are often mentioned in the same context. In early work, search engine queries were used that contained both the terms (e.g. [10][12]). The number of Google hits to such queries were used as co-occurrence count  $\text{co}(u, v)$ . However, work in [15] has shown that a pattern-based approach is not only more efficient in the number of queries, but also gives better results. Hence, we opt for an approach similar as described in the previous section, where we use patterns to identify relatedness between persons. We use the total numbers of co-occurrences  $\text{co}(p, q)$  for persons  $p \neq q$  to compute the relatedness  $T(p, q)$  of  $p$  to  $q$ . Using the hypothesis that enumerated items are often related, we use patterns expressing enumerations (Table 2) to obtain the snippets.

For each historical person  $p$  we could consider the person  $q$  with the highest  $\text{co}(p, q)$  to be the most related to  $p$ . However, we observe that, in that case, frequently occurring persons have a relatively large probability to be related to any person. This observation leads to a normalized approach.

$$T(a, b) = \frac{\text{co}(a, b)}{\sum_{c, c \neq b} \text{co}(c, b)} \quad (1)$$

For each of the best 3,000 ranked persons, we computed a ranked list of most related persons in the large set of the 10,000 persons with biographies.

**Table 2.** Patterns used to find co-occurrences within the search engine snippets

"like [person] and [person]"	"namely [person] and [person]"
"such as [person] and [person]"	"[person] and [person]"
"including [person] and [person]"	"[person] [person] and other"
"for example [person] and [person]"	

## 6 Experimental Results

In this section we discuss the results of the ontology population method applied to the ontology on historical persons. We present examples of the extracted data to give the reader an impression of the results. Moreover, we show that structured data can be used to gain insights. Section 6.1 handles the extracted instances of *Person* and the identified biographical relations, while Section 6.2 handles the social network identified of the extracted persons. As no prior work is known in this domain, we cannot compare the results with others.

### 6.1 Evaluating the Identified Biographical Information

The rank assigned to each of the persons in the list provides a mechanism to present the extracted data in an attractive manner. Table 3 gives the list of the 25 best ranked persons and the identified biographical information. Using the criterion defined in Section 4, Johann Sebastian Bach is thus the best known historical figure.

As the data is structured, we can also perform queries to select subsets of the full ranked list of persons. For example, we can create a list of best ranked artists (Table 4), or a 'society' of poets (Table 5). We note that Frédéric Chopin is often referred to as 'the poet of the piano'. Table 6 shows that Vincent van Gogh is the best ranked Dutch painter.

The reader can verify that the given list of extracted persons are highly accurate. However, lacking a benchmark set of *the* best known historical persons, we manually evaluated samples of the extracted ontology to estimate precision and recall.

**Precision.** To estimate the precision of the class *Person*, we selected three decennia, namely 1220-1229, 1550-1559 and 1880-1889, and analyzed for each the candidate persons that were found to be born in this decennium. For the first two decennia we analyzed the complete list, for decennium 1880-1889 we analyzed only the first 1000 as well as the last 1000 names. This resulted in a precision of 0.94, 0.95, and 0.98, respectively. As the decennium of 1880-1889 resulted in considerably more names, we take a weighted average of these results. This yields an estimated precision for the complete list of 0.98.

We compare the precision of the rule-based approach with a state-of-the-art machine-learning-based algorithm, the Stanford Named Entity Recognizer (SNER [20]), trained on the CoNLL 2003 English training data. Focussing on persons born in the year 1882, using the rule-based approach we extracted 1,211 terms. SNER identified 24,652 unique terms as person names in the same snippets. When we apply the same post-processing on SNER extracted data (i.e. removing typos by string matching, single-word names

**Table 3.** The 25 historical persons with the highest rank

Johann Sebastian Bach (1685-1750)	Germany	composer,organist
Wolfgang Amadeus Mozart (1756-1791)	Austria	composer,musician
Ludwig van Beethoven (1770-1827)	Germany	composer
Albert Einstein (1879-1955)	Germany	scientist,physicist
Franz Schubert (1797-1828)	Austria	composer
Johannes Brahms (1833-1897)	Germany	composer
William Shakespeare (1564-1616)	United Kingdom	author,poet
Joseph Haydn (1732-1809)	Austria	composer
Johann Wolfgang Goethe (1749-1832)	Germany	philosopher,director,poet..
Charles Darwin (1809-1882)	United Kingdom	naturalist
Robert Schumann (1810-1856)	Germany	composer
Leonardo da Vinci (1452-1519)	Italy	artist,scientist,inventor
Giuseppe Verdi (1813-1901)	Italy	composer
Frederic Chopin (1810-1849)	Poland	composer,pianist,poet
Antonio Vivaldi (1678-1741)	Italy	composer
Richard Wagner (1813-1883)	Germany	composer
Ronald Reagan (1911-2004)	United States	president
Franz Liszt (1811-1886)	Hungary	pianist,composer
Claude Debussy (1862-1918)	France	composer
Henry Purcell (1659-1695)	United Kingdom	composer
Immanuel Kant (1724-1804)	Germany	philosopher
James Joyce (1882-1941)	Ireland	author
Friedrich Schiller (1759-1805)	Germany	poet,dramatist
Georg Philipp Telemann (1681-1767)	Germany	composer
Antonin Dvorak (1841-1904)	Czech Republic	composer

**Table 4.** The 25 artists with the highest rank

Leonardo da Vinci (1452 - 1519)	Italy	artist, scientist,...
Pablo Picasso (1881 - 1973)	Spain	artist
Vincent van Gogh (1853 - 1890)	Netherlands	artist, painter
Claude Monet (1840 - 1926)	France	artist, painter,...
Pierre-Auguste Renoir (1841 - 1919)	France	painter
Paul Gauguin (1848 - 1903)	France	painter
Edgar Degas (1834 - 1917)	France	artist, painter,...
Paul Cezanne (1839 - 1906)	France	painter, artist
Salvador Dali (1904 - 1989)	Spain	artist
Henri Michaux (1899 - 1984)	Belgium	artist, poet
Gustav Klimt (1862 - 1918)	Austria	painter, artist
Peter Paul Rubens (1577 - 1640)	Belgium	artist, painter
Katsushika Hokusai (1760 - 1849)	Japan	painter
Amedeo Modigliani (1884 - 1920)	Italy	artist, painter
JMW Turner (1775 - 1851)	United Kingdom	artist, painter
James McNeill Whistler (1834 - 1903)	United States	artist
Rene Magritte (1898 - 1967)	Belgium	artist, painter
Henri Matisse (1869 - 1954)	France	artist
Rembrandt van Rijn (1606 - 1669)	Netherlands	artist, painter
Edouard Manet (1832 - 1883)	France	artist, painter
Herm Albright (1876 - 1944)	-	artist, engraver,...
Marc Chagall (1887 - 1985)	Russia	painter, artist
Edvard Munch (1863 - 1944)	Norway	painter, artist
Wassily Kandinsky (1866 - 1944)	Russia	artist, painter
Francisco Goya (1746 - 1828)	Spain	artist, painter

and names extracted for different periods), 2,760 terms remain, of which 842 overlap with the terms extracted using the rule-based approach.

We manually inspected each of these 2,760 terms, resulting in a precision of only 62%. Around half of the correctly extracted names are not recognized by the rule-based

**Table 5.** The 20 best ranked poets

William Shakespeare (1564-1616)	United Kingdom	author,poet
Johann Wolfgang Goethe (1749-1832)	Germany	poet, psychologist, philosopher..
Frederic Chopin (1810-1849)	Poland	composer,pianist,poet
Friedrich Schiller (1759-1805)	Germany	poet,dramatist
Oscar Wilde (1854-1900)	Ireland	author,poet
Jorge Luis Borges (1899-1986)	Argentina	author,poet
Victor Hugo (1802-1885)	France	author,poet,novelist
Ralph Waldo Emerson (1803-1882)	United States	poet,philosopher,author
William Blake (1757-1827)	United Kingdom	poet
Dante Alighieri (1265-1321)	Italy	poet
Robert Frost (1874-1963)	United States	poet
Heinrich Heine (1797-1856)	Germany	poet
Robert Louis Stevenson (1850-1894)	Samoa	engineer,author,poet
Alexander Pope (1688-1744)	United Kingdom	poet
Hildegard von Bingen (1098-1179)	Germany	composer,scientist,poet
Lord Byron (1788-1824)	Greece	poet
John Donne (1572-1631)	United Kingdom	poet,author
Henri Michaux (1899-1984)	Belgium	poet
Walt Whitman (1819-1892)	United States	poet
Robert Burns (1759-1796)	United Kingdom	poet

**Table 6.** The 10 best ranked painters from the Netherlands

Vincent van Gogh (1853-1890)	Kees van Dongen (1877-1968)
Rembrandt van Rijn (1606-1669)	Willem de Kooning (1904-1997)
Johannes Vermeer (1632-1675)	Pieter de Hooch (1629-1684)
Piet Mondrian (1872-1944)	Jan Steen (1626-1679)
Carel Fabritius (1622-1654)	Adriaen van Ostade (1610-1685)

approach, most of them due to the fact that these names did not directly preceded the queried period.

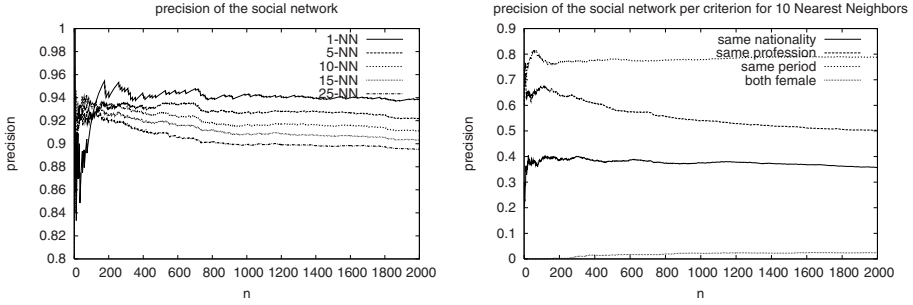
To estimate the precision of the extracted biographical relations, we inspected randomly selected sublists of the top 2500 persons. When we focus on the best scoring professions for the 2500 persons, we estimate the precision of this relation to be 96%. We did not encounter erroneously assigned genders, while we found 98% of the cases the right *Nationality*, if one is found.

Hence, we conclude that the ontology populated using the rule-based approach is precise.

**Recall.** We estimate the recall of the instances found for *Person* by choosing a diverse set of six books containing short biographies of historical persons. Of the 1049 persons named in the books, 1033 were present in our list, which gives a recall of 0.98. For further details on the chosen books we refer to [21].

From Wikipedia, we extracted a list of important 1882-born people<sup>2</sup>. The list contains 44 persons. Of these 44 persons, 34 are indeed mentioned in the Google snippets found with the queried patterns. Using the rule-based approach, we identified 24 of these persons within the snippets. The other ones were only mentioned once (and hence not recognized) or found in different places in the snippets, i.e. not directly preceding the queried period. Using SNER, we identified 27 persons from the Wikipedia list.

<sup>2</sup> <http://en.wikipedia.org/wiki/1882>, visited January 2007.



**Fig. 2.** (l.) Precision for the social network for the  $n$  best ranked persons and their  $k$  nearest neighbors. (r.) Precision for 10-NN per criterion.

For the recall of the identified biographical relations, we observe that for the 10,000 persons that we considered all were given a gender, 77% were given a nationality, and 95% were given one or more professions.

### 6.2 Evaluating the Social Network

Aiming for a reflection of the collective knowledge of web contributors on historical figures, the extracted social network of historical persons is not a verifiable collection of facts. We illustrate the social network extracted by two examples. Figure 3 depicts the relatedness among the best ranked persons. An arrow from person  $p$  to  $q$  is drawn if  $q$  is among the 20 nearest neighbors of  $p$ . Using the same criterion, Figure 4 depicts the relatedness among the best ranked authors.

We are able to verify the precision of the relatedness between historical persons if we make the following assumptions. We consider two persons to be related if either (1.) they lived in the same period, i.e. there is an overlap in the periods the two lived, (2.) they shared a profession, (3.) they shared a nationality, or (4.) they are both female.

Of course we cannot evaluate recall of the algorithm on these criteria, as for example not all persons sharing a nationality need to be consider to be related. We therefore the evaluate precision of the social network on these “minimal criteria”. For the 3,000 best ranked persons, we select the  $k$  most related persons. Per pair we evaluate whether either one of the four criteria is being met. This precision rate is presented in Figure 2 (l.). In comparison, the probability of any of the 3,000 persons to be related to a person in the large list of 10,000 is 45%. The precision rates for the social network per criterion can be found in Figure 2 (r.). The probabilities for two randomly selected persons to share a period, profession and nationality are 38%, 7.5% and 6.5% respectively. The chance that two historical persons are both female is only 0.5%. We hence conclude that these results give good confidence on the quality of extracted social network.

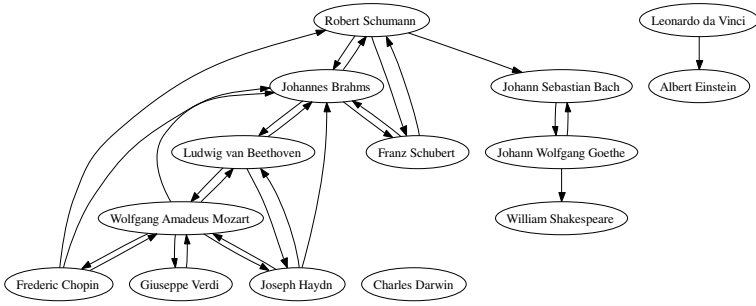


Fig. 3. The extracted social network for the 15 most best ranked persons

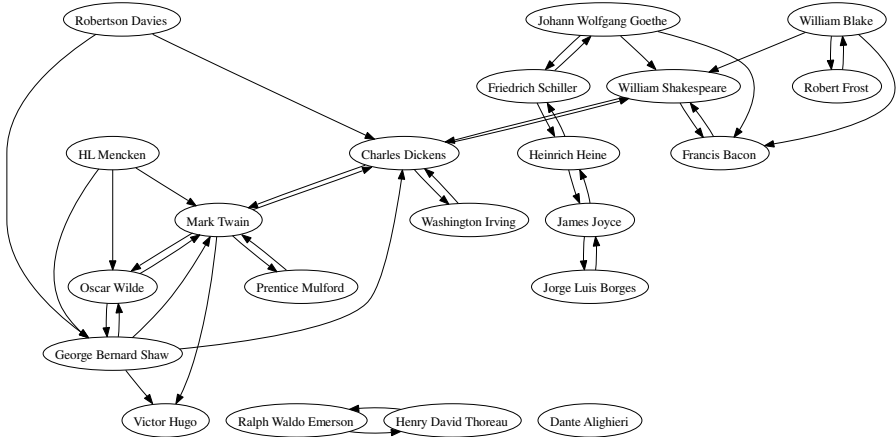


Fig. 4. The extracted social network for the best ranked authors

## 7 Conclusions

We illustrated a simple method to populate an ontology using a web search engine by finding biographical information on historical persons. Starting with the empty class *Person*, we found over 28 thousand historical persons with a high precision rate. The biographical information identified for the best ranked persons has shown to be of high quality as well. The same method is used to create a social network for the historical persons, with convincing results.

Hence, we show that simple Web Information Extraction techniques can be used to precisely populate ontologies. By combining and structuring information from the Web, we create a valuable surplus to the knowledge already available.

In future work, we plan to further address the automatic identification of characterizations for other items such as movies and musical artists. The identification of collective knowledge and opinions is perhaps more interesting than collecting plain facts,

which often can be mined from semi-structured sources. By combining information extracted from multiple web pages, we plan to research methods to automatically tag items.

## References

1. McDowell, L., Cafarella, M.J.: Ontology-driven information extraction with ontosyphon. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 428–444. Springer, Heidelberg (2006)
2. Etzioni, O., Cafarella, M.J., Downey, D., Popescu, A., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence* 165(1), 91–134 (2005)
3. van Hage, W.R., Kolb, H., Schreiber, G.: A method for learning part-whole relations. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 723–736. Springer, Heidelberg (2006)
4. Geleijnse, G., Korst, J.: Learning effective surface text patterns for information extraction. In: ATEM 2006. Proceedings of the EACL 2006 workshop on Adaptive Text Extraction and Mining, Trento, Italy, pp. 1–8 (2006)
5. Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th conference on Computational linguistics, Nantes, France, pp. 539–545 (1992)
6. Crescenzi, V., Mecca, G.: Automatic information extraction from large websites. *Journal of the ACM* 51(5), 731–779 (2004)
7. Downey, D., Etzioni, O., Soderland, S.: A probabilistic model of redundancy in information extraction. In: IJCAI 2005. Proceeding of the 19th International Joint Conference on Artificial Intelligence, Edinburgh, UK pp. 1034–1041 (2005)
8. Downey, D., Broadhead, M., Etzioni, O.: Locating Complex Named Entities in Web Text. In: IJCAI 2007. Proceedings of the Twentieth International Joint Conference on Artificial Intelligence, Hyderabad, India (2007)
9. Sumida, A., Torisawa, K., Shinzato, K.: Concept-instance relation extraction from simple noun sequences using a full-text search engine. In: WebConMine. Proceedings of the ISWC 2006 workshop on Web Content Mining with Human Language Technologies, Athens, GA (2006)
10. Cimiano, P., Staab, S.: Learning by Googling. *SIGKDD Explorations Newsletter* 6(2), 24–33 (2004)
11. Ravichandran, D., Hovy, E.: Learning surface text patterns for a question answering system. In: ACL 2002. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, pp. 41–47 (2002)
12. Cilibrasi, R., Vitanyi, P.: Automatic meaning discovery using Google (2004), <http://www.cwi.nl/~{}paulv/papers/amdug.pdf>
13. Zadel, M., Fujinaga, I.: Web services for music information retrieval. In: ISMIR 2004. Proceedings of 5th International Conference on Music Information Retrieval, Barcelona, Spain (2004)
14. Véronis, J.: Weblog (2006), <http://aixtal.blogspot.com>
15. Geleijnse, G., Korst, J., de Boer, V.: Instance classification using co-occurrences on the web. In: WebConMine. Proceedings of the ISWC 2006 workshop on Web Content Mining with Human Language Technologies, Athens, GA (2006), <http://orestes.ii.uam.es/workshop/3.pdf>

16. Mori, J., Tsujishita, T., Matsuo, Y., Ishizuka, M.: Extracting relations in social networks from the web using similarity between collective contexts. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 487–500. Springer, Heidelberg (2006)
17. Jin, Y., Matsuo, Y., Ishizuka, M.: Extracting a social network among entities by web mining. In: WebConMine. Proceedings of the ISWC 2006 workshop on Web Content Mining with Human Language Technologies, Athens, GA (2006)
18. Zhou, G., Su, J.: Named entity recognition using an hmm-based chunk tagger. In: ACL 2002. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, pp. 473–480 (2002)
19. Brothwick, A.: A Maximum Entropy Approach to Named Entity Recognition. PhD thesis, New York University (1999)
20. Finkel, J.R., Grenager, T., Manning, C.D.: Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, Ann Arbor, MI (2005)
21. Korst, J., Geleijnse, G., de Jong, N., Verschoor, M.: Ontology-based extraction of information from the World Wide Web. In: Intelligent Algorithms in Ambient and Biomedical Computing, pp. 149–167. Springer, Heidelberg (2006)



# OBO and OWL: Leveraging Semantic Web Technologies for the Life Sciences

Christine Golbreich<sup>1</sup>, Matthew Horridge<sup>2</sup>, Ian Horrocks<sup>3</sup>, Boris Motik<sup>3</sup>,  
and Rob Shearer<sup>3</sup>

<sup>1</sup>University of Versailles Saint-Quentin  
55 avenue de Paris, 78035 Versailles, France  
`Christine.Golbreich@uvsq.fr`

<sup>2</sup>School of Computer Science, University of Manchester  
Oxford Road, Manchester, M13 9PL, UK  
`horridge@cs.man.ac.uk`

<sup>3</sup>Oxford University Computing Laboratory  
Wolfson Building, Parks Road, Oxford, OX1 3QD, UK  
{`Ian.Horrocks`, `Boris.Motik`, `Rob.Shearer`}@comlab.ox.ac.uk

**Abstract.** OBO is an ontology language that has often been used for modeling ontologies in the life sciences. Its definition is relatively informal, so, in this paper, we provide a clear specification for OBO syntax and semantics via a mapping to OWL. This mapping also allows us to apply existing Semantic Web tools and techniques to OBO. We show that Semantic Web reasoners can be used to efficiently reason with OBO ontologies. Furthermore, we show that grounding the OBO language in formal semantics is useful for the ontology development process: using an OWL reasoner, we detected a likely modeling error in one OBO ontology.

## 1 Introduction

The Open Biomedical Ontologies (OBO) repository is a large library of ontologies from the biomedical domain hosted by the National Center for Biomedical Ontology (NCBO) [\[1\]](http://www.bioontology.org/repositories.html). The majority of the ontologies in that repository are written in OBO Flat File Format [\[2\]](http://www.geneontology.org/G0.format.obo-1.2.shtml)—an ontology language originally designed for the Gene Ontology (GO) [\[3\]](http://oboedit.org/). This language (from now on called simply OBO) uses a simple textual syntax that was designed to be compact, readable by humans, and easy to parse. The OBO community has dedicated significant effort to developing tools such as OBO-Edit [\[3\]](http://oboedit.org/)—an integrated OBO editor and reasoner.

In parallel with the OBO effort, the Semantic Web community has developed the Web Ontology Language (OWL). Three dialects of the language have been defined, two of which are based on Description Logics—a well-understood family of knowledge representation formalisms with desirable computational properties.

---

<sup>1</sup> <http://www.bioontology.org/repositories.html>

<sup>2</sup> <http://www.geneontology.org/G0.format.obo-1.2.shtml>

<sup>3</sup> <http://oboedit.org/>

Formal semantics and the availability of efficient and provably correct reasoning tools have made the OWL DL dialect of OWL the language of choice for ontology development in fields as diverse as biology [22], medicine [5], geography [6], astronomy [2], geology<sup>4</sup> agriculture [24], and defense [14]. Furthermore, OWL has been used to develop several large biomedical ontologies, such as the Biological Pathways Exchange (BioPAX) ontology [21], the GALEN ontology [20], the Foundational Model of Anatomy (FMA) [5], and the National Cancer Institute thesaurus [9]. Recently, the community of OWL users and developers proposed an extension of OWL called OWL 1.1 [18], which has been accepted as a member submission by the W3C. At the same time, a number of OWL-based tools have been developed, such as the Protégé [13] and SWOOP [11] editors, and the FaCT++ [25], RACER [8], and Pellet [23] reasoners.

In Section 2, we argue that there are many benefits in applying the tools and techniques of the Semantic Web to OBO. For example, Semantic Web reasoners could be used to provide guidance during ontology development; furthermore, modularization techniques for OWL [7] could simplify the reuse of existing OBO ontologies. This has been difficult up to now, however, since the OBO and Semantic Web communities have been largely disjoint.

To enable interoperability between OBO and Semantic Web tools and systems, we establish in Section 3 an exact relationship between OBO and OWL. This has not been straightforward, mainly because the OBO specification is quite informal. The syntax of the OBO language has not been formally specified, so our first step was to formalize the syntax of OBO itself; we discuss the results in Section 3.1. Likewise, there is no formal specification of OBO's semantics: the effects of different constructs have been described using natural language. We resolved ambiguities in these descriptions through extensive discussions with OBO developers. Hence, our mapping, presented in Section 3.2, formalizes the consensus interpretation in the OBO community. We also relate our mapping to several existing mappings from OBO to OWL.

In Section 4, we discuss how our mapping is used in practice. In Section 4.1 we discuss the technical aspects of our implementation. The complete replacement of OBO with OWL is not desirable for the OBO community, as many OBO users are familiar with both OBO-Edit and the OBO language, and find them convenient to use. Therefore, instead of simply implementing a translator from OBO to OWL, we have embedded support for OBO into existing Semantic Web ontology management infrastructure. In particular, we extended the well-known OWL API [10] with an OBO parser and serializer. All tools built on top of the OWL API can thus directly load, process, and save OBO ontologies. Moreover, tools such as OBO-Edit could use the new API to provide similar features, including direct access to OWL reasoners.

In Section 4.2, we show that reasoners implementing the formal semantics of OWL can derive subsumption inferences that are missed by OBO-Edit's reasoner. In fact, on one of the OBO ontologies, our reasoner derived a new inference that highlights a probable modeling error.

---

<sup>4</sup> <http://sweet.jpl.nasa.gov/ontology/>

Classifying large biomedical ontologies requires optimized reasoners. In Section 4.3, we show that OWL-based tools can be used to efficiently reason with OBO ontologies. To this end, we classified a number of OBO ontologies using the FaCT++ [25] and Pellet [23] systems, as well as the novel hypertableau-based reasoner Hermit [17].<sup>5</sup> The design of Hermit was motivated by an analysis of the structure of biomedical ontologies such as GALEN and NCI. Our results show that Hermit’s improved handling of GALEN is applicable to OBO ontologies as well: on several ontologies containing complex cyclic definitions of terms, Hermit outperforms the other reasoners by orders of magnitude. Thus, our mapping allows the OBO community to benefit from current and future advances in reasoning technology while continuing to use their familiar ontology language and tools.

## 2 Why Map OBO to OWL 1.1?

### 2.1 OBO at a Glance

An OBO ontology is a collection of *stanzas*, each of which describes one element of the ontology. A stanza is introduced by a line containing a *stanza name* that identifies the type of element being described. The rest of the stanza consists of lines, each of which contains a *tag* followed by a colon, a *value*, and an optional comment introduced by “!”.

The following is an example of an OBO stanza defining the *term* (the OBO equivalent of a class) GO:0001555 with name `oocyte growth`. This term is a subclass of the term GO:0016049. (The comment tells us that GO:0016049 is named `cell growth`.) Furthermore, GO:0001555 has a `part_of` relationship to the term GO:0048601 (which is named `oocyte morphogenesis`). Finally, GO:0001555 is defined as an intersection of GO:0040007 (`growth`) and of a relationship `has_central_participant` to CL:0000023 (`oocyte`).

```
[Term]
id: GO:0001555
name: oocyte growth
is_a: GO:0016049 ! cell growth
relationship: part_of GO:0048601 ! oocyte morphogenesis
intersection_of: GO:0040007 ! growth
intersection_of: has_central_participant CL:0000023 ! oocyte
```

The following stanza defines the *relationship type* (the OBO equivalent of a property) `propreo:is_described_by`. The terms `propreo:chemical_entity` and `_Description177` are used as the domain and range, respectively, of the relationship type being defined.

```
[Typedef]
id: propreo:is_described_by
domain: propreo:chemical_entity
range: _Description177
```

<sup>5</sup> <http://www.cs.man.ac.uk/~bmotik/Hermit/>

Finally, the following stanza defines the *instance* (the OBO equivalent of an individual) `propreo:water_molecule`. The instance is a member of the term `propreo:inorganic_solvent_molecule` and has `propreo:CHEBI_15377` for the value of the relationship `propreo:is_described_by`.

```
[Instance]
id: propreo:water_molecule
instance_of: propreo:inorganic_solvent_molecule
property_value: propreo:is_described_by propreo:CHEBI_15377
```

## 2.2 Why Formalize OBO Syntax?

The line-oriented syntax of OBO makes parsing ontologies into stanzas and tag-value pairs straightforward. The tag values, however, usually have a structure that depends on the tag type. This structure is described in the OBO specification in natural language. For example, the structure of `intersection_of` tag values is described as follows:

This tag indicates that this term represents the intersection of several other terms. The value is either a term id, or a relationship type id, a space, and a term id. [...]

This style of description is quite informal and it does not make the conceptual structure of the OBO language clear. For example, the above description does not provide any intuition behind the distinction between the two alternative structures of allowed values. Furthermore, the specification of the structure is conflated with low-level lexical issues, such as whitespace handling. As a result, neither aspect of the language is robustly addressed; for example, the treatment of escape characters is dependent on the structure of tag values. These issues make the implementation of an OBO parser quite difficult in practice.

## 2.3 Why Formalize OBO Semantics?

The semantics of OBO is also defined informally, by providing natural-language descriptions for different types of tag-value pairs. For example, the OBO specification defines the semantics of the `relationship` tag as follows:

This tag describes a typed relationship between this term and another term. [...] The **necessary** modifier allows a relationship to be marked as “not necessarily true”. [...]

Such a description is clearly ambiguous and informal. The notion of a relationship being “necessarily true” is completely undefined; in fact, the notion of a relationship has not been formalized either. Computational logic can be used to provide an unambiguous interpretation for such statements. For example, the `relation` tag from the stanza for the term `G0:0001555` from Section 2.1 can be interpreted in at least three different ways:

- *Existential interpretation*: Each instance of the term `G0:0001555` must have at least one `part_of` relationship to an instance of the term `G0:0048601`. This reading corresponds to the DL axiom `G0:0001555 ⊑ ∃part_of.G0:0048601`.
- *Universal interpretation*: Instances of `G0:0001555` can be connected through `part_of` relationships only to instances of `G0:0048601`. This reading corresponds to the DL axiom `G0:0001555 ⊑ ∀part_of.G0:0048601`.
- *Constraint interpretation*: Instances of the term `G0:0001555` can be connected through `part_of` relationships; furthermore, the end-points of the relationship must be *known* to be instances of `G0:0048601`. Such a statement cannot be formalized in standard DLs; however, it can be expressed in various extensions of DLs [153].

As another example, consider the natural-language explanation of the semantics for the `intersection_of` tag:

[...] For example:

```
intersection_of: G0:00001
intersection_of: part_of G0:00002
```

This means that the term is a subclass of any term that is both a subclass of `G0:00001` and has a `part_of` relationship to `G0:00002`. [...]

Here, it is not clear whether the defined term is equivalent to or a subclass of the intersection of the other terms. The textual description has a “procedural” flavor: it says that the defined term should be inferred to be a subclass of other terms, so one might conclude that the subclass relationship is the proper reading. Our discussions with the OBO developers, however, revealed that the intended interpretation is equivalence. The `union_of` tag suffers from analogous problems.

The description of OBO-Edit’s reasoner provides an insight into the intended semantics of OBO. The OBO-Edit User’s Guide<sup>6</sup> defines the following three reasoning rules:

1. For each transitive relationship  $R$  (such as `is_a` or `part_of`), whenever the ontology contains  $a \rightarrow R \rightarrow b$  and  $b \rightarrow R \rightarrow c$ , an implied relationship  $a \rightarrow R \rightarrow c$  is added.
2. For each term  $a$  defined as an intersection of terms  $b_1$  and  $b_2$ , implied relationships  $a \rightarrow \text{is\_a} \rightarrow b_1$  and  $a \rightarrow \text{is\_a} \rightarrow b_2$  are added.
3. For each term  $a$  defined as an intersection of terms  $b_1$  and  $b_2$ , whenever some term  $c$  has relationships  $c \rightarrow \text{is\_a} \rightarrow b_1$  and  $c \rightarrow \text{is\_a} \rightarrow b_2$ , an implied relationship  $c \rightarrow \text{is\_a} \rightarrow a$  is added.

This definition is procedural, and it misses important inferences. Consider the following example:

<pre>[Term] id: A relationship: R B</pre>	<pre>[Term] id: B is_a: C</pre>
---	---------------------------------

<sup>6</sup> Available as part of the OBO-Edit distribution.

This simple OBO ontology says that **A** has an **R**-relationship to **B**, and that **B** is a subclass of **C**. Regardless of which of the three previously suggested interpretations for the **relationship** tag we choose, we should derive that **A** has an **R**-relationship to **C**; however, OBO-Edit’s reasoning procedure does not derive that. Furthermore, the second and third inference rules clearly state that **intersection\_of** is interpreted as equivalence, which may be in conflict with the natural-language description of the semantics.

To sum up, OBO suffers from problems very similar to those identified in semantic networks [19]. The DL family of ontology languages was developed precisely to address such problems—that is, to unambiguously specify the semantic properties of all ontology constructs. A mapping of OBO into OWL lends itself as an obvious way of providing formal semantics to OBO, and it allows for the application of sound and complete reasoning algorithms.

## 2.4 Why Use OWL 1.1?

In OBO, it is possible to make a property reflexive and/or (anti-)symmetric, as well as to say that one property is “transitive over” another: if  $P_1$  is transitive over  $P_2$ , then for any individuals  $x$ ,  $y$ , and  $z$ , the relationships  $x \rightarrow P_1 \rightarrow y$  and  $y \rightarrow P_2 \rightarrow z$  imply the relationship  $x \rightarrow P_1 \rightarrow z$ . Such axioms cannot be expressed in OWL DL; however, they can be expressed in the 1.1 extension of OWL. Thus, by using OWL 1.1 as the target language, we can capture a larger subset of OBO [7]. Since OWL 1.1 is fully backwards compatible with OWL, OBO ontologies that do not use any of the additional features of OWL 1.1 are mapped into OWL DL ontologies.

## 2.5 Reusing Existing Tools

An obvious practical benefit of a mapping from OBO to OWL is that it allows OBO users to exploit the multitude of existing OWL tools and services, instead of reimplementing the same functionality from scratch.

The foundation of many Semantic Web tools is provided by various APIs that provide means for the programmatic manipulation of ontologies. The OWL API [10] is a prominent example of such an API that is now very widely used. Recently, it has been completely reengineered and made compliant with the OWL 1.1 version of the language. Jena [8] is a similar API that is comparable in its functionality with the OWL API and also has a large user base.

The OWL API has been used as the core data model for several ontology editors. For example, Protégé [13] is a well-known editor that can be used to edit OWL ontologies. Its newest incarnation, Protégé 4, supports all of OWL 1.1 and is based on the new OWL API. SWOOP [11] is another OWL editor that is based

<sup>7</sup> Our translation captures all of the OBO 1.2 specification except for cyclic properties (the semantics of which is not completely clear) and negative assertions about properties (e.g., the assertion that a property is *not* transitive).

<sup>8</sup> <http://jena.sourceforge.net/>

on the OWL API. These editors have been developed over years and are de facto standards for ontology editing. Furthermore, they are imbued with functionality that can provide guidance to the user during the ontology development and maintenance processes; for example, SWOOP supports ontology debugging [12] and modularization [7]. Finally, a number of plug-ins have been written for Protégé, implementing features such as UML-based ontology editing and graph-based ontology visualization.

Several highly optimized, sound, and complete reasoners for OWL exist. Pellet 1.4 [23] is built around the OWL API and is tightly integrated into SWOOP, and RacerPro [8], FaCT++ [25], and KAON2 [16] can be used with ontology editors through the Description Logics Implementors Group (DIG) protocol.<sup>9</sup> These reasoners can be used to classify an ontology and detect inconsistencies in class definitions, which is valuable during ontology development. Furthermore, reasoners can be used for query answering, which is the core inference underpinning many applications of OWL and OBO.

Apart from leveraging existing results, our integration allows the OBO community to reap the benefits of the future Semantic Web research. Conversely, OBO can provide to the OWL community significant user input as well as a growing library of OWL ontologies.

### 3 Providing a Formal Specification for OBO

In this section, we present a formal specification of the syntax and semantics of OBO. Due to space limitations, we highlight in this paper only the salient points; the full specification is available online.<sup>10</sup>

#### 3.1 Formalization of OBO Syntax

We have formalized the OBO syntax by defining a BNF grammar, which maintains backward compatibility with the original OBO specification. Our grammar has been specifically designed to provide a conceptual description of the structure of OBO ontologies. To this end, it includes nonterminal symbols that identify and group certain OBO structures. Additionally, the grammar can be used to generate OBO parsers using automated tools.

For example, consider the definition of the `intersection_of` tag in BNF:

$$\begin{aligned} \textit{intersection} &:= \textit{intersection\_of} : \textit{termOrRestr} \\ \textit{termOrRestr} &:= \textit{term-id} \mid \textit{restriction} \\ \textit{restriction} &:= \textit{relationship-id} \textit{ term-id} \end{aligned}$$

As explained in Section 2.2, the value of the `intersection_of` tag can be either a term, or a relationship type followed by a term. Our grammar introduces structure to such a “flat” definition as follows. We introduce a nonterminal

<sup>9</sup> <http://dl.kr.org/dig/>

<sup>10</sup> <http://www.cs.man.ac.uk/~horrocks/obo/>

*term-id*, which denotes a “named” term (mimicking OWL’s named classes), and a nonterminal *restriction*, which denotes a “restricted term” (mimicking OWL’s restriction classes). Then, we introduce the nonterminal *termOrRestr* (mimicking OWL’s complex classes). Finally, we say that the value of the *intersection* tag is a *termOrRestr* (mimicking OWL’s intersection classes that can contain arbitrary classes as conjuncts).

### 3.2 Mapping OBO to OWL

Our conceptualization of OBO’s underlying model (described in Section 3.1) is quite similar to that of OWL, so the mapping between the two is relatively straightforward. We map OBO terms to OWL classes, OBO relationship types to OWL properties, and OBO instances to OWL individuals. The *id* assigned to each of these elements is used in OBO to uniquely identify each term, relationship type, and instance; hence, we use the value of *id* as the URI of the corresponding OWL element. The value of the *name* tag provides a human-readable description of OBO ontology elements, so it is translated into an OWL label.

Unlike OBO, OWL requires a strict separation between *object properties* (that relate individuals to each other) and *data properties* (that associate individuals with data values). To map OBO to OWL, we must infer which kind of property is appropriate for each OBO relationship type. If the range of a relationship type *R* is specified to be an XML datatype, or if *R* is asserted to be a subtype of another relationship type with such a range, then we translate *R* as an OWL datatype property; otherwise, we translate *R* as an object property.

OBO constructs such as *is\_a*, *disjoint\_from*, *domain*, and *range* have obvious equivalents in OWL and are translated in the straightforward manner. As discussed in Section 2.3, the official specification of the OBO language allows for several interpretations of the *intersection\_of*, *union\_of*, and *restriction* tags; hence, our mapping into OWL must pick the appropriate one.

Our discussions with OBO developers, as well as a survey of existing OBO ontologies, revealed that the existential interpretation (see Section 2.3) captures the intention behind the *relationship* tag. Hence, we translate the statement *relationship: R B* in a stanza defining the term *A* to the OWL axiom `SubClassOf(A ObjectSomeValuesFrom(R B))`.

Similarly, we concluded that the *intersection\_of* tags should be interpreted as equivalences between classes (see Section 2.3). Furthermore, values for the *intersection\_of* tag that consist of a relationship type and a term should be interpreted as existential constraints, just like *relationship* tags. Hence, we translate the statements *intersection\_of: C* and *intersection\_of: R D* in a stanza defining the term *A* to the following OWL axiom:

```
EquivalentClasses(A
  ObjectIntersectionOf(C ObjectSomeValuesFrom(R D)))
```

There was no consensus on the formal semantics of several OBO constructs, such as the *not\_necessary* modifier to the *relationship* tag and the use of



```
EntityAnnotation(OWLClass(GO_0001555) Label("oocyte_growth"))
SubClassOf(GO_0001555 GO_0016049) SubClassOf(GO_0001555
ObjectSomeValuesFrom(part_of GO_0048601))
EquivalentClasses(GO_0001555 ObjectIntersectionOf(
    GO_0040007 ObjectSomeValuesFrom(
        has_central_participant CL_0000023)))

ObjectPropertyDomain(propreo_is_described_by chemical_entity)
ObjectPropertyRange(propreo_is_described_by Description177)

ClassAssertion(propreo_water_molecule inorganic_solvent_molecule)
ObjectPropertyAssertion(
    is_described_by propreo_water_molecule CHEBI_15377)
```

Fig. 1. The OWL Interpretation of the Stanzas from Section 2.1

relationship tags in Typedef stanzas. In fact, these tags are likely to be deprecated in future releases of OBO, and are currently treated as annotations by our mapping.

Figure 1 shows the translation of the stanzas from Section 2.1.

*Related Work.* Other mappings between the OBO Flat File Format and OWL exist, and a summary can be found online.<sup>11</sup> None of these mappings are based on a formal analysis of OBO. The Common Mapping—a new version of a mapping originally developed by Chris Mungall—is defined via an XSLT style sheet,<sup>12</sup> and has been implemented as a Protégé plug-in. Common Mapping differs from our mapping in several important respects. For example, it reifies OBO annotation values and turns them into first-class elements of the interpretation domain subject to consistency checking and inferencing; in contrast, our translation simply encodes them as OWL 1.1 annotations. Common Mapping translates neither **reflexive** nor **transitive\_over** tags, whereas our encoding preserves their semantics. Finally, for several OBO ontologies, Common Mapping produces OWL Full ontologies, which cannot be efficiently processed by existing Semantic Web tools. In contrast, our translation produces OWL 1.1 ontologies, which can be processed by existing tools without problems.

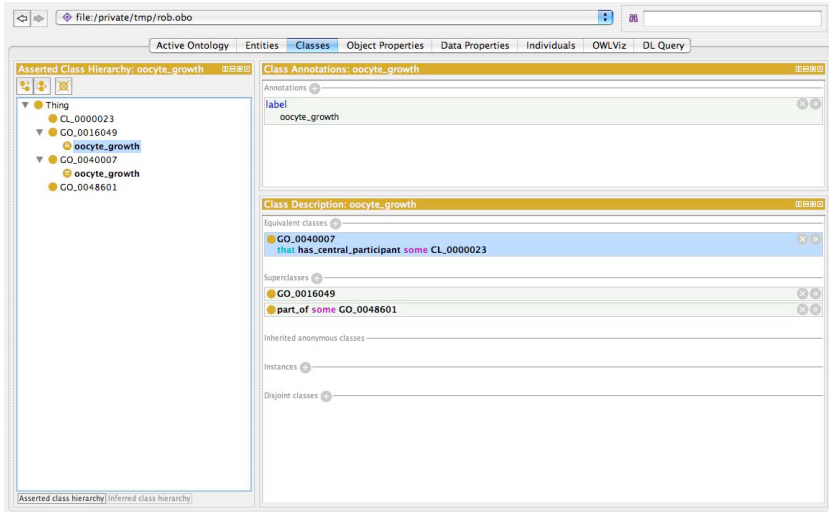
## 4 Integrating OBO with the Semantic Web

### 4.1 Extending Semantic Web Tools to Support OBO

As mentioned in Section 1, simply replacing the OBO language with OWL would not be desirable for the OBO community—OBO users are familiar with the existing syntax of the language and the available tools, and want to continue to

<sup>11</sup> [http://spreadsheets.google.com/ccc?key=pWN\\_4sBrd9l1Umn1LN8WuQQ](http://spreadsheets.google.com/ccc?key=pWN_4sBrd9l1Umn1LN8WuQQ)

<sup>12</sup> [http://www.godatabase.org/dev/xml/xsl/oboxml\\_to\\_owl.xsl](http://www.godatabase.org/dev/xml/xsl/oboxml_to_owl.xsl)



**Fig. 2.** A Screenshot of the Example From Section 2.1 Viewed in Protégé 4

use them. Therefore, we adopted a less intrusive path of integration and have extended existing OWL tools with support for OBO.

As we discussed in Section 2.5, the OWL API lies at the core of many Semantic Web tools and supports fundamental tasks such as reading, saving, and manipulating ontologies. We extended the OWL API with an OBO parser and serializer, thus making OBO just another format supported by the API. This conveniently extends all applications based on the OWL API with support for OBO. For example, Protégé 4 can automatically read, edit, and save OBO ontologies; see Figure 2. Furthermore, the OWL API can be used to convert OBO files into OWL and vice versa by simply loading the file in one format and saving it in another. This functionality can be used to import OBO ontologies into tools that are not based on the OWL API and use custom OWL parsers.

The central new component in the API is an OBO parser, which consists of two distinct parts. The lower-level part is concerned with recognizing the syntax of OBO flat files, and it has been generated automatically from the BNF grammar described in Section 3.1. The upper-level part accepts different constructs from the OBO language and translates them into corresponding OWL 1.1 axioms according to the mapping described in Section 3.2.

## 4.2 Reasoning Support for OBO

An immediate benefit of our work is that it allows the application of Semantic Web reasoners to OBO ontologies. These reasoners are based on well-known algorithms with well-understood formal properties; furthermore, they provide formal guarantees about the completeness of reasoning, which makes the interpretation of derived results much easier. This can be quite useful in practice: on the OBO

ontology `so.obo`, we used an OWL reasoner to detect a probable modeling error that is not detected by the OBO-Edit reasoner. This ontology contains the following stanzas that define the terms `S0:0000992` and `S0:0000914`:

[Term]	[Term]
id: S0:0000992	id: S0:0000914
name: BAC_cloned_genomic_insert	name: cloned_genomic_insert
intersection_of: S0:0000914	

Note that the stanza for `S0:0000992` contains only one `intersection_of` tag. This seems to be a modeling error: presumably, the author simply forgot to add another `intersection_of` tag-value pair.

According to the mapping from Section 3.2, `intersection_of` defines a term to be equivalent to the intersection of other terms. Because the above intersection contains only one term, it effectively makes `S0:0000992` equivalent to `S0:0000914`. Indeed, OWL reasoners (correctly) derive that `S0:0000992` is a subclass of `S0:0000914` and vice versa. OBO-Edit’s reasoner, however, only derives that `S0:0000992` is a subclass of `S0:0000914`, so the error remains undetected.

It is instructive to examine why OBO’s reasoner does not derive the required inference. Namely, this inference could potentially be derived by applying the third inference rule from Section 2.3 on page 173 for  $a = S0:0000992$  and  $b_1 = b_2 = c = S0:0000914$ ; however, for the rule to be applicable, we would need an `is_a` relationship from `S0:0000914` to itself. Semantically, each class is a subclass of itself; this fact, however, is not represented explicitly in the OBO ontology model, so the mentioned inference rule is not applicable.

This error might have been detected by checking whether each stanza contains at least two `intersection_of` tags. Since the syntax of the OBO language, however, has not been formally specified, it is hard to implement a comprehensive set of such checks, so errors often fall thorough to the semantic level. Furthermore, if our example stanza contained two `intersection_of` tags with the same value, the ontology would be syntactically correct, but would imply the same consequence. The OBO-Edit reasoner does not derive all inferences even with respect to the informal semantics, so such an error would not be detected at the semantic level either. In contrast, the syntax and the semantics of OWL 1.1 have been formally specified, which makes the detection of errors easier.

### 4.3 Performance of Reasoning with OBO

Ontologies for the life sciences frequently contain many highly interconnected axioms with “cyclic definitions” of ontology terms. Such ontologies pose significant challenges to state-of-the-art tableau-based OWL reasoners [417]. Hence, it is interesting to see whether the OBO ontologies can be effectively handled using modern Semantic Web reasoning tools. Our mapping would clearly be much less useful if OWL reasoners were unable to process OBO ontologies.

Therefore, we conducted several reasoning experiments using different OBO ontologies and tools. In particular, we measured the times needed to compute the subsumption hierarchy of a large set of OBO ontologies. We used the well-known

**Table 1.** Performance of Reasoning with OBO Ontologies

Tools	No. of ontologies classified in						
	200 ms	1 s	5 s	25 s	53 s	163 s	3925 s
Pellet	2	13	36	51	59	64	79
FaCT++	25	58	72	77	78	80	80
HermiT	48	65	74	81	82	82	82

reasoners Pellet and FaCT++, and a new reasoner HermiT. The latter reasoner employs novel reasoning algorithms based on hypertableau and hyperresolution [17]. The development of these algorithms has been motivated by an analysis of the structure of GALEN—the well-known biomedical terminology for which reasoning has proved to be quite hard. HermiT is currently the only reasoner that can classify the original version of GALEN [17] [13]

Although the reasoning algorithms from [17] support most of the OWL language, currently only the so-called Horn subset of OWL has been implemented in HermiT. Of the 88 ontologies available in the OBO repository, 83 fall into the supported fragment, so we used these ontologies in our performance tests. The ontologies are of varying sizes: the smallest one contains 166 axioms, whereas the largest one contains 37,943 axioms.

We summarize the times needed to classify these ontologies in Table 1. Because of the large number of ontologies, we do not present individual times for each ontology; instead, we just show how many ontologies each tool can classify within a certain time limit. The first four times were selected arbitrarily, whereas the last three times were chosen to show how long it takes for each tool to process the hardest ontology that it can classify.

Our results show that HermiT efficiently deals with all but one ontology—that is, it classifies them in at most 53 seconds. FaCT++ exhausts the available resources on two ontologies that HermiT can classify. Thus, HermiT’s novel reasoning techniques seem to be critical in dealing with complex ontologies. In the future, similar advances are likely to follow. By defining OBO in terms of a mapping to OWL, the OBO community can reap the benefits of these advances while continuing to enjoy the existing OBO syntax and tool set.

## 5 Conclusion

OBO is a language that has been extensively used for ontology modeling in the life sciences. Until now, the OBO language, as well as accompanying tools, have been developed independently from Semantic Web languages and tools. We argue that much can be gained from bringing the two languages closer together.

<sup>13</sup> The original version of GALEN could not be processed by existing reasoners. Therefore, different versions of GALEN were derived from the original one by removing several cyclic definitions. Please refer to HermiT’s web page for more information on this issue.

On the one hand, this allows the OBO community to reuse OWL tools, while on the other hand, it provides new requirements and makes a large new corpus of ontologies available to the Semantic Web community.

The official specification of OBO is relatively informal. To obtain an unambiguous specification that can be easily implemented, we first formalized the syntax of the OBO language by capturing it in BNF. To capture the semantics, we developed a mapping between OBO and OWL. We have implemented this transformation in a new parser that we integrated into the OWL API, thus allowing numerous Semantic Web applications to use OBO as a native format. Finally, we showed that existing Semantic Web reasoners, such as HermiT, can be used to efficiently reason with OBO ontologies and can even identify likely modeling errors.

## Acknowledgment

We thank Chris Mungall and John Day-Richter for their invaluable input on the definition and common usage of the OBO language.

## References

1. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Gene Ontology: Tool for the Unification of Biology. *Nature Genetics* 25(1), 25–29 (2000)
2. Derriere, S., Richard, A., Preite-Martinez, A.: An Ontology of Astronomical Object Types for the Virtual Observatory. In: Proc. of the 26th meeting of the IAU: Virtual Observatory in Action: New Science, New Technology, and Next Generation Facilities, Prague, Czech Republic, August 21–22, 2006, pp. 17–18 (2006)
3. Donini, F.M., Nardi, D., Rosati, R.: Description Logics of Minimal Knowledge and Negation as Failure. *ACM Transactions on Computational Logic* 3(2), 177–225 (2002)
4. Gardiner, T., Horrocks, I., Tsarkov, D.: Automated Benchmarking of Description Logic Reasoners. In: Proc. of DL 2006. CEUR, vol. 189 (2006)
5. Golbreich, C., Zhang, S., Bodenreider, O.: The Foundational Model of Anatomy in OWL: Experience and Perspectives. *Journal of Web Semantics* 4(3), 181–195 (2006)
6. Goodwin, J.: Experiences of using OWL at the Ordnance Survey. In: Proc. of OWL-ED 2005. CEUR, vol. 188 (2005)
7. Grau, B.C., Horrocks, I., Kazakov, Y., Sattler, U.: Just the Right Amount: Extracting Modules from Ontologies. In: Proc. of WWW 2007, pp. 717–726 (2007)
8. Haarslev, V., Möller, R.: RACER System Description. In: Goré, R.P., Leitsch, A., Nipkow, T. (eds.) IJCAR 2001. LNCS (LNAI), vol. 2083, pp. 701–706. Springer, Heidelberg (2001)
9. Hartel, F.W., de Coronado, S., Dionne, R., Fragoso, G., Golbeck, J.: Modeling a Description Logic Vocabulary for Cancer Research. *Journal of Biomedical Informatics* 38(2), 114–129 (2005)

10. Horridge, M., Bechhofer, S., Noppens, O.: Igniting the OWL 1.1 Touch Paper: The OWL API. In: Proc. OWL-ED 2007. CEUR, vol. 258 (2007)
11. Kalyanpur, A., Parsia, B., Hendler, J.: A Tool for Working with Web Ontologies. *International Journal on Semantic Web and Information Systems* 1(1), 36–49 (2005)
12. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging Unsatisfiable Classes in OWL Ontologies. *Journal of Web Semantics* 3(4), 243–366 (2005)
13. Knublauch, H., Fergerson, R.W., Noy, N.F., Musen, M.A.: The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 229–243. Springer, Heidelberg (2004)
14. Lacy, L., Aviles, G., Fraser, K., Gerber, W., Mulvehill, A., Gaskill, R.: Experiences Using OWL in Military Applications. In: Proc. of OWL-ED 05. CEUR, vol. 188 (2005)
15. Motik, B., Horrocks, I., Sattler, U.: Bridging the Gap Between OWL and Relational Databases. In: Proc. of WWW 2007, pp. 807–816 (2007)
16. Motik, B., Sattler, U.: A Comparison of Reasoning Techniques for Querying Large Description Logic ABoxes. In: Hermann, M., Voronkov, A. (eds.) LPAR 2006. LNCS (LNAI), vol. 4246, pp. 227–241. Springer, Heidelberg (2006)
17. Motik, B., Shearer, R., Horrocks, I.: Optimized Reasoning in Description Logics using Hypertableaux. In: Proc. of CADE-21. LNCS (LNAI), vol. 4603, pp. 67–83. Springer, Heidelberg (2007)
18. Patel-Schneider, P.F., Horrocks, I.: OWL 1.1 Web Ontology Language Overview. W3C Member Submission (December 19, 2006), Available at <http://www.w3.org/Submission/owl11-overview/>
19. Quillian, M.R.: Semantic Memory. In: Minsky, M. (ed.) *Semantic Information Processing*, pp. 216–270. MIT Press, Cambridge, MA, USA (1968)
20. Rector, A., Rogers, J.: Ontological and Practical Issues in Using a Description Logic to Represent Medical Concept Systems: Experience from GALEN. In: *Reasoning Web, Second International Summer School, Tutorial Lectures*, Lisbon, Portugal, September 4–8 2006, pp. 197–231 (2006)
21. Ruttenberg, A., Rees, J., Luciano, J.: Experience Using OWL DL for the Exchange of Biological Pathway Information. In: Proc. of OWL-ED 2005. CEUR, vol. 188 (2005)
22. Sidhu, A., Dillon, T., Chang, E., Sidhu, B.S.: Protein Ontology Development using OWL. In: Proc. of OWL-ED 2005. CEUR, vol. 188 (2005)
23. Sirin, E., Parsia, B.: Pellet: An OWL DL Reasoner. In: Proc. of DL 2004. CEUR, vol. 104 (2004)
24. Soergel, D., Lauser, B., Liang, A., Fisseha, F., Keizer, J., Katz, S.: Reengineering Thesauri for New Applications: The AGROVOC Example. *Journal of Digital Information* 4(4) (2004)
25. Tsarkov, D., Horrocks, I.: FaCT++ Description Logic Reasoner: System Description. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 292–297. Springer, Heidelberg (2006)

# History Matters: Incremental Ontology Reasoning Using Modules

Bernardo Cuenca Grau<sup>1</sup>, Christian Halaschek-Wiener<sup>2</sup>, and Yevgeny Kazakov<sup>1</sup>

<sup>1</sup> The University of Manchester, School of Computer Science, Manchester, M13 9PL, UK

<sup>2</sup> Department of Computer Science, University of Maryland, College Park, MD 20740, USA

**Abstract.** The development of ontologies involves continuous but relatively small modifications. Existing ontology reasoners, however, do not take advantage of the similarities between different versions of an ontology. In this paper, we propose a technique for incremental reasoning—that is, reasoning that reuses information obtained from previous versions of an ontology—based on the notion of a module. Our technique does not depend on a particular reasoning calculus and thus can be used in combination with any reasoner. We have applied our results to incremental classification of OWL DL ontologies and found significant improvement over regular classification time on a set of real-world ontologies.

## 1 Introduction

The design and maintenance of OWL ontologies are highly complex tasks. The support of a reasoner is crucial for detecting modeling errors, which typically manifest themselves as concept unsatisfiability and unintended subsumption relationships.

The development of ontologies involves continuous but relatively small modifications. Even after a number of changes, an ontology and its previous version usually share most of their axioms. Unfortunately, when an ontology evolves, current reasoners do not take advantage of the similarities between the ontology and its previous version. That is, when reasoning over the latest version of an ontology, current reasoners do not reuse existing results already obtained for the previous one and repeat the whole reasoning process. For large and complex ontologies this may require a few minutes, or even a few hours. If the response of the reasoner is too slow, ontology engineers may end up not using the reasoner as often as they would wish. For ontology development and maintenance tasks it is important to detect possible errors as soon as possible; for such a purpose, the reasoner should be executed often and real time response from the reasoner becomes an important issue.

In this paper, we propose a technique for incremental ontology reasoning—that is, reasoning that reuses the results obtained from previous computations. Our technique is based on the notion of a *module* and can be applied to arbitrary queries against ontologies expressed in OWL DL. We focus on a particular kind of modules that exhibit a set of compelling properties and apply our method to incremental classification of OWL DL ontologies. Our techniques do not depend on a particular reasoner or reasoning method and could be easily implemented in any existing prover, such as Pellet,



FaCT++, KAON2 or RACER. Our empirical results using Pellet<sup>1</sup> show substantial performance improvements over regular classification time.

## 2 Preliminaries

We introduce the syntax of the description logic  $\mathcal{SHOIQ}$  [11], which provides the logical underpinning for OWL DL.

A  $\mathcal{SHOIQ}$ -signature is the disjoint union  $\mathbf{S} = \mathbf{R} \uplus \mathbf{C} \uplus \mathbf{I}$  of sets of *atomic roles* (denoted by  $R, S, \dots$ ), *atomic concept* (denoted by  $A, B, \dots$ ) and *nominals* (denoted by  $a, b, c, \dots$ ). A  $\mathcal{SHOIQ}$ -role is either  $R \in \mathbf{R}$  or an *inverse role*  $R^-$  with  $R \in \mathbf{R}$ . We denote by  $\mathbf{Rol}$  the set of  $\mathcal{SHOIQ}$ -roles for the signature  $\mathbf{S}$ . The set  $\mathbf{Con}$  of  $\mathcal{SHOIQ}$ -concepts for  $\mathbf{S}$  is defined by the following grammar:

$$\mathbf{Con} ::= \perp \mid a \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid \exists R.C \mid \geq n S.C$$

where  $a \in \mathbf{I}$ ,  $A \in \mathbf{C}$ ,  $C_{(i)} \in \mathbf{Con}$ ,  $R, S \in \mathbf{Rol}$ , with  $S$  a *simple role*<sup>2</sup> and  $n$  a positive integer. We use the following abbreviations:  $C \sqcup D$  stands for  $\neg(\neg C \sqcap \neg D)$ ;  $\top$  stands for  $\neg \perp$ ;  $\forall R.C$  stands for  $\neg(\exists R.\neg C)$ ; and  $\leq n S.C$  stands for  $\neg(\geq n+1 S.C)$ .

A  $\mathcal{SHOIQ}$  ontology  $\mathcal{O}$  is a finite set of *role inclusion axioms* (RIs)  $R_1 \sqsubseteq R_2$  with  $R_i \in \mathbf{Rol}$ , *transitivity axioms*  $\text{Trans}(R)$  with  $R \in \mathbf{R}$  and *general concept inclusion axioms* (GCIs)  $C_1 \sqsubseteq C_2$  with  $C_i \in \mathbf{Con}$ <sup>3</sup>. The concept definition  $A \equiv C$  is an abbreviation for the two GCIs  $A \sqsubseteq C$  and  $C \sqsubseteq A$ . The signature  $\text{Sig}(\alpha)$  of an axiom  $\alpha$  is the union  $\text{RN}(\alpha) \cup \text{CN}(\alpha) \cup \text{Ind}(\alpha)$  of atomic roles, atomic concepts, and nominals that occur in  $\alpha$ . The signature  $\text{Sig}(\mathcal{O})$  of an ontology  $\mathcal{O}$  is defined analogously.

For the semantics of  $\mathcal{SHOIQ}$ , we refer the interested reader to [11].

## 3 The Challenge for Incremental Reasoning in Ontologies

Consider the medical ontology  $\mathcal{O}^1$  given in Table 1, which consists of three concept definitions D1 – D3 and two inclusion axioms C1 – C2. For exposition, suppose that an ontology engineer in charge of this ontology notices that the definition D1 for the concept `Cystic_Fibrosis` is incomplete and reformulates it by adding the new conjunct  $\exists \text{has\_Origin.Genetic\_Origin}$ . As a result, a new version  $\mathcal{O}^2$  of the ontology is obtained. In order to ensure that no errors have been introduced by this change, the ontology engineer uses a reasoner to classify the new ontology  $\mathcal{O}^2$ .

Table 2 shows some subsumption relationships between atomic concepts in  $\mathcal{O}^1$  and  $\mathcal{O}^2$ , which should be computed for classification. We can see that some of these subsumption relations have changed as a result of a modification in the ontology: axiom  $\alpha_1$  follows from the axioms D3, C2 and D1 in  $\mathcal{O}^1$ , but does not follow from  $\mathcal{O}^2$  anymore since D1 has been modified; in contrast, the subsumption  $\alpha_2$ , which did not follow from  $\mathcal{O}^1$ , is now a consequence of the modified D1, D2 and C1 in  $\mathcal{O}^2$ . Other subsumptions

<sup>1</sup> Pellet Homepage: <http://pellet.owldl.com>

<sup>2</sup> See [11] for a precise definition of simple roles.

<sup>3</sup> Note that ABox assertions  $a : C$  can be expressed in  $\mathcal{SHOIQ}$  using GCIs  $a \sqsubseteq C$ .



**Table 1.** Evolution of a Bio-Medical Ontology  $\mathcal{O}$ 

Original Ontology $\mathcal{O}^1$ :	Modified Ontology $\mathcal{O}^2$ :
<b>D1</b> Cystic_Fibrosis $\equiv$ Fibrosis $\sqcap$ $\exists$ located_In.Pancreas	Cystic_Fibrosis $\equiv$ Fibrosis $\sqcap$ $\exists$ located_In.Pancreas $\sqcap$ $\exists$ has_Origin.Genetic_Origin
<b>D2</b> Genetic_Fibrosis $\equiv$ Fibrosis $\sqcap$ $\exists$ has_Origin.Genetic_Origin	Genetic_Fibrosis $\equiv$ Fibrosis $\sqcap$ $\exists$ has_Origin.Genetic_Origin
<b>D3</b> Pancreatic_Fibrosis $\equiv$ Fibrosis $\sqcap$ Pancreatic_Disorder	Pancreatic_Fibrosis $\equiv$ Fibrosis $\sqcap$ Pancreatic_Disorder
<b>C1</b> Genetic_Fibrosis $\sqsubseteq$ Genetic_Disorder	Genetic_Fibrosis $\sqsubseteq$ Genetic_Disorder
<b>C2</b> Pancreatic_Disorder $\sqsubseteq$ Disorder $\sqcap$ $\exists$ located_In.Pancreas	Pancreatic_Disorder $\sqsubseteq$ Disorder $\sqcap$ $\exists$ located_In.Pancreas
$\Delta\mathcal{O} = \text{diff}(\mathcal{O}^1, \mathcal{O}^2) = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$	
$\Delta^-\mathcal{O} = \text{Cystic\_Fibrosis} \equiv \text{Fibrosis} \sqcap \exists\text{located\_In.Pancreas}$	
$\Delta^+\mathcal{O} = \text{Cystic\_Fibrosis} \equiv \text{Fibrosis} \sqcap \exists\text{located\_In.Pancreas} \sqcap \exists\text{has\_Origin.Genetic\_Origin}$	

**Table 2.** Subsumption Relations Before and After the Change

$\alpha$	Axiom:	$\mathcal{O}^1 \models^? \alpha$ , follows from:	$\mathcal{O}^2 \models^? \alpha$ , follows from:
$\alpha_1$	Pancreatic_Fibrosis $\sqsubseteq$ Cystic_Fibrosis	Yes D3, C2, <b>D1</b>	No —
$\alpha_2$	Cystic_Fibrosis $\sqsubseteq$ Genetic_Disorder	No —	Yes <b>D1</b> , D2, C1
$\alpha_3$	Pancreatic_Fibrosis $\sqsubseteq$ Disorder	Yes D3, C2	Yes D3, C2
$\alpha_4$	Genetic_Fibrosis $\sqsubseteq$ Cystic_Fibrosis	No —	No —

such as  $\alpha_3$  and  $\alpha_4$  did not change:  $\alpha_3$  is a consequence of axioms D3 and C2 which have not been modified;  $\alpha_4$  follows neither from  $\mathcal{O}^1$  nor from  $\mathcal{O}^2$ .

It is reasonable to expect that small changes in ontologies will not affect many subsumption relations. That is, the number of subsumptions that change their entailment status w.r.t. the ontology, like, say,  $\alpha_1$  or  $\alpha_2$  in Table 2, is probably small compared to the number of subsumptions that do not, like  $\alpha_3$  or  $\alpha_4$ . If so, then many (possibly expensive) re-computations can be avoided by reusing the subsumption relations computed for the previous version of the ontology. In order to realize this idea, one has to identify which subsumptions could be affected by a change and which are not.

Suppose we know that a subsumption  $\alpha$  holds in  $\mathcal{O}^1$ . Then we can guarantee that  $\alpha$  still holds in  $\mathcal{O}^2$  provided the axioms from which  $\alpha$  follows in  $\mathcal{O}^1$  have not been modified. For example, in Table 2 the subsumption  $\alpha_3$  is a consequence of axioms D3 and C2, both of which have not been modified in  $\mathcal{O}^2$ . Hence, we can conclude that  $\alpha_3$  holds in  $\mathcal{O}^2$  without performing reasoning over  $\mathcal{O}^2$ . In contrast, this test is not applicable for the subsumption  $\alpha_1$ , since  $\alpha_1$  is a consequence of axioms D3, C2 and D1 in  $\mathcal{O}^1$ , and D1 has been modified in  $\mathcal{O}^2$ . In this case, the status of  $\alpha_1$  in  $\mathcal{O}^2$  has to be computed by other means, e.g. using a reasoner. Thus, the status of every subsumption

relation  $\alpha$  that holds in  $\mathcal{O}^1$  requires re-computation for  $\mathcal{O}^2$  only if in every justification for  $\alpha$  (every minimal subset of  $\mathcal{O}^1$  which implies  $\alpha$ ) some axiom has been modified. This approach is reminiscent of the way Truth Maintenance Systems (TMS) maintain logical dependencies between axioms [6,3]. The notion of justification for an axiom has also been used for pinpointing the axioms responsible for errors in ontologies, such as unsatisfiable concepts and unintended subsumptions [14,13].

The situation is principally different in the case of subsumptions  $\alpha$  that do *not* hold in  $\mathcal{O}^1$ . In this case, if to follow the previous approach, one has to keep track of “evidences” for *non-entailments* of subsumptions in ontologies and verify if at least one such “evidence” for  $\alpha$  in  $\mathcal{O}^1$  can be reused in  $\mathcal{O}^2$ . Here, the “evidence” might be, for example, a (part of a) counter-model for  $\alpha$  in  $\mathcal{O}^1$  that is constructed by tableau-based procedures. Such techniques based on *model caching* have been recently proposed in the context incremental reasoning [8]. These techniques, however, have only been applied so far to additions and deletions of ABox assertions, since changes in general axioms often require considerable modifications of the models. Moreover, such techniques require close interaction with the model construction routine of the tableau reasoner, which precludes their use in arbitrary “off-the-shelf” reasoners without considerable modifications. In particular, these techniques cannot be directly used in reasoners like KAON2, which are not tableaux-based.

We stress that the challenge for incremental ontology reasoning is mainly to maintain non-subsumptions since, in typical ontologies, almost 99% of subsumption relations between atomic concepts do not hold. In other words, the case of axiom  $\alpha_4$  in Figure 2 is likely to be the most one after a change in an ontology.<sup>4</sup>

In this paper we propose an alternative approach for incremental reasoning based on the module-extraction techniques introduced in [2]. Our technique can be used to keep track of “evidences” for *both* subsumptions and non-subsumptions modulo arbitrary changes in ontologies, and works in combination with any DL-reasoner providing for standard reasoning services.

## 4 Modules and Syntactic Locality

In this section we define the notion of a module [2], which underlies our technique for incremental reasoning. We also outline the algorithm proposed in [2] for extracting a particular kind of modules, called *locality-based* modules.

**Definition 1 (Module for an Axiom and a Signature).** *Let  $\mathcal{O}$  be an ontology and  $\mathcal{O}_1 \subseteq \mathcal{O}$  is a (possibly empty) subset of axioms in  $\mathcal{O}$ . We say that  $\mathcal{O}_1$  is a module for an axiom  $\alpha$  in  $\mathcal{O}$  (or short, an  $\alpha$ -module in  $\mathcal{O}$ ) if:  $\mathcal{O}_1 \models \alpha$  iff  $\mathcal{O} \models \alpha$ .*

*We say that  $\mathcal{O}_1$  is a module for a signature  $\mathbf{S}$  if for every axiom  $\alpha$  with  $\text{Sig}(\alpha) \subseteq \mathbf{S}$ , we have that  $\mathcal{O}_1$  is a module for  $\alpha$  in  $\mathcal{O}$ .*

Intuitively, a module for an axiom  $\alpha$  in an ontology  $\mathcal{O}$  is a subset  $\mathcal{O}_1$  of  $\mathcal{O}$  which contains the axioms that are “relevant” for  $\alpha$  in  $\mathcal{O}$ , in the sense that  $\mathcal{O}$  implies  $\alpha$  if and only if  $\mathcal{O}_1$  implies  $\alpha$ . In case  $\mathcal{O}$  implies  $\alpha$ , then every module  $\mathcal{O}_1$  for  $\alpha$  should contain

<sup>4</sup> In Section 6 we provide empirical evidences confirming our conjectures.

at least one justification for  $\alpha$  (that is, a minimal set of axioms which imply  $\alpha$ ). In case  $\mathcal{O}$  does not imply  $\alpha$  (that is, there are no justifications for  $\alpha$ ),  $\mathcal{O}_1$  can be any subset of  $\mathcal{O}$ . Hence, knowing all the justifications for  $\alpha$  in  $\mathcal{O}$  is sufficient for identifying all modules for  $\alpha$  in  $\mathcal{O}$ .

The notion of module *for a signature* has been introduced in [2]. Intuitively, a module for a signature is a subset of the ontology that is a module for every axiom constructed over this signature. An algorithm for extracting modules based on a notion of syntactic locality was proposed in [2], and it was empirically verified that this algorithm extracts reasonably small modules in existing ontologies.

**Definition 2 (Syntactic Locality for SHOIQ).** *Let  $\mathbf{S}$  be a signature. The following grammar recursively defines two sets of concepts  $\mathbf{Con}^0(\mathbf{S})$  and  $\mathbf{Con}^\Delta(\mathbf{S})$  for  $\mathbf{S}$ :*

$$\begin{aligned} \mathbf{Con}^0(\mathbf{S}) &::= A^\emptyset \mid (\neg C^\Delta) \mid (C^\emptyset \sqcap C) \mid (C \sqcap C^\emptyset) \\ &\quad \mid (\exists R^\emptyset.C) \mid (\exists R.C^\emptyset) \mid (\geq n R^\emptyset.C) \mid (\geq n R.C^\emptyset). \\ \mathbf{Con}^\Delta(\mathbf{S}) &::= (\neg C^\emptyset) \mid (C_1^\Delta \sqcap C_2^\Delta). \end{aligned}$$

where  $A^\emptyset \notin \mathbf{S}$  is an atomic concept,  $R^\emptyset$  is (possibly inverse of) an atomic role  $r^\emptyset \notin \mathbf{S}$ ,  $C$  is any concept,  $R$  is any role, and  $C^\emptyset \in \mathbf{Con}^0(\mathbf{S})$ ,  $C_i^\Delta \in \mathbf{Con}^\Delta(\mathbf{S})$ ,  $i = 1, 2$ .

An axiom  $\alpha$  is local w.r.t.  $\mathbf{S}$  if it is of one of the following forms: (1)  $R^\emptyset \sqsubseteq R$ , or (2)  $\text{Trans}(R^\emptyset)$ , or (3)  $C^\emptyset \sqsubseteq C$  or (4)  $C \sqsubseteq C^\Delta$ .<sup>5</sup>

Intuitively, an axiom  $\alpha$  is syntactically local w.r.t.  $\mathbf{S}$  if, by simple syntactical simplifications, one can demonstrate that  $\alpha$  is true in every interpretation  $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$  in which concept and atomic roles not from  $\mathbf{S}$  are interpreted with the empty set. For example, the axiom D2 from Table 1 is local w.r.t.  $\mathbf{S} = \{\text{Fibrosis}, \text{has\_Origin}\}$ : if we interpret the remaining symbols in this axiom with the empty set, we obtain a model of the axiom, independently of the interpretation of the symbols in  $\mathbf{S}$ .

$$\overbrace{\text{Genetic\_Fibrosis}}^\emptyset \equiv \text{Fibrosis} \sqcap \underbrace{\exists \text{has\_Origin. Genetic\_Origin}}_\emptyset$$

If an ontology  $\mathcal{O}$  can be partitioned as  $\mathcal{O} = \mathcal{O}_1 \cup \mathcal{O}_s$  such that every axiom in  $\mathcal{O}_s$  is syntactically local w.r.t.  $\mathbf{S} \cup \text{Sig}(\mathcal{O}_1)$ , then  $\mathcal{O}_1$  is a module for  $\mathbf{S}$  in  $\mathcal{O}$  [2]. Algorithm 1 extracts a module  $\mathcal{O}_1$  for a signature  $\mathbf{S}$  from an ontology  $\mathcal{O}$  using this property. The procedure first initializes  $\mathcal{O}_1$  to the empty set and then iteratively moves to  $\mathcal{O}_1$  those axioms  $\alpha$  from  $\mathcal{O}$  that are not local w.r.t.  $\mathbf{S} \cup \text{Sig}(\mathcal{O}_1)$  until all such axioms have been moved. We assume that  $\text{s\_local}(\alpha, \mathbf{S})$  tests for syntactic locality of an axiom  $\alpha$  w.r.t. signature  $\mathbf{S}$  according to Definition 2. In Table 3 we provide a trace of Algorithm 1 for the input ontology  $\mathcal{O}^1$  in Table 1 and signature  $\mathbf{S} = \{\text{Pancreatic\_Fibrosis}\}$ .

**Proposition 1 (Correctness of Algorithm 1 (see [2]) for details).** *Given an SHOIQ ontology  $\mathcal{O}$  and a signature  $\mathbf{S}$ , Algorithm 1 terminates in polynomial time in the size of  $\mathcal{O}$  and returns a module  $\mathcal{O}_1$  for  $\mathbf{S}$  in  $\mathcal{O}$ .*

<sup>5</sup> Recall that  $\forall R.C$ ,  $(\leq n R.C)$  and  $C_1 \sqcup C_2$  are expressed using the other constructors, so they can be used in local axioms as well.

**Table 3.** An algorithm for extracting syntactic locality-based modules from ontologies

Algorithm 1 $\text{extract\_module}(\mathcal{O}, \mathbf{S})$	A sample trace for the Algorithm 1 for $\mathcal{O} = \mathcal{O}^1$ from Table 1 and $\mathbf{S} = \{\text{Pancreatic\_Fibrosis}\}$ :			
Input:	$\mathcal{O}_1$	$\mathcal{O}_2$	New $X \in \mathbf{S} \cup \text{Sig}(\mathcal{O}_1)$	$\alpha$ loc?
$\mathcal{O}$ : ontology	1	–	Pancreatic_Fibrosis	D3   No
$\mathbf{S}$ : signature		D1, D2, D3, C1, C2		
Output:	2	D3	Fibrosis, Pancreatic_Disorder	D1   Yes
$\mathcal{O}_1$ : a module for $\mathbf{S}$ in $\mathcal{O}$		D1, D2, C1, C2		
1: $\mathcal{O}_1 \leftarrow \emptyset$ $\mathcal{O}_2 \leftarrow \mathcal{O}$	3	D3	D2, C1, C2	D2   Yes
2: <b>while not</b> $\text{empty}(\mathcal{O}_2)$ <b>do</b>	4	D3	C1, C2	C1   Yes
3: $\alpha \leftarrow \text{select\_axiom}(\mathcal{O}_2)$	5	D3	C2	C2   No
4: <b>if</b> $\text{s.local}(\alpha, \mathbf{S} \cup \text{Sig}(\mathcal{O}_1))$	6	D3, C2	D1, D2, C1,	D1   Yes
<b>then</b>	7	D3, C2	D2, C1	D2   Yes
5: $\mathcal{O}_2 \leftarrow \mathcal{O}_2 \setminus \{\alpha\}$	8	D3, C2	C1	C1   Yes
6: <b>else</b>	9	D3, C2	–	–
7: $\mathcal{O}_1 \leftarrow \mathcal{O}_1 \cup \{\alpha\}$				
8: $\mathcal{O}_2 \leftarrow \mathcal{O} \setminus \mathcal{O}_1$				
9: <b>end if</b>				
10: <b>end while</b>				
11: <b>return</b> $\mathcal{O}_1$				

In order to extract a module for an axiom  $\alpha$  in  $\mathcal{O}$  it is sufficient to run Algorithm 1 for  $\mathbf{S} = \text{Sig}(\alpha)$ . However, when  $\alpha$  is a subsumption between atomic concepts,  $\top$  or  $\perp$ , it suffices to extract a module only for  $\mathbf{S} = \text{Sig}(X)$ , as given in the following proposition.

**Proposition 2** (see [2] for details). *Let  $\mathcal{O}$  be a SHOIQ ontology,  $X, Y \in \text{CN}(\mathcal{O}) \cup \{\top\} \cup \{\perp\}$ , and  $\mathcal{O}_X$  the output of Algorithm 1 for input  $\mathcal{O}$  and  $\mathbf{S} = \text{Sig}(X)$ . Then  $\mathcal{O}_X$  is a module in  $\mathcal{O}$  for  $\alpha = (X \sqsubseteq Y)$ .*

Finally, we point out that the modules extracted using Algorithm 1 are not necessary minimal ones. That is, if  $\mathcal{O} \models \alpha$ , the computed module for  $\alpha$  might be a strict superset of a justification for  $\alpha$  in  $\mathcal{O}$ , and if  $\mathcal{O} \not\models \alpha$  then the module for  $\text{Sig}(\alpha)$  might not necessarily be the empty set. In fact, if  $\alpha$  is not a tautology, computing a minimal module for  $\alpha$  in  $\mathcal{O}$  is at least as hard as checking whether  $\mathcal{O} \not\models \alpha$  since  $\mathcal{O} \models \alpha$  iff the minimal module for  $\alpha$  is empty. The last problem is computationally expensive for many ontology languages, including OWL DL. The advantage of the module-extraction algorithm described in this section is that, on the one hand, it runs in polynomial and, on the other hand, it still generates reasonably small modules.

## 5 Incremental Classification Using Locality-Based Modules

In this section we show how to use then notion of module for incremental reasoning over ontologies. First, we outline the general idea behind using modules for incrementally maintaining (non)entailment of axioms and then describe an algorithm for incremental classification of ontologies using locality-based modules, as described in Section 4.

The following proposition, which is a simple consequence of Definition 1, provides the basic property underlying incremental reasoning using modules:

**Proposition 3.** *Let  $\mathcal{O}^1, \mathcal{O}^2$  be ontologies,  $\alpha$  an axiom, and  $\mathcal{O}_\alpha^1, \mathcal{O}_\alpha^2$  respectively modules for  $\alpha$  in  $\mathcal{O}^1$  and  $\mathcal{O}^2$ . Then:*

1. *If  $\mathcal{O}^1 \models \alpha$  and  $\mathcal{O}_\alpha^1 \subseteq \mathcal{O}^2$ , then  $\mathcal{O}^2 \models \alpha$*
2. *If  $\mathcal{O}^1 \not\models \alpha$  and  $\mathcal{O}_\alpha^2 \subseteq \mathcal{O}^1$ , then  $\mathcal{O}^2 \not\models \alpha$*

Proposition 3 suggests that, in order to test if the entailment of an axiom  $\alpha$  has not been affected by a change  $\mathcal{O}^1 \Rightarrow \mathcal{O}^2$ , it is sufficient to compute, depending on whether  $\mathcal{O}^1 \models \alpha$  or  $\mathcal{O}^1 \not\models \alpha$ , a module  $\mathcal{O}_\alpha^1$  for  $\alpha$  in  $\mathcal{O}^1$ , or a module  $\mathcal{O}_\alpha^2$  for  $\alpha$  in  $\mathcal{O}^2$  respectively. If the change does not involve any of the axioms in the module, then the status of the entailment of  $\alpha$  also does not change. The converse of this is not necessarily true: even if the corresponding module has been modified, the status of  $\alpha$  might still remain unaffected. For example, the axiom  $\alpha = (\text{Cystic\_Fibrosis} \sqsubseteq \text{Fibrosis})$  follows from D1 both before and after the change, even though D1 has been modified. In such a case, the status of  $\alpha$  w.r.t.  $\mathcal{O}^2$  should be verified using the reasoner. The use of modules, however, is also valuable in this situation: instead of checking if  $\alpha$  follows from  $\mathcal{O}^2$ , one could equivalently check if  $\alpha$  follows from the (hopefully much smaller) module  $\mathcal{O}_\alpha^2$ .

Therefore, the use of modules provides two compelling advantages for incremental reasoning: first, the computation of a given query may be avoided and the answer can be simply reused from a previous test; second, even if the query needs to be performed, the use of modules allows for filtering out irrelevant axioms and reduces the search space.

Note that the sizes of modules  $\mathcal{O}_\alpha^1$  and  $\mathcal{O}_\alpha^2$  have a direct impact on the quality of the incremental entailment test for  $\alpha$ . The smaller the modules, the more likely it is that they do not contain the modified axioms. Nevertheless, as pointed out in Section 4, computing a smallest possible module is computationally expensive: it is at least as hard as just checking whether  $\mathcal{O}^1 \models \alpha$  (respectively  $\mathcal{O}^2 \models \alpha$ ). Thus, there is a trade-off between the complexity of computing a module on the one hand, and its usefulness for incremental reasoning on the other hand. Intuitively, the smaller the module, the more useful and the harder it is to compute. We demonstrate empirically that Algorithm 1 computes small enough modules to be useful for incremental reasoning.

In the remainder of this section we apply the general idea for incremental reasoning sketched above for incremental classification of ontologies using the module-extraction procedure given by Algorithm 1. Classification of an ontology  $\mathcal{O}$  amounts to computing *subsumption relations*  $X \sqsubseteq Y$  where  $X$  and  $Y$  range over all atomic concepts from  $\mathcal{O}$ ,  $\perp$ , and  $\top$ . The relations are non-trivial when  $X \in \text{CN}(\mathcal{O}) \cup \{\top\}$  and  $Y \in \text{CN}(\mathcal{O}) \cup \{\perp\}$ . As shown in Proposition 2, in order to check incrementally a subsumption relation  $\alpha = (X \sqsubseteq Y)$ , it is sufficient to keep track of the modules  $\mathcal{O}_X$  for  $\text{Sig}(X)$  in  $\mathcal{O}$ .

Consider the ontologies  $\mathcal{O}^1$  and  $\mathcal{O}^2$  in Table 1 and the axioms  $\alpha_1$ – $\alpha_4$  in Table 2. Each of these axioms is of the form  $\alpha = (X \sqsubseteq Y)$ , with  $X$  and  $Y$  atomic concepts. Table 4 provides the locality-based modules for  $\alpha_1$ – $\alpha_4$  in  $\mathcal{O}^1$  and in  $\mathcal{O}^2$  computed using Algorithm 1. Note that the modules are not minimal: in our case, they are strict supersets of the actual minimal modules from Table 2 where the additional axioms are underlined. The modules for axioms  $\alpha_1$ – $\alpha_3$  have been changed, whereas the module for the axiom  $\alpha_4$  has remained unchanged. Hence, the sufficient test for preservation

**Table 4.** Modules For Subsumptions and Concept Names in Ontologies from Table 1

$\alpha$	Axiom $X \sqsubseteq Y$ :	$\mathcal{O}_X^1$	$\mathcal{O}_X^2$	$X$	$\mathcal{O}_X^1$	$\mathcal{O}_X^2$
$\alpha_1$	Pancreatic_Fibrosis $\sqsubseteq$ Cystic_Fibrosis	D3,C2, <b>D1</b>	<u>D3,C2</u>	Cystic_Fibrosis Fibrosis Pancreas	<b>D1</b> $\emptyset$ $\emptyset$	<b>D1,D2,C1</b> $\emptyset$ $\emptyset$
$\alpha_2$	Cystic_Fibrosis $\sqsubseteq$ Genetic_Disorder	<b>D1</b>	<b>D1,D2,C1</b>	Genetic_Fibrosis Genetic-Origin	C1 $\emptyset$	C1 $\emptyset$
$\alpha_3$	Pancreatic_Fibrosis $\sqsubseteq$ Disorder	D3,C2, <b>D1</b>	D3,C2	Pancreatic_Fibrosis Pancreatic_Disorder Genetic_Disorder	D3,C2, <b>D1</b> C2 C1	D3,C2 C2 C1
$\alpha_4$	Genetic_Fibrosis $\sqsubseteq$ Cystic_Fibrosis	<u>C1</u>	<u>C1</u>	Disorder $\top$	C2 $\emptyset$	C2 $\emptyset$

of (non)subsumptions using modules gave us only one “false positive” for subsumption  $\alpha_3$ , where the subsumption relation did not change, but the modules have been modified.

The right part of Table 4 provides the full picture on the modules and their changes for our example ontology from Table 1. The only modules that have been changed are the ones for  $X = \text{Cystic\_Fibrosis}$  and  $X = \text{Pancreatic\_Fibrosis}$ , where for the first module axiom D1 has been changed, and for in the second module axiom D1 has been removed. Applying Proposition 2 and Proposition 3 we can conclude that every subsumption that disappears as a the result of the change should be either of the form  $\alpha = (\text{Cystic\_Fibrosis} \sqsubseteq Y)$  or  $\alpha = (\text{Pancreatic\_Fibrosis} \sqsubseteq Y)$ , and every subsumption that can appear should be of the form  $\alpha = (\text{Cystic\_Fibrosis} \sqsubseteq Y)$ .

Algorithm 2 outlines an incremental classification procedure based on the ideas just discussed. Given an ontology  $\mathcal{O}^1$  and a change  $\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$  consisting of the sets of removed and added axioms, the algorithm computes the subsumption partial order  $\sqsubseteq_2$  for the resulting ontology  $\mathcal{O}^2 = (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$  by reusing the one  $\sqsubseteq_1$  already computed for  $\mathcal{O}^1$ . In order to perform this operation, the algorithm internally maintains the modules  $\mathcal{O}_X^1$  and  $\mathcal{O}_X^2$  for every atomic concept or the top concept  $X$ . We will show that maintaining these additional modules does not involve a significant overhead in practice. The algorithm consists of the following phases:

1. *Process the new symbols (lines 2-6)*: The modules  $\mathcal{O}_X^1$  and the subsumption partial order  $\sqsubseteq_1$  for  $\mathcal{O}^1$  are extended for every newly introduced atomic concept  $A$ . The module for  $A$ , about which nothing has been said yet, is equivalent to the module for the empty signature—that, is the module for  $\top$ . Thus, we have: (i)  $\mathcal{O}_A^1 = \mathcal{O}_\top^1$ , (ii)  $\mathcal{O}^1 \models A \sqsubseteq Y$  iff  $\mathcal{O}^1 \models \top \sqsubseteq Y$ , and (iii)  $\mathcal{O}^1 \models X \sqsubseteq A$  iff  $\mathcal{O}^1 \models X \sqsubseteq \perp$ .
2. *Identifying the affected modules (lines 7-19)*: The sets  $M^-$  and  $M^+$  contain those  $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$  for which the corresponding modules must be modified by removing and/or adding axioms. If  $\alpha$  removed from  $\mathcal{O}^1$  is non-local w.r.t.  $\text{Sig}(\mathcal{O}_X^1)$  then at least  $\alpha$  should be removed from  $\mathcal{O}_X^1$ . If  $\alpha$  is added to  $\mathcal{O}^1$  and is non-local w.r.t.  $\text{Sig}(\mathcal{O}_X^1)$ , then the module  $\mathcal{O}_X^1$  needs to be extended at least with  $\alpha$ .
3. *Computing new modules and subsumptions (lines 20-34)*: The affected modules found in the previous phase are re-extracted and those that are not are just copied

---

**Algorithm 2.**  $\text{inc\_classify}(\mathcal{O}^1, \Delta\mathcal{O}, \sqsubseteq_1, X \rightarrow \mathcal{O}_X^1)$ 


---

**Input:**

- $\mathcal{O}^1$ : an ontology
- $\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$ : removed / added axioms
- $\sqsubseteq_1$ : subsumption relations in  $\mathcal{O}^1$
- $X \rightarrow \mathcal{O}_X^1$ : a module for every  $X \in \text{CN}(\mathcal{O}^1) \cup \{\top\}$

**Output:**

- $\mathcal{O}^2$ : the result of applying the change  $\Delta\mathcal{O}$  to  $\mathcal{O}^1$
  - $\sqsubseteq_2$ : subsumption relations in  $\mathcal{O}^2$
  - $X \rightarrow \mathcal{O}_X^2$ : a module for every  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$
- 

```

1:  $\mathcal{O}^2 \leftarrow (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ 
2: for each  $A \in \text{CN}(\mathcal{O}^2) \setminus \text{CN}(\mathcal{O}^1)$  do
3:    $\mathcal{O}_A^1 \leftarrow \mathcal{O}_A^1$ 
4:   for each  $\top \sqsubseteq_1 Y$  do  $A \sqsubseteq_1 Y \leftarrow \text{true}$ 
5:   for each  $X \sqsubseteq_1 \perp$  do  $X \sqsubseteq_1 A \leftarrow \text{true}$ 
6: end for
7:  $M^- \leftarrow \emptyset$   $M^+ \leftarrow \emptyset$ 
8: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
9:   for each  $\alpha \in \Delta^-\mathcal{O}$  do
10:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
11:       $M^- \leftarrow M^- \cup \{X\}$ 
12:    end if
13:  end for
14:  for each  $\alpha \in \Delta^+\mathcal{O}$  do
15:    if not  $\text{s\_local}(\alpha, \text{Sig}(\mathcal{O}_X^1))$  then
16:       $M^+ \leftarrow M^+ \cup \{X\}$ 
17:    end if
18:  end for
19: end for
20: for each  $X \in \text{CN}(\mathcal{O}^2) \cup \{\top\}$  do
21:  if  $X \in M^- \cup M^+$  then
22:     $\mathcal{O}_X^2 \leftarrow \text{extract\_module}(\text{Sig}(X), \mathcal{O}^2)$ 
23:  else
24:     $\mathcal{O}_X^2 \leftarrow \mathcal{O}_X^1$ 
25:  end if
26:  for each  $Y \in \text{CN}(\mathcal{O}^2) \cup \{\perp\}$  do
27:    if  $(X \in M^- \text{ and } X \sqsubseteq_1 Y)$  or
28:       $(X \in M^+ \text{ and } X \sqsupseteq_1 Y)$  then
29:       $X \sqsubseteq_2 Y \leftarrow \text{test}(\mathcal{O}_X^2 \models X \sqsubseteq Y)$ 
30:    else
31:       $X \sqsubseteq_2 Y \leftarrow X \sqsubseteq_1 Y$ 
32:    end if
33:  end for
34: end for
35: return  $\mathcal{O}^2, \sqsubseteq_2, X \rightarrow \mathcal{O}_X^2$ 

```

---

(lines 21-25). Then, every subsumption  $X \sqsubseteq Y$ , using Proposition 3, is either re-computed against the module  $\mathcal{O}_X^2$ , or is reused from  $\mathcal{O}^1$  (lines 26-33).

In Algorithm 5, the procedure `extract_module(S, O)` refers to Algorithm 1 in Section 4. The procedure `test(O ⊨ X ⊆ Y)` uses a reasoner to check if  $\mathcal{O}$  entails the subsumption  $X \sqsubseteq Y$ . The correctness of the algorithm is easy to prove using Proposition 2 and Proposition 3.

It is worth emphasizing that, in our algorithm, the reasoner is only used as a black box to answer subsumption queries; this provides two important advantages: on the one hand, the internals of the reasoner need not be modified and, on the other hand, *any* sound and complete reasoner for OWL DL can be plugged in, independently of the reasoning technique it is based on (e.g. tableaux or resolution).

To conclude, we illustrate the execution of Algorithm 5 on the ontologies  $\mathcal{O}^1, \mathcal{O}^2$  in Table 1, where the sets  $\Delta^-\mathcal{O}$  and  $\Delta^+\mathcal{O}$  of removed and added axioms for our example are given in the lower part of Table 1. In our case,  $\mathcal{O}^2$  doesn't introduce new atomic concepts w.r.t.  $\mathcal{O}^1$ . Thus, Phase 1 in Algorithm 2 can be skipped. The sets  $M^-, M^+$  computed in Phase 2 are as follows:  $M^- = \{\text{Cystic\_Fibrosis}, \text{Pancreatic\_Fibrosis}\}$  and  $M^+ = \{\text{Cystic\_Fibrosis}\}$  since the axiom in  $\Delta^-\mathcal{O}$  (see Table 1) is not syntactically local w.r.t. the signature of the module in  $\mathcal{O}^1$  for `Cystic_Fibrosis` and `Pancreatic_Fibrosis`; analogously, the axiom in  $\Delta^+\mathcal{O}$  is non-local w.r.t. the signature of the module in  $\mathcal{O}^2$  for `Cystic_Fibrosis`. In Phase 3, the modules for `Cystic_Fibrosis` and `Pancreatic_Fibrosis` are re-computed. In the former module, the algorithm recomputes only the subsumption relations between `Cystic_Fibrosis` and `Pancreatic_Fibrosis` and their subsumers in  $\mathcal{O}^1$ ; in the latter one, the only the subsumption relations between the non-subsumers of `Cystic_Fibrosis` in  $\mathcal{O}^1$  are computed.

## 6 Empirical Evaluation

We have implemented Algorithm 2 and used the OWL reasoner Pellet for evaluation. Our implementation is, however, independent from Pellet, and our results intend to determine the usefulness of our approach for optimizing any reasoner. Our system implements a slightly more simplistic procedure than the one in Algorithm 2; in particular, once the affected modules have been identified, our implementation simply reclassifies the union of these modules using Pellet to determine the new subsumption relations, instead of using the procedure described in lines 20-34 of Algorithm 2.

As a test suite, we have selected a set of well-known ontologies that are currently being developed. NCI<sup>6</sup>, and the Gene Ontology<sup>7</sup> are expressed in a simple fragment of OWL DL. In contrast, GALEN<sup>8</sup>, and NASA's SWEET ontology<sup>9</sup> are written in a more expressive language. Table 5 includes their expressivity, number of atomic concepts and axioms, total classification time in Pellet, and the percentage of possible subsumption relations that actually hold between atomic concepts. Note that for large ontologies,

<sup>6</sup> <http://www.mindswap.org/2003/CancerOntology/nciOncology.owl>

<sup>7</sup> <http://www.geneontology.org>

<sup>8</sup> [http://www.openclinical.org/prj\\_galen.html](http://www.openclinical.org/prj_galen.html)

<sup>9</sup> <http://sweet.jpl.nasa.gov/ontology/>



**Table 5.** Test suite ontologies

Ontology	Logic	# Concept Names	# Axioms	Class. Time (s.)	% Subs	Init. Mod. Extract (s.)	Mod. Size (Avg/Max)	Non-Loc. Axioms
SWEET	<i>SHOIF</i>	1400	2573	3.6	0.37	1.05	76 / 420	28
Galen	<i>SHF</i>	2749	4529	15.7	0.37	4.8	75 / 530	0
GO	<i>EL</i>	22357	34980	63	0.04	69.6	17.6 / 161	0
NCI	<i>EL</i>	27772	46940	41.1	0.03	76.5	28.9 / 436	0

over 99% of subsumption relations do not hold. Table 5 also shows the average time to extract the modules for all atomic concepts, as well as the average and maximum size of these modules (in terms of the number of axioms). Even if the initial module extraction may introduce overhead, we argue that this “startup-cost” is bearable since the set of all modules needs only be computed once. We observe that, in general, the modules are very small relative to the size of the ontology.

We have performed the following experiment for each ontology: for various numbers  $n$ , we have 1) removed  $n$  random axioms; 2) classified the resulting ontology using Pellet; then, we have repeated the following two steps 50 times: 3) extracted the minimal locality-based module for each atomic concept, 4) removed an additional  $n$  axioms, added back the previously removed  $n$  axioms, and reclassified the ontology using our incremental algorithm. Our goal is to simulate the ontology evolution process where  $n$  axioms are changed (which can be viewed as a simultaneous deletion and addition); all results have been gathered during step 4) of the experiment. We considered different types of axioms, namely concept definitions, GCIs and role axioms.

Table 6 summarizes the results of the experiments for  $n = 2$ . Columns 1 and 2 detail the number of affected modules and their total size respectively. It can be observed that, in general, only a very small number of the modules are affected for a given update. Column 3 provides the total time to locate and re-extract the affected modules; Column 4 shows the reclassification time for all the affected modules after they have been re-extracted. In all cases, the average time is significantly smaller than standard re-classification. It can be observed that, in the case of Galen, the maximum time to classify the affected modules actually takes longer than classifying the entire ontology. While unexpected, this is likely caused as traditional classification optimizations (e.g., model merging, top-bottom search, etc.) are not as effective, due to affected modules

**Table 6.** Results for varying update sizes for class and role axioms. Time in seconds.

	$n$	1: # Mod. Affected (Av / Mx)	2: # Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: # New (Non)Sub. (Av / Mx)	7: # Mod. (Non)Sub (Av / Mx)
NCI	2	67 / 936	545 / 3025	.81 / 5.4	.21 / 1.2	1.03 / 6.77	54 / 1268	17 / 348
SWEET	2	36.9 / 300	281 / 857	.097 / .929	.182 / 1.4	.280 / 2.3	39 / 686	20.1 / 255
Galen	2	134 / 1045	1003 / 2907	.833 / 3.6	2.8 / 13	3.6 / 16.5	111 / 1594	17 / 158
GO	1	39.2 / 1513	127 / 1896	.24 / 1.4	.05 / .47	.29 / 1.5	69 / 2964	33 / 1499
GO	2	46 / 891	216 / 1383	.5 / 2.8	.07 / .43	.57 / 3.2	51 / 1079	26 / 775
GO	4	97 / 1339	474 / 3021	1.4 / 10.1	.25 / 3.2	1.7 / 13.4	94 / 1291	44 / 1034

**Table 7.** Results for varying update sizes for role axiom changes only. Time in seconds.

	$n$	1: # Mod. Affected (Av / Mx)	2: # Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: # New (Non)Sub. (Av / Mx)	7: # Mod. (Non)Sub (Av / Mx)
NCI	2	2274 / 10217	12161 / 29091	25.7 / 60.4	10.4 / 30.8	36.2 / 91.3	0 / 0	0 / 0
SWEET	2	116 / 296	411 / 956	.42 / .93	.6 / 1.4	1.03 / 2.33	.56 / 28	.28 / 14
Galen	2	524 / 1906	1813 / 3780	2.1 / 4.7	6.5 / 15.6	8.6 / 20.4	3.3 / 82	2.5 / 37

**Table 8.** Results for varying update sizes for concept axiom changes only. Time in seconds.

	$n$	1: # Mod. Affected (Av / Mx)	2: # Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: # New (Non)Sub. (Av / Mx)	7: # Mod. (Non)Sub (Av / Mx)
NCI	2	33 / 847	396 / 5387	.59 / 8.7	.15 / 2.6	.75 / 11.4	67 / 2228	20 / 610
SWEET	2	15.2 / 243	276 / 800	.02 / .07	.07 / .65	.095 / .732	31 / 553	12 / 241
Galen	2	131 / 1463	913 / 3397	.84 / 4.5	2.6 / 15.4	3.4 / 19.5	69 / 4323	42 / 1178

containing a subset of the original axioms; therefore, additional subsumption checks have to be performed. We note, however, that on average this does not occur. Column 5 presents the total time to update the modules, load them into the reasoner, and re-classify them; it can be seen that this outperforms reclassifying from scratch. For future work, we plan to more tightly integrate the approach into Pellet, as this will avoid the additional overhead attributed to loading the affected modules into the reasoner for classification. Column 6 shows the number of new subsumption and non-subsumption relations (i.e., the sum) for each ontology, and column 7 provides the average number of modules which have a new subsumption or non-subsumption after a change. The number of new (non) subsumptions is very small, which supports our initial hypothesis that changes do not typically affect a large portion of the original ontology. In the case of SWEET, the ratio of modules with new (non)subsumptions is relatively high when compared to the average number of modules affected; specifically in these cases, almost 50% of the affected modules actually contains a new subsumption/non-subsumption relation after the update. This empirically demonstrates that locality-based modules can be very effective for maintaining (non)subsumptions relations as the underlying ontology changes. Finally, the last two rows of the Table show the results for  $n = 1, 2, 4$  in the case of the Gene Ontology. These results suggests that incremental classification time may grow linearly with the number of modified axioms; similar behavior can be observed for the remaining ontologies.

Table 7 considers the particular case of changes to role axioms only<sup>10</sup>. As shown in Table 7, for SWEET the results are comparable to those presented in Table 6. For NCI and Galen, changes in role axioms do have a more substantial impact.

The particular case of changes to concept axioms only is provided in Table 8<sup>11</sup>. It can be observed that the are much better than those when only role axiom changes are

<sup>10</sup> GO has not been included in Table 7 as it only contains one role axiom.

<sup>11</sup> Again GO has not been included in Table 8 as it only contains one role axiom.

performed. These results confirm that role axioms may cause larger effects than changes in concept definitions.

## 7 Related Work

While there has been substantial work on optimizing reasoning services for description logics (see [10] for an overview), the topic of reasoning through evolving DL knowledge bases remains relatively unaddressed. Notable exceptions include [7,8,9,12]; these papers, however, investigate the problem of incremental reasoning using model-caching techniques in application scenarios that involve changes *only* in the ABox.

There has been substantial work on incremental query and view maintenance in databases (e.g., [11,15,16]) and rule-based systems (e.g., Datalog [4,5]). While related, our work addresses a more expressive formalism; further, traditionally in database systems the problem of incremental maintenance is considered with respect to data (corresponding to DL ABoxes) and not with respect to the database schema (corresponding to DL TBoxes). Our technique, however, focuses on schema reasoning.

There has additionally been extensive work in Truth Maintenance Systems (TMSs) for logical theories (e.g., [3,6]). As pointed out in Section 3 a justification-based approach would be advantageous for incremental classification only if the number of positive subsumptions was larger than the number of non-subsumptions; that is, if most of the formulas the justifications keep track of were provable. This is, however, not the case, as typically there are far more non-subsumptions than subsumptions. Additionally, a TMS system designed to support non-subsumptions (e.g., by caching models) would most likely be impractical due to the potentially large size of these models and substantial modifications likely to be caused by changes in general axioms; however, in our approach, maintaining locality-based modules introduces limited overhead. Finally, the representation language in practical TMSs is mostly propositional logic, whereas we focus on much more expressive languages.

## 8 Conclusion

We have proposed a general technique for incremental reasoning under arbitrary changes in an ontology. We have used locality-based modules due to their compelling properties and applied our method to incremental classification of OWL DL ontologies.

For ontology development, it is desirable to re-classify the ontology after a small number of changes. In this scenario, our results are very promising. Incremental classification using modules is nearly real-time for almost all ontologies and therefore the reasoner could be working transparently to the user in the background without slowing down the editing of the ontology. There are, however, some disadvantages of our approach. First, there are cases where a change which does not affect the concept hierarchy, affects a large number of modules; second, for complex ontologies including nominals, such as the Wine ontology, the modules can be large; third classifying a (large enough) fragment might be more expensive than classifying the whole ontology. In most cases, however, our incremental approach provides a substantial speed-up w.r.t.

regular classification. For future work, we are planning to exploit modules for incremental ABox reasoning tasks, such as query answering.

*Acknowledgements.* This work is supported by the EU Project TONES (Thinking ONtologieS) ref:IST-007603 and by the EPSRC Project REOL (Reasoning in Expressive Ontology Languages) ref:EP/C537211/1. We would like to acknowledge Evren Sirin and Mike Smith for their contributions to the empirical evaluation presented here.

## References

1. Blakeley, J.A., Larson, P.-A., Tompa, F.W.: Efficiently updating materialized views. In: Proc. of SIGMOD 1986. ACM SIGMOD International Conference on Management of Data, pp. 61–71. ACM Press, New York (1986)
2. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Just the right amount: Extracting modules from ontologies. In: Proc. of WWW2007 (2007)
3. Kleer, J.d.: An assumption-based TMS. *Artif. Intell.* 28(2), 127–162 (1986)
4. Dong, G., Su, J., Topor, R.W.: Nonrecursive incremental evaluation of datalog queries. *Annals of Mathematics and Artificial Intelligence* 14(2-4) (1995)
5. Dong, G., Topor, R.W.: Incremental evaluation of datalog queries. In: Proc. of the 4th Int. Conference on Database Theory (1992)
6. Doyle, J.: A truth maintenance system. *Readings in nonmonotonic reasoning* , 259–279 (1987)
7. Haarslev, V., Möller, R.: Incremental query answering for implementing document retrieval services. In: Proc. of DL-2003, pp. 85–94 (2003)
8. Halaschek-Wiener, C., Hendler, J.: Toward expressive syndication on the web. In: WWW 2007. Proc. of the 16th Int. World Wide Web Conference (2007)
9. Halaschek-Wiener, C., Parsia, B., Sirin, E.: Description logic reasoning with syntactic updates. In: Proc. of ODBase2006 (2006)
10. Horrocks, I.: Implementation and optimisation techniques. In: *The Description Logic Handbook: Theory, Implementation, and Applications*, pp. 313–355. Cambridge University Press, Cambridge (2003)
11. Horrocks, I., Sattler, U.: A tableaux decision procedure for SHOIQ. In: Proc. of IJCAI 2005, pp. 448–453. Professional Book Center (2005)
12. Parsia, B., Halaschek-Wiener, C., Sirin, E.: Towards incremental reasoning through updates in OWL-DL. In: *Reasoning on the Web - Workshop at WWW 2006* (2006)
13. Parsia, B., Sirin, E., Kalyanpur, A.: Debugging OWL ontologies. In: WWW, pp. 633–640 (2005)
14. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Proc. of IJCAI, 2003, pp. 355–362. Morgan Kaufmann, San Francisco (2003)
15. Stonebraker, M.: Implementation of integrity constraints and views by query modification. In: SIGMOD 1975. Proc. of the 1975 ACM SIGMOD international conference on Management of data, San Jose, California, pp. 65–78. ACM Press, New York (1975)
16. Terry, D.B., Goldberg, D., Nichols, D., Oki, B.M.: Continuous queries over append-only databases. In: Proc. of the Intl. Conf. on Management of Data (1992)

# SALT: Weaving the Claim Web

Tudor Groza, Knud Möller, Siegfried Handschuh, Diana Trif, and Stefan Decker

DERI, National University of Ireland, Galway,  
IDA Business Park, Lower Dangan, Galway, Ireland  
{tudor.groza, knud.moeller, siegfried.handschuh, diana.trif,  
stefan.decker}@deri.org  
<http://www.deri.ie/>

**Abstract.** In this paper we present a solution for “weaving the claim web”, i.e. the creation of knowledge networks via so-called claims stated in scientific publications created with the SALT (Semantically Annotated L<sup>A</sup>T<sub>E</sub>X) framework. To attain this objective, we provide support for claim identification, evolved the appropriate ontologies and defined a claim citation and reference mechanism. We also describe a prototypical claim search engine, which allows to reference to existing claims and hence, weave the web. Finally, we performed a small-scale evaluation of the authoring framework with a quite promising outcome.

## 1 Introduction

Semantic metadata constitutes the foundation for the Semantic Web and Semantic Desktop. One way to create semantic metadata is by authoring and annotating semantic documents on the desktop. In a previous paper [1] we have described the first version of SALT (Semantically Annotated L<sup>A</sup>T<sub>E</sub>X), an authoring framework for creating semantic documents for scientific publication. In our approach we used the PDF file format as container, thus being able to store both content and metadata in a single file. One of our main goals was to support the creation of knowledge networks via claims stated in scientific publications by leveraging the document semantics. Each of the claims is addressable by a unique identifier, and a set of such structured documents can be integrated to form a network of interlinked resources. In this way a semantic web of scientific literature is growing, document by document. Metaphorically speaking, the information in the documents is liberated from their surrounding document and can be integrated and re-used in new and unexpected ways.

We define claims as the most important statements within a publications i.e. the statements that characterize best the publication’s content. We believe that by reading only the claims, a reader will get a clear view of the message of the respective paper.

An example of a claim is: *[The document structure in general influences the reader’s perception on the document]*. The claim here is not a particular occurrence of text or a particular utterance, but rather the abstract idea of a statement as such. That said, a claim can have a number of textual representations, e.g. in

different scientific publications: (i) *The structure of a document has an important influence on its perception*, or (ii) *The document structure influences the reader's perception on the document*, or (iii) *The way people understand and perceive a text is often influenced by the way it is structured*.

In this paper, we propose a solution for representing and identifying the claims in a publication, and through this, we introduce a novel way for authoring and referencing. While until now everything was done at a macro-level by citing a full paper, our solution will give authors the possibility of working at a finer-grained level, by being able to cite claims within a publication. To achieve this goal, a mechanism for global and local claim identification is needed. Local identification refers the claim in the scope of the document: given the PDF file of a publications, how can one identify and reference a claim and point to its representation in that file. Global identification means identifying a claim in the entire pool of scientific publications, without referring to a particular representation.

We present our extension of SALT, which adds capabilities for specifying, referencing and searching of claims. SALT allows the author of a scientific publication to enrich the document with formal descriptions of claims, supports and rhetorical relation as part of their writing process. In the previous version of SALT, the identifiers of those elements were defined manually by the author. This approach had the following limitations: (i) the identifiers had have a local scope, (ii) the way to identify and reference a particular representation of a claim was lacking and (iii) there was no way to search for claims globally. These limitations meant that identical claims in different papers could not be linked.

SALT is based on a set of three ontologies: (i) the Document Ontology, (ii) the Annotation Ontology, and (iii) the Rhetorical Ontology. To address the limitations given above and allow the modelling of multiple claim representations, we mainly extend the Rhetorical Ontology. We also implement a means to identify a representation inside a PDF document (as a pointer). In addition we introduce a Bib<sub>T</sub>EX-like claim bibliography format and a series of special L<sup>A</sup>T<sub>E</sub>X commands for claim reference.

In Sect. 2 we detail the motivation of our work. Then we describe the way in which we model identification in the new version of SALT (Sect. 3). In Sect. 4 we present a prototypical implementation for a claim search engine. The evaluation of the SALT framework is described in Sect. 5. Before concluding, we give an overview of the related work 6.

## 2 Motivation

Claim finding in scientific publications and a means to create claim identifiers can be modelled at different levels. We discuss this under the following aspects: (i) general publication management, and (ii) claim identifier management. In both cases we make the assumption that the publications in question are annotated using the SALT framework.

In the last decade the number of conferences, workshops and implicitly publications, increased substantially. For example, in the bio-medical domain, the

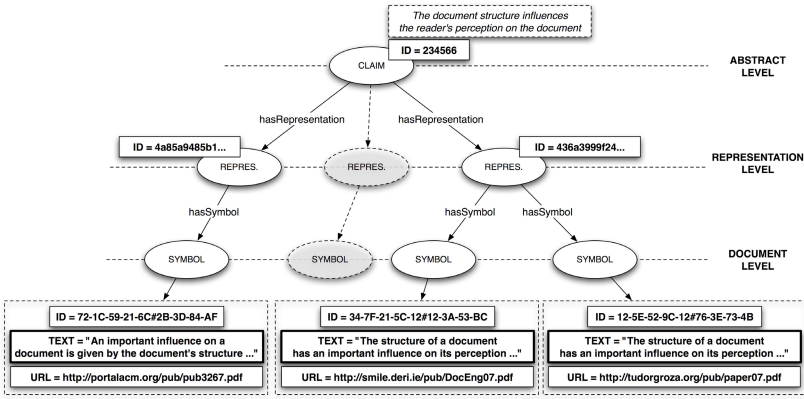


Fig. 1. Example of instantiation of the claim identification tree

number of publications in the MEDLINE 2004 database is currently growing by about 500.000 publications a year [2]. As a direct consequence, it has become very hard to find all relevant publications for a certain topic or stating a particular claim. The problem is also aggravated by the lack of uniformity in publishing papers or proceedings online. Publications are scattered on various Web sites, or, in the best case, on an online proceedings site. For the entirety of the publication space, there is no standard format for referring to publications.

Our model of claims and representations lends itself well to be mapped to the Semiotic Triangle [3], which illustrates the relations between (i) *objects* in the real world, (ii) *concepts* — abstract representations of objects in people’s minds — and (iii) *symbols*, which are expressions of concepts (in the domain of language this is an utterance or a piece of written text). There is no direct relation between objects and symbols, but instead only an indirect one via the concept. In our model, *objects* correspond to *claims*, *concepts* to *representations* of a claim in a publication as such (e.g. “Joe’s paper at ISWC2007”) and *symbols* are the actual strings of text in a particular, physical PDF file. There are three layers that have to be taken into account regarding the identification of a claim: (i) the *abstract layer*, which represents the abstraction (the *object*) referred by the claim, (ii) the *representation layer*, which denotes the collection of forms that the claim can take. Each representation is uniquely identified and referenced by the source claim. (iii) The *document layer* represents the physical form of a representation (the *symbol*). There is a 1:n relation between the symbol and the representation. The symbol has a unique identifier based on the document where it occurs.

In Fig. 1 we show an example of instantiation of the claim identification tree. The claim is assigned a random identifier when it is (ID=234566). The symbol’s identifier represents a pointer into the actual document, where the textual



representation resides (here: ID=12-5E-52-9C-12#76-3E-73-4B). For the representation in the text, we use the SHA1<sup>1</sup> sum of the symbol (for the symbol shown above, ID=436a3999f24...). The reason behind this approach, especially for the lower part of the tree is the following: the symbol’s identifier will always be generated in the scope of the host document, independently of the actual text chunk representing the claim, and thus is modified each time the document is modified. The representation’s ID is constant if the text chunk is the same. Thus, if the document is replaced but the text is unchanged, only the symbol needs to be updated.

By providing this layered approach to identification, the author can not only to reference the claim as such, but also a particular representation. As described in the following section, the identification at document level can be performed automatically.

### 3 Knowledge Identification in SALT

Previously, SALT had support for claim identification, but did not take into account the issue of multiple representations. Also, from the document engineering point of view, SALT was lacking a proper solution for identifying a particular text chunk in the resulting semantic PDF document. In the following, we describe the improvements that we brought both to the semantic and syntactic layer of SALT, in order to introduce easy and useful support for claim referencing.

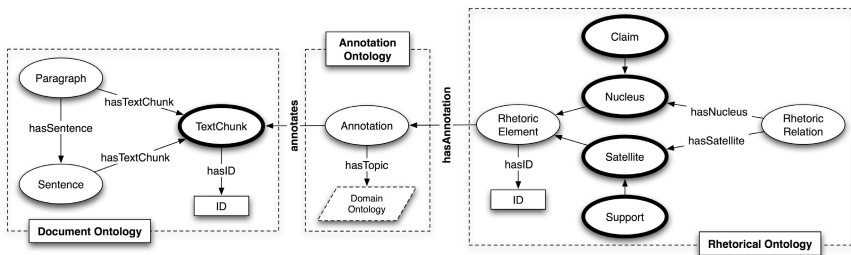


Fig. 2. Improved SALT semantic layer to support better identification

#### 3.1 Improved Identification Support

In our previous work, each rhetorical element had an identifier, introduced manually by the author, during the writing process. At the document level, the elements were represented by a sentence which was entirely duplicated in the ontology. The relation between the rhetorical element and its textual representation was realized by an annotation which also offered support for denoting the actual part of the sentence representing rhetorical element.

<sup>1</sup> RFC3174: <http://www.faqs.org/rfcs/rfc3174.html>



With respect to claim identification, the modifications we introduced affect the Rhetorical and Document ontologies<sup>2</sup>, as sketched in Fig. 2 (the newly introduced concepts are depicted with a thicker line). Conceptually, a rhetorical element still represents an annotation of a text chunk, but the way in which we model this relation has changed:

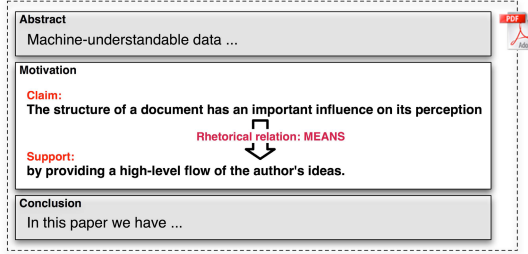


Fig. 3. Example of rhetorical elements in a scientific document

**Rhetorical Ontology** — Previously, a *Rhetorical Element* was represented by a *Claim*, or by an *Explanation* and had its own identifier. The ontology contained concepts for defining rhetorical relations between claims and explanations in the publication. It also contained the support for defining rhetorical blocks, such as *Abstract*, *Motivation* and *Conclusion*, and link the rhetorical elements and relations to them. We extended the ontology by adding two more concepts (*Nucleus* and *Satellite*). The *Nucleus* represents a generalization of a *Claim*, whereas a *Satellite* is the generalization of a *Support* (the renamed *Explanation*). As an example, given the *Claim* specified in Sect. 1 [The document structure ...], a possible *Support* is: ... by providing a high-level flow of the author's ideas. The introduction of these concepts allows us to model the publication at a more fine-grained level: text chunks can now be annotated as nuclei or satellites, and the most important nuclei can be marked as being claims. This gives us the opportunity to build complete rhetorical structure trees for the annotated publication, and emphasize the place of the claims in these trees, in parallel with the existing support for rhetorical claim trees. An example for the use of some of the elements of this ontology is shown in Fig. 3.

**Document Ontology** — The main improvements for identification, were introduced in this ontology. Where previously the *Sentence* was the finest annotation granularity, we now introduced the *TextChunk* concept. This can be instantiated as part of a *Sentence* or of a *Paragraph*, and the major change is that it will not contain the annotated text, but a pointer in the document for it. This pointer will serve as a local identifier at the document level, as described in Sect. 2.

<sup>2</sup> We also introduced a means to connect annotations to external domain ontologies, but this has no implications on the handling of claims.

### 3.2 Identification Via Document Engineering

One of the major issues that we encountered in the discussion section in [1] was a proper mechanism for identifying a chunk of text inside a PDF document. By addressing this problem, we are now able to determine the exact position of the piece of text inside a PDF document and use this pointer as a local document identifier. In the following we describe the reasons why this represents a real issue.

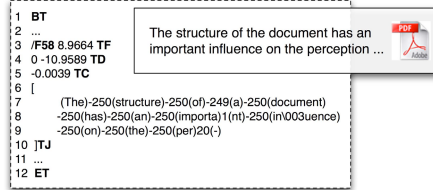


Fig. 4. Example of text encoded in PDF format

The PDF file format [4] allows representation of documents independently of the software, hardware or operating system they are created on, and of the device (display, printer) used for output. A PDF document consists of a collection of objects that describe the appearance of one or more pages, and can be accompanied by interactive elements and high-level application data. A PDF file contains the objects that make up the document and associated structural information, in the form of a self-contained sequence of bytes. Objects are incrementally added to the end of the file as the document is edited. The document pages can contain any combination of text, graphics and images grouped into articles. The appearance of a page is described by a content stream (as shown for example in Fig. 4), which contains the sequence of graphics objects that are to appear on the page. The layout and formatting is fully specified in the content stream, and it is common for the stream to be encrypted, either for security or for space-saving purposes.

Fig. 4B. shows the encoding of a part of a page in a PDF document. The entire page is encoded between the BT and ET tags [3], and every line is introduced by a TJ operator and placed between squared brackets. Before the actual content, some other parameters can be set, like the font (in this case F58 or alignment and spacing corrections, by applying the TD operator (which moves the text a little bit down in this case — line 4) or the TC operator (which specifies the spacing between words with respect to the standard setting — line 5). The real text to be analyzed is separated depending on the relative space between the individual glyphs. If the space remains the same, a sequence of glyphs is represented within the same text block between parentheses. If the distance between the glyphs is not standard, it is specified by a value written between glyphs, as shown in lines 7–9.

<sup>3</sup> The PDF tags are conforming with the PDF 1.6 Reference document.

As mentioned in Sect. 2 the symbol's identifier is represented by a pointer in the PDF document. The identifier presented in the example has two parts, delimited by a hash (#). Based on the internal organization of the PDF document, the two parts represent the beginning and the end of the text, as follows: pagecode — article in page — stream inside article — position in the stream. The first part has attached at the end the length of the text.

### 3.3 Claim Citation and Reference

In the following we use the example introduced in Sect. 2 to describe the improvements that we brought to SALT at the syntactical level, to support the referencing and storing of claims.

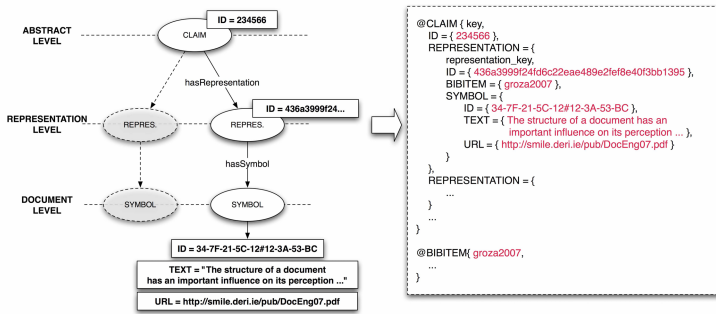


Fig. 5. Example of transformation of a claim identification tree into a CBIB item

In our vision of a Web of documents, an author will want to refer to and cite claims, not whole publications. Once the author finds the claim to be cited or referred, they need support for compiling a list of claim references and syntactical support for referencing them. For the first issue, we propose a solution similar to the current approach for storing bibliography items, i.e. defining *claim bibliography items* and storing them in format similar to the Bib<sub>TEX</sub>files. Each claim bibliography item (CBIB item) can have a key assigned for easier referencing. An example of such a claim bib item in our proposed format and based on the example introduced earlier can be seen in Fig. 5. Claim items and bib items can be mixed in the same file, as shown on the figure.

To support referencing of claims in a  $\text{\LaTeX}$  document, we introduced the `\claimref` command, which functions exactly like the usual `\cite` command, only that it refers to claim items, not bib items. To provide a maximum of usability, the author can use the command not only to cite a claim, but also to cite a particular representation of the claim. This is the reason why each of the representations has its own key, as shown in Fig. 5.

## 4 Prototypical Implementation of a Claim Search Engine

In the previous section, we described how an author can create and reference claims, based on the SALT syntactical support and the claim bibliography format. We imagine that, when an author writes a piece of text that represents a claim, they will use a claim lookup service to find out if this claim already exists. Based on the result of the lookup, they will either create a new claim or reference an existing one. Similarly, when they want to reference a particular claim representation, e.g. made by another author, they will need a means to search for and find it.

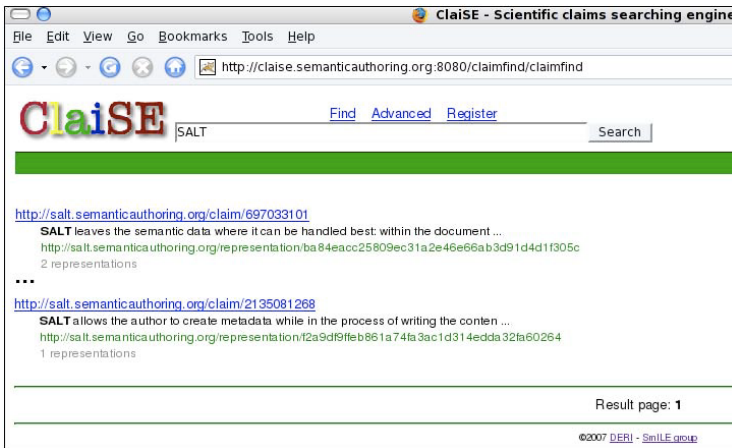


Fig. 6. Example of the claim search engine's results page

To make this possible, we designed and implemented a claim registration and search engine (Fig. 6 depicts the results page in the case of searching for the word *SALT*<sup>4</sup>). The engine fulfills a series of requirements, which we considered important and extracted from relevant literature in the domain:

**Distribution, persistence and un-ambiguity of identifiers.** In order to ensure persistence and uniqueness for identifiers, someone must take the responsibility of managing them [5]. Our system takes the responsibility of managing identifiers for claims, representations and symbols. It provides a unique identifier for each claim and takes care of creating the appropriate relations between the claim and the representations specified by authors.

**Retrievability.** The result of dereferencing a URI. [6,7,8] In our case, if the URI represents a claim, the user receives information about the representations,

<sup>4</sup> The prototype can be found at

<http://claise.semanticauthoring.org:8080/claimfind>

whereas if the URI denotes a representation, the user received information about the symbol and the claim.

**Clear determination.** [9] The identification schema, as depicted in Fig. 1 shows that our solution determines the concept that the URI is identifying and its place in the hierarchy.

**URI ownership.** [10] The user creating the URI is its owner.

The architecture of our system is strictly split into two main functionalities, which we detail as follows:

**Registration** represents the process through which the system acquires new knowledge by extracting metadata from publications. We provide two options of registering claims: (i) single publication, by providing the URL of the publication, and (ii) online proceedings, by providing the URL which contains all the publications accepted at that event. In other words, claims are registered by registering the publications that contain them. Due to the fact that the metadata is generated during the authoring process, the registration involves only the introduction of URL pointing to publications, whereas the metadata extraction and claim processing are performed automatically by the system. In the case of the online proceedings, the system extracts the metadata by iterating over all publications.

**Searching** allows a user to retrieve the URI and representation information of a claim, given a textual query. Alternatively, a textual representation can be retrieved by providing the URI. The real challenge will represent the design and implementation of a component to deal with word sense disambiguation and other related NLP techniques in order to return the closest match for a textual representation of a claim. The current prototype offers limited keyword based search.

## 5 Evaluation

We performed a small-scale evaluation of the SALT framework with a group of eight researchers, who all had knowledge of Semantic Web and Semantic Annotations technologies. Each participant annotated one of their own papers and was given a set of explanations of the model and instructions on how to perform the annotations beforehand. The evaluation consisted of seven annotations tasks as follows: (i) *Rhetorical block annotation*, (ii) *Claims markup*, (iii) *Support markup*, (iv) *Annotation of rhetorical relations between claims and supports*, (v) *Nuclei and satellites markup*, (vi) *Annotation of rhetorical relations between nuclei and satellites*, and (vii) *Annotation of domain knowledge*.

With respect to these seven annotation tasks, we asked the participants about (i) their complexity, (ii) the plausibility and soundness of the underlying model and (iii) their perceived benefit. The results for each of those areas is shown in Figs. 7 and 8 A and B. The questionnaire and results can be found online [5]. In

<sup>5</sup> <http://salt.semanticauthoring.org/evaluation/2007/05>

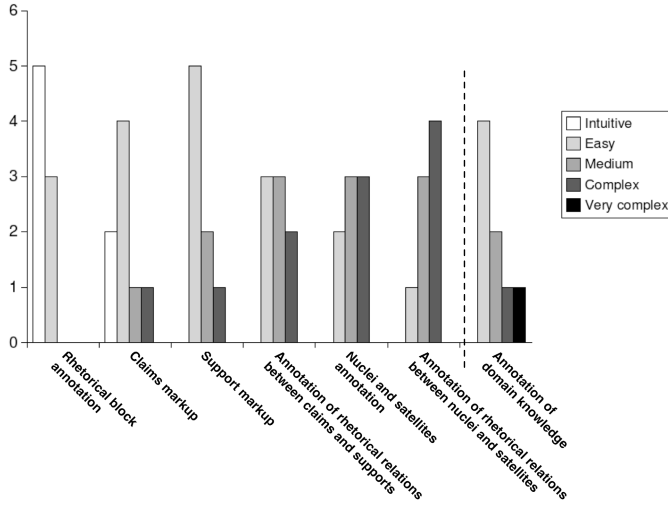


Fig. 7. Annotation tasks evaluation

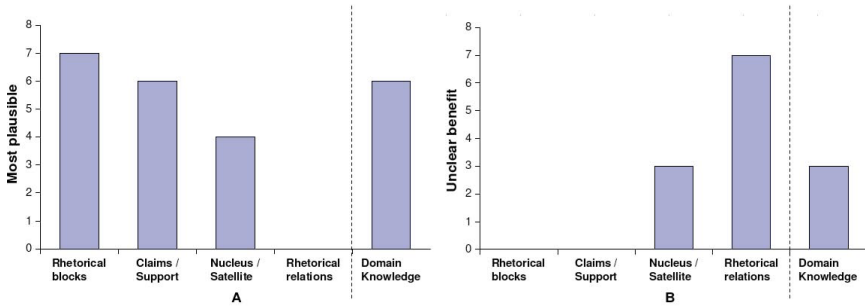


Fig. 8. A. Soundness evaluation; B. Benefit uncertainty evaluation

the following, we will elaborate our findings and discuss what can be deduced from them.

### 5.1 Findings

The target of the questions in the evaluation was to find out the researchers’ impression on how intuitive the framework is, whether it is well adapted to scientific writing or if it introduces a significant overhead in the writing process, and how plausible the overall model is perceived. In general, we observed that *with increasing complexity of the annotation task at hand, the participants’ overall impression decreased*. The annotation of rhetorical blocks, which is also the simplest of the tasks, was deemed most intuitive, while the annotation of rhetorical relations at the other end of the scale was deemed not intuitive, but instead

rather complex by most participants (see Fig. 7). Similarly, Fig. 8 A shows that the plausibility of terminology and soundness of the model were considered high by 7 out of 8 participants, whereas fewer and fewer participants said this about the more complex annotation tasks (6 found annotation of claims and support plausible, 4 found nucleus/satellite annotation plausible, while no one found the handling of rhetorical relations plausible). Figure 8 B supports the same tendency from yet another angle: none of the participants found the benefit of simple annotations (rhetorical blocks and claim/support) unclear, while 3 were unsure about the benefit of nucleus/satellite annotation and 7 about rhetorical relations.

This general tendency was expected — it seems intuitive that simpler annotations are easier to grasp, and as a result their usefulness is much more apparent. As a consequence, we see the development of tools which aid the user in the authoring process as crucial.

The annotation of domain knowledge within SALT, which is not on the scale of complexity of the other annotations, but rather a separate entity, was well received by most participants (4 found it easy, 2 said the difficulty was medium, and only 2 people found it complex or very complex to handle). Similarly, most people (6) found the supporting infrastructure plausible, and only 3 people had difficulty in seeing a benefit in the annotation of domain knowledge.

Apart from the findings which were directly taken from the questionnaire, we also observed a number of other things during the evaluation: most of the participants needed to revisit the documentation numerous times during the annotation process, and some had in fact admitted that they had not read the documentation at all before the evaluation. These observations fit very well with the general tendency we noted: users, even if they come from a technical background like the ones in the evaluation, tend to be lazy. Good documentation is useful, but we cannot make the assumption that it will actually be read thoroughly. Instead, a platform such as SALT should hide most of its complexity and guide the user with an interface that can be understood intuitively.

## 6 Related Work

The ontologies which are part of our framework have their roots in the Rhetorical Structure of Text (RST) Theory [11]. The paper provides the underlying semantics of the concepts modelled by the theory together with their definitions. A second publication by the same authors [12] provides a deep analysis of the application domains in which RST was used until a certain point in time. It is interesting to observe that the mentioned range of domains varies from computational linguistics, cross-linguistic studies and dialogue to multimedia presentations. There were several attempts to ontologize RST, and one of the first ones, including also some implications based on temporal relations is presented in [13].

Most of the similar ontologies present in the domain, capture only a part of the RST foundation, or use parallel theories as input. One of the closest to our research is presented in [14]. The authors describe the framework for sensemaking

tools in the context of the Scholarly Ontologies Project. Their starting point is represented by the requirements for a discourse ontology, which has its roots in the CCR (Cognitive Coherence Relations) Theory and models the rhetorical links in terms of similarity, causality and challenges. A similar approach is presented by Tempich et. al in [15]. The *DILIGENT Argumentation Ontology* was designed in line with the terminology proposed by the IBIS methodology [16] and captures the argumentative support for building discussions in DILIGENT processes. In DILIGENT, the argumentative support is equivalent to one of the three parts of our Rhetorical Ontology and therefore is less expressive.

None of the above mentioned approaches combines the logical and rhetorical structure of the text, in the way SALT does. This direction was mostly pursued in the NLG (Natural Language Generation) field. Here, however, the analysis performed is strictly based on NLP (Natural Language Processing) techniques. For example, Bouayad-Agha et. al analyze in [17] if the logical structure of the text can be incompatible with RST. The same authors later provide in [18] a deep analysis on how can the document structure help in the language generation process. Other approaches following the same direction but more focused on a particular domain are the ones presented in [19] and [20].

In terms of applications, we found the Compendium methodology<sup>6</sup> and the set of sensemaking tools described by Uren et. al [21] as the most interesting one in terms of similarity with our research. Their set contains tools for creating, visualizing and searching claims in scholarly documents (represented as HTML files) using a central knowledge server. Although very close to our methodology, the focus is different. We focus on the publication as an entity and on the methods for referencing claims directly, by using the metadata embedded in the publication. In regards to the claim searching engine, we used the centralized approach only for storing the minimum information about the claims and thus creating a distributed knowledge network rather than a central pool of publications.

Another interesting approach is the one of Peter et al. [22]. Their goal is to extract semantics from a  $\text{\LaTeX}$  document based on the references and index present in the document (for example *see* and *see also* references). We have a similar approach when it comes to extracting the structural information, but our focus is more oriented towards the rhetorical structure of the text and the semantic links between claims placed in different documents.

## 7 Conclusion and Future Steps

In this paper we described a solution for “weaving the claim web”, i.e. the creation of knowledge networks via the claims stated in scientific publications created with the SALT (Semantically Annotated  $\text{\LaTeX}$ ) semantic authoring framework. Like resources on the Semantic Web, each of the claims is addressable by a unique identifier.

We provide support for claim identification, evolved the appropriate ontologies and defined a claim citation and reference mechanism. We did not only describe

<sup>6</sup> <http://www.compendiuminstitute.org/>



how an author can create and reference claims, but we also implemented a prototypical claim search engine, which allows to find references to existing claims and hence, weave the web. Finally we performed a small-scale evaluation of the authoring framework with a quite promising outcome. Since the authoring process is highly incremental, we got the expected result that simpler annotations are easier to grasp, and as a result their usefulness is much more apparent. As a consequence, we see the development of tools which aid the user in the authoring process as crucial. Those tools will actively support the author in marking up their document, and will visualize them at the same time. We believe e.g. that authors will feel much more comfortable with the rhetorical relations if they can *see* a graph of those relations that is built up incrementally, while they are authoring them. This will give the direct benefit of being able to see their documents from a birds-eye view. Similarly, tools and applications that make use of those complex annotations *after* the document has been published will be needed to fully convince authors of their usefulness. We envision e.g. that the claim search engine could be enhanced by functionality that allows a user to view a summaries of documents based on their nuclei.

We should also point out SALT does not confront the user with an either-or situation: it is not required to perform annotations at *all* levels. Partial annotations are not a problem, and each type of annotation will give a different benefit by itself. Rhetorical blocks and relations, as well as the marking up of nuclei and satellites will allow new views on a document and aid both readers in understanding it, as well as a authors in checking their own argumentative structure as they write. Annotating claims and their representations, as well as linking to domain ontologies, will result in the weaving of a fine-grained web of references between documents, and offer new ways of searching and finding relevant literature in an otherwise overwhelming mass of publications.

**Acknowledgements.** The work presented in this paper was supported (in part) by the European project NEPOMUK No FP6-027705 and (in part) by the Lion project supported by Science Foundation Ireland under Grant No. SFI/02/CE1/I131. The authors would like to thank Laura Dragan, Renaud Delbru and Andreas Harth for their support and involvement in the SALT implementation.

## References

1. Groza, T., Handschuh, S., Möller, K., Decker, S.: SALT - Semantically Annotated  $\LaTeX$  for Scientific Publications. In: ESWC 2007, Innsbruck, Austria (2007)
2. Mitchell, J., Aronson, A., Mork, J., Folk, L., Humphrey, S., Ward, J.: Gene indexing: Characterization and analysis of NLM's GeneRIFs. In: Proc. AMIA Symp., pp. 460–464 (2003)
3. Ogden, C.K., Richards, I.A.: The Meaning of Meaning: A Study of the Influence of Language upon Thought and of the Science of Symbolism. Magdalene College, University of Cambridge (1923)
4. Incorporated, A.S.: PDF reference - Adobe portable document format (2004), <http://partners.adobe.com/public/developer/en/pdf/PDFReference16.pdf>

5. Walsh, N.: Names and addresses (2006), <http://norman.walsh.name/2006/07/25/namesAndAddresses>
6. Parsia, B., Patel-Schneider, P.: Meaning and the semantic web. In: WWW 2006. "Identity, Reference, and the Web" Workshop, Edinburgh, Scotland (2006)
7. Clark, T., Martin, S., Liefeld, T.: Globally distributed object identification for biological knowledgebases. In: WWW 2006. "Identity, Reference, and the Web" Workshop, Edinburgh, Scotland (2006)
8. Lawrence, S., Pennock, D.M., Flake, G.W., Krovetz, R., Coetzee, F.M., Glover, E., Nielsen, F., Kruger, A., Giles, C.L.: Persistence of web references in scientific research. *Computer* 34(2), 26–31 (2001)
9. Ginsberg, A.: The big schema of things. In: WWW 2006. "Identity, Reference, and the Web" Workshop, Edinburgh, Scotland (2006)
10. Pepper, S.: Towards the semantic superhighway: A manifesto for published subjects. In: WWW 2006. "Identity, Reference, and the Web" Workshop, Edinburgh, Scotland (2006)
11. Taboada, M., Mann, W.C.: Rhetorical structure theory: looking back and moving ahead. *Discourse Studies* 8(3), 423–459 (2006)
12. Taboada, M., Mann, W.C.: Applications of rhetorical structure theory. *Discourse Studies* 8(4), 567–588 (2006)
13. Nicholas, N.: Problems in the application of rhetorical structure theory to text generation. Master's thesis, The University of Melbourne (1994)
14. Mancini, C., Shum, S.J.B.: Modelling discourse in contested domains: a semiotic and cognitive framework. *Int. J. Hum.-Comput. Stud.* 64(11), 1154–1171 (2006)
15. Tempich, C., Pinto, H.S., Sure, Y., Staab, S.: An Argumentation Ontology for Distributed, Loosely-controlled and evolInG Engineering processes of oNTologies (DILIGENT). In: Gómez-Pérez, A., Euzenat, J. (eds.) *ESWC 2005*. LNCS, vol. 3532, pp. 241–256. Springer, Heidelberg (2005)
16. Kunz, W., Rittel, H.: Issues as elements of information system. Working paper 131, Institute of Urban and Regional Development, University of California (1970)
17. Bouayad-Agha, N., Power, R., Scott, D.: Can text structure be incompatible with rhetorical structure? In: *INLG 2000*, Mitzpe Ramon, Israel, pp. 194–200 (2000)
18. Power, R., Scott, D., Bouayad-Agha, N.: Document structure. *Computational Linguistics* 29, 211–260 (2003)
19. Linden, K.V., Martin, J.H.: Expressing rhetorical relations in instructional text: A case study of the purposes relation. *Computational Linguistics* 21(1), 29–57 (1995)
20. Silva, N.D., Henderson, P.: Narrative support for technical documents: Formalising rhetorical structure theory. In: *ICEIS*, Miami, FL, USA (2005)
21. Uren, V., Shum, S.B., Li, G., Bachler, M.: Sensemaking tools for understanding research literatures: Design, implementation and user evaluation. *Int. Jnl. Human Computer Studies* 64(5), 420–445 (2006)
22. Peter, H., Sack, H., Beckstein, C.: Document indexing - providing a basis for semantic document annotation. In: *XML-Tage 2006*, Berlin (2006)

# YARS2: A Federated Repository for Querying Graph Structured Data from the Web

Andreas Harth, Jürgen Umbrich, Aidan Hogan, and Stefan Decker

National University of Ireland, Galway  
Digital Enterprise Research Institute

**Abstract.** We present the architecture of an end-to-end semantic search engine that uses a graph data model to enable interactive query answering over structured and interlinked data collected from many disparate sources on the Web. In particular, we study distributed indexing methods for graph-structured data and parallel query evaluation methods on a cluster of computers. We evaluate the system on a dataset with 430 million statements collected from the Web, and provide scale-up experiments on 7 billion synthetically generated statements.

## 1 Introduction

The technological underpinnings of the Web are constantly evolving. With markup and representation languages, we have witnessed an upgrade from HTML to XML, mainly in the blogosphere where early adopters embraced the XML-based RSS (Really Simple Syndication) format to exchange news items. Data encoded in XML is better structured than HTML due to stricter syntax requirements and the tagging of data elements as opposed to document elements. Although the XML web is smaller in size than the HTML web, specialised search engines make use of the structured document content.

Whilst XML is appropriate in data transmission scenarios where actors agree on a fixed schema prior to document exchange, ad-hoc combination of data across seemingly unrelated domains rarely happens. Collecting data from multiple XML sources requires applications to merge data. The data merge problem is addressed by RDF, whereby, ideally, identifiers in the form of URIs are agreed-upon across many sources. In this scenario, RDF data on the Web organises into a large well-linked directed labelled graph that spans a large number of data sources.

There is an abundance of data on the Web hidden in relational databases, which represents a rich source of structured information that could automatically be published to the Web. Some weblog hosting sites have already begun exporting RDF user profiles in the Friend of a Friend (FOAF) vocabulary. Community-driven projects such as Wikipedia and Science Commons, and publicly funded projects – for example, in the cultural heritage domain – plan to make large amounts of structured information available under liberal licence models.

Hence, we see the benefit of a system that allows for interactive query answering and large-scale data analysis over the aggregated Web structured-data graph.

We study such a system as part of the Semantic Web Search Engine (SWSE) project. The goal of SWSE is to provide an end-to-end entity-centric system for collecting, indexing, querying, navigating, and mining graph-structured Web data. The system will provide improved search and browsing functionality over existing web search systems; returning answers instead of links, indexing and handling entity descriptions as opposed to documents. The core of SWSE is YARS2 (Yet Another RDF Store, Version 2), a distributed system for managing large amounts of graph-structured data.

Our work unifies experience from three related communities: information retrieval, databases, and distributed systems. We see our main contribution as identifying suitable well-understood techniques from traditional computer systems research, simplifying and combining these techniques to arrive at a scalable system to manage massive amounts of graph-structured data collected from the World Wide Web.

The remainder of this paper is organised as follows:

1. We describe the architecture and modus operandi of a distributed Web search and query engine operating over graph-structured data.
2. We present a general indexing framework for RDF, instantiated by a read-optimised, compressed index structure with near-constant access times with respect to index size.
3. We investigate different data placement techniques for distributing the index structure.
4. We present methods for parallel concurrent query processing over the distributed index.
5. We provide experimental measurements of scaling up the system to billions of statements.

## 2 Motivating Example

In the following we describe a scenario which current search engines fail to address: to answer structured queries over a dataset combined from multiple Web sources. A well interlinked graph-structured dataset furthermore enables new types of mining applications to detect common patterns and correlations on Web scale.

The use-case scenario is to find mutual acquaintances between two people. More specifically, the query is as follows: *give me a list of people known to both Tim Berners-Lee and Dave Beckett*. The query can be answered using data combined from a number of different sources.

Having aggregated all data from the sources, a query engine can evaluate the query over the combined graph. For our example query, **Dan Brickley** is one resulting answer to the question of *who are mutual acquaintances of Tim and Dave?*, that can only be derived by considering data integrated from a number of sources.

From the motivating example we can derive a number of requirements:

- **Keyword searches.** The query functionality has to provide means to determine the identifier of an entity<sup>1</sup> which can be found via keyword based searches (such as `tim berners lee`).
- **Joins.** To follow relationships between entities we require the ability to perform lookups on the graph structure. We cater for large result sets for high level queries, which is in contrast to Web searches where typically only the first few results are relevant.
- **Web data.** Since we collect data from the open Web environment, we need to pre-process the data (e.g., fusing identifiers); in addition, the index structures have to be domain independent to deal with schema-less data from the Web.
- **Scale.** Anticipating the growth of data on the Web, a centralised repository aggregating available structured content has to scale competently. The system has to exhibit linear scale-up to keep up with fast growth in data volume. A distributed architecture is imperative to meet scale requirements. To allow for good price/benefit ratio, we deploy the system on commodity hardware through use of a shared-nothing paradigm.
- **Speed.** Answers to interactive queries have to be returned promptly; fast response times are a major challenge as we potentially have to carry out numerous expensive joins over data sizes that exceed the storage capacity of one machine. To achieve adequate response times over large amounts of data, the indexing has to provide constant lookup times with respect to scale.

### 3 Preliminaries

Before describing the architecture and implementation of our system, we provide definitions for concepts used throughout the paper.

**Definition 1.** (*RDF Triple, RDF Node*) Given a set of URI references  $\mathcal{R}$ , a set of blank nodes  $\mathcal{B}$ , and a set of literals  $\mathcal{L}$ , a triple  $(s, p, o) \in (\mathcal{R} \cup \mathcal{B}) \times \mathcal{R} \times (\mathcal{R} \cup \mathcal{B} \cup \mathcal{L})$  is called an *RDF triple*.

In a triple  $(s, p, o)$ ,  $s$  is called subject,  $p$  predicate or property, and  $o$  object. To be able to track the provenance of a triple in the aggregated graph, we introduce the notion of context.

**Definition 2.** (*Triple in Context*) A pair  $(t, c)$  with a triple  $t$  and  $c \in (\mathcal{R} \cup \mathcal{B})$  is called a *triple in context*  $c$ .

Please note that we refer to a triple  $((s, p, o), c)$  in context  $c$  as a quadruple or quad  $(s, p, o, c)$ . The context of a quad denotes the URL of the data-source from hence the contained triple originated.

---

<sup>1</sup> e.g., <http://www.w3.org/People/Berners-Lee/card#i>

## 4 Architecture

We present the distributed architecture of SWSE, combining techniques from databases and information retrieval systems. A system orientated approach [6] is required for graph-based data from the Web because of scale. The system architecture of a Semantic Web Search Engine requires the following components:

- **Crawler.** To harvest web-documents, we use MultiCrawler [14]: a pipelined crawling architecture which is able to syntactically transform data from a variety of sources (e.g., HTML, XML) into RDF for easy integration into a Semantic Web system.
- **Indexer.** The Indexer provides a general framework for locally creating and managing inverted keyword indices and statement indices; we see these two index types as the fundamental building blocks of a more complex RDF index. Our framework, with combinations of keyword and statement indices, can be used to implement specialised systems for indexing RDF.
- **Object Consolidator.** Within RDF, URIs are used to uniquely identify entities. However, on the web, URIs may not be provided or may conflict for the same entities. We can improve the linkage of the data graph by resolving equivalent entities. For example, we can merge equivalent entities representing a particular person through having the same values for an email property; see [17] for more details.
- **Index Manager.** The Index Manager provides network access to the local indices, offering atomic lookup functionality over the local indices. Local indices can include keyword indices on text and statement indices such as quad indices on the graph structure, and join indices on recurring combinations of data values.
- **Query Processor.** The Query Processor creates and optimises the logical plan for answering both interactive browsing and structured queries. The Query Processor then executes the plans over the network in a parallel multi-threaded fashion, accessing the interfaces provided by the local Index Managers resident on the network.
- **Ranker.** To score importance and relevance of results during interactive exploration, we use ReConRank [16]. ReConRank is a links analysis technique which is used to simultaneously derive ranks of entities and data-sources. Ranking is an important addition to search and query interfaces and is used to prioritise presentation of more pertinent results.
- **User Interface.** To provide user-friendly search, query and browsing over the data indexed, we provide a user interface which is the human access point to the Semantic Web Search Engine. Users incrementally build queries to browse the data-graph – through paths of entity relationships – and retrieve information about entities.

The focus of the paper is on describing YARS2, the indexing and query processing functionality as illustrated in Figure 1. In the remainder of the paper, we first describe the Index Manager, next discuss the Indexer and data placement strategies, and then present the Query Processor.

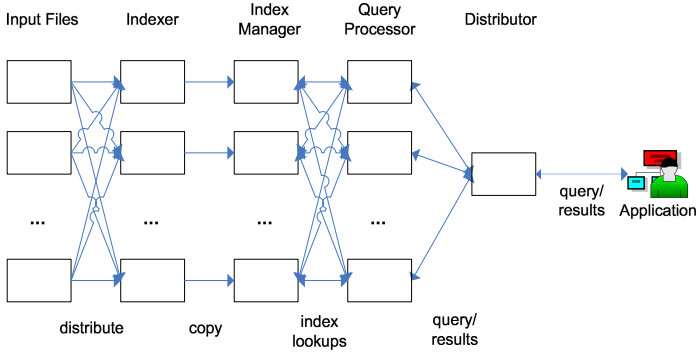


Fig. 1. Parallel index construction and query processing data flow

## 5 Anatomy of the Index Manager

We require index support to provide acceptable performance for evaluating queries. The indices include

- a keyword index to enable keyword lookups.
- quad indices to perform atomic lookup operations on the graph structure
- join indices to speed up queries containing certain combinations of values, or paths in the graph.

For the keyword index, we deploy Apache Lucene<sup>2</sup>, an inverted text index [20]. The keyword index maps terms occurring in an RDF object of a triple to the subject. We implement the quad index using a generic indexing framework using (key, value) pairs distributed over a set of machines. Similarly, join indices can be deployed using the generic indexing architecture. In the following, we illustrate the indexing framework using the quad index; join indices can be deployed analogously.

### 5.1 Complete Index on Quadruples

The atomic lookup construct posed to our index is a quadruple pattern.

**Definition 3.** (*Variable, Quadruple Pattern*) Let  $\mathcal{V}$  be the set of variables. A quadruple  $(s, p, o, c) \in (\mathcal{R} \cup \mathcal{B} \cup \mathcal{V}) \times (\mathcal{R} \cup \mathcal{V}) \times (\mathcal{R} \cup \mathcal{B} \cup \mathcal{L} \cup \mathcal{V}) \times (\mathcal{R} \cup \mathcal{B} \cup \mathcal{V})$  is called a quadruple pattern.

A naïve index structure for RDF graph data with context would require four indices: on subject, predicate, object, and context. For a single quad pattern lookup containing more than one constant, such a naïve index structure needs to execute a join over up to four indices to derive the answer. Performing joins on the quad pattern level would severely hamper performance.

<sup>2</sup> <http://lucene.apache.org/java/docs/fileformats.html>

Instead, we implement a complete index on quads [13] which allows for direct lookups on multiple dimensions without requiring joins. If we abstract each of the four elements of a quad pattern as being either a variable  $\mathcal{V}$  or a constant  $\mathcal{C} = \mathcal{RUBUL}$ , we can determine that there are  $2^4 = 16$  different quad lookup patterns for quadruples. Naïvely, we can state that 16 complete quad indices are required to service all possible quad patterns; however, assuming that prefix lookups are supported by the index, all 16 patterns can be covered by six alternately ordered indices. Prefix lookups allow the execution of a lookup with a partial key; in our case an incomplete quad.

We continue by examining three candidate data structures for providing complete coverage of the quad patterns. In examining possible implementations, we must also take into account the unique data distribution inherent in RDF. The most noteworthy example of skewed distribution of RDF data elements is that of `rdf:type` predicate; almost all entities described in RDF are typed. Also, specific schema properties can appear regularly in the data. Without special consideration for such data skew, performance of the index would be impacted.

## 5.2 Index Structure Candidates

For implementing a complete index on quadruples, we consider three index structures: B-tree, hash table, and sparse index [11].

- A **B-tree** index structure provides prefix lookups which would allow us to implement a complete index on quads with only six indices as justified in Section 5.1, one index can cover multiple access patterns. However, assuming a relatively large number of entries ( $10^6 - 10^9$ ), the logarithmic search complexity requires prohibitively many disk I/O operations (20 - 30) given that we are limited as to the portion of the B-tree we can fit into main memory.
- **Hash-tables** enable search operations in constant time; however, a hash-table implementation does not allow for prefix lookups. A complete index on quads implemented using hash tables would thus require maintaining all 16 indices. The distribution of RDF data elements is inherently skewed; elements such as `rdf:type` would result in over-sized hash buckets. If the hash value of a key collides with such an oversized bucket, a linear scan over all entries in the hash bucket is prohibitively expensive.
- A third alternative, and the one we implement, is that of a **sparse index**, which is an in-memory data structure that refers to an on-disk sorted and blocked data file. The sparse index holds the first entry of each block of the data file with a pointer to the on-disk location of the respective block. To perform a lookup, we perform binary search on the sparse index in memory to determine the position of the block in the data file where the entry is located, if present. With the sparse index structure, we are guaranteed to use a minimum number of on-disk block accesses, and thus achieve constant lookup times similar to hash tables. Since the sparse index allows for prefix lookups, we can use concatenated keys for implementing the complete index structure on quads.



### 5.3 Implementing a Complete Index on Quads

The overall index we implement comprises of an inverted text index and six individual blocked and sorted data files containing quads in six different combinations. For the sparse indices over the data files, we only store the first two elements of the first quad of each block to save memory at the expense of more data transfers for lookups keys with more than two dimensions.

More generally, the sparse index represents a trade-off decision: by using a smaller block size and thus more sparse index entries, we can speed up the lookup performance. By using a larger block size and thus less sparse index entries, we can store more entries in the data file relative to main memory at the expense of performance. The performance cost of larger block sizes is attributable to the increase of disk I/O for reading the larger blocks.

To save disk space for the on-disk indices, we compress the individual blocks using Huffman coding. Depending on the data values and the sorting order of the index, we achieve a compression rate of  $\approx 90\%$ . Although compression has a marginal impact on performance, we deem that the benefits of saved disk space for large index files outweighs the slight performance dip.

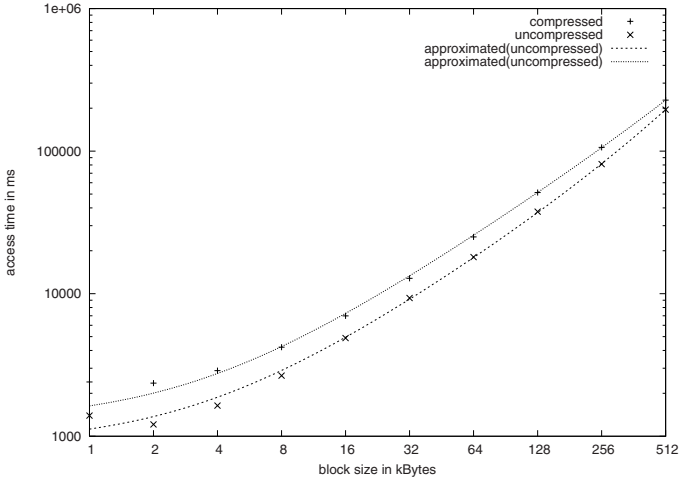
Figure 2 shows the correspondence between block size and cumulated lookup time for 100k random lookups, and also shows the impact of Huffman coding on the lookup performance; block sizes are measured pre-compression. The average lookup time for a data file with 100k entries using a 64k block size is approximately 1.1 ms for the uncompressed and 1.4 ms for the compressed data file. For 90k random lookups over a 7 GB data file with 420 million synthetically generated triples, we achieve an average seek time of 8.5 ms.

## 6 Indexer and Data Placement

The Indexer component handles the local creation of the keyword and sparse indices for the given data. For our specific complete quad index, we require building six distinctly ordered, sorted and compressed files from the raw data. The following outlines the process for local index creation orchestrated by the Indexer component:

1. Block and compress the raw data into a data file ordered in subject, predicate, object, context order (SPOC).
2. Sort the SPOC data file using a multi-way merge-sort algorithm.
3. Reorder SPOC to POCS and sort the POCS data file.
4. Complete step 3 for the other four index files.
5. Create the inverted text index from the sorted SPOC index file.

We performed an initial evaluation of the multi-way merge-sort of a file containing over 490M quads. We sorted segments of the file in memory, wrote the sorted quads to batch files, and then merge-sorted the resulting batch files. Depending on the size of the in-memory segments, the process took between 19 hours 40 minutes (80k statements in-memory) and 9 hours 26 minutes (320k statements in-memory).



**Fig. 2.** Effect of block size on lookup performance using uncompressed and compressed blocks. We performed random lookups on all keys in a file containing 100k entries with varying block sizes. Results plotted on log/log scale.

Thus far, we have covered local index management. Since our index needs to implement a distributed architecture for scalability and we require multiple machines running local Index Managers, we need to examine appropriate data placement strategies.

We consider three partitioning methods to decide which machine(s) a given quad will be indexed on:

1. random placement with flooding of queries to all machines
2. placement based on a hash function with directed lookup to machines where quads are located
3. range-based placement with directed lookups via a global data structure

We focus on the hash-based placement, which requires only a globally known hash function to decide where to locate the entry. The hash placement method can utilise established distributed hash table substrates to add replication and fail safety. For more on how to distribute triples in such a network see [8].

We avoid complex algorithms to facilitate speed optimisation. The peer to which an index *entry* (e.g. SPOC, POCS) is placed is determined by:

$$\text{peer}(\text{entry}) = h(\text{entry}[0]) \bmod m$$

where  $m$  is the number of available Index Managers.

Hashing the first element of an index entry assumes an even distribution of values for the element which is not true for predicates. The issue of load balancing based on query forwarding in hash-distributed RDF stores has been investigated in [2]. However, a simpler solution which does not require query forwarding is to resort to random distribution where necessary (for POCS), where the index is split into even sizes, and queries are flooded to all machines in parallel.

**Table 1.** Index statistics for synthetically generated dataset

Description	1 Machine	16 Machines
Number of statements	425 million	6.8 billion
Index size (complete index)	6*7 GB	672 GB
Index size (lucene)	16 GB	256 GB

To evaluate the indexing component, we created a univ(50000) dataset using the Lehigh University Benchmark [12], which we adapted to also produce variable-length text strings from an English dictionary in order to test Lucene. Table 1 summarises the indices deployed for the scale-up experiments.

## 7 Distributed Query Evaluation

We implement a general-purpose query processor operating on multiple remote Index Managers to enable evaluation of queries in SPARQL format<sup>3</sup>. In this section, we

- discuss network lookup optimisations for stream-processing large result sizes and evaluate our approach with a dataset of 7 billion statements
- devise a query processing method to perform joins over the distributed Index Managers.

### 7.1 Atomic Lookups Over the Network

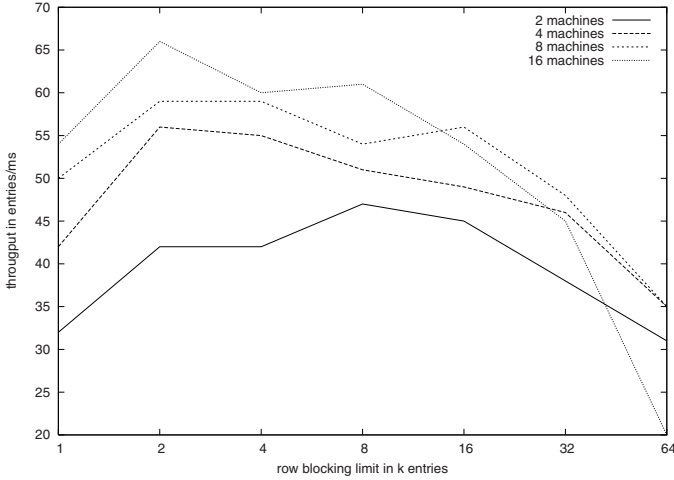
Before we can perform join processing in the Query Processor, we must implement optimised methods for handling the network traffic and memory overhead involved in sending large amounts of atomic lookup requests and receiving large amounts of response data over the network, to and from the remote Index Managers.

We implement multi-threaded requests and responses between the Query Processor and the Index Managers. For example, with our flooding distribution, each machine in the network receives and processes the lookup requests in parallel.

To be able to handle large result sets, we have to be careful not to overload main memory with intermediate results that occur during the query processing and therefore we need a streaming results model where the main memory requirements of the machines are finite since results are materialised in-memory as they are being consumed.

For a quad pattern lookup, multiple remote Index Managers are probed in parallel using multiple threads. The threaded connections to the Index Managers output results into a coordinating blocking queue with fixed capacity. The multiple threads synchronise on the queue and pause output if the queue capacity is reached.

<sup>3</sup> <http://www.w3.org/TR/rdf-sparql-query/>



**Fig. 3.** Throughput for index scan with varying row blocking sizes

Iterators that return sets instead of tuples to increase performance have been described in [18] as row blocking. We measure the impact of row blocking via an index scan query over 2, 4, 8, and 16 Index Managers. Each index manager provides access to a over 7 GB data file with 420 million synthetically generated triples, which amounts to a total capacity of roughly 7 billion statements. To be able to test keyword performance, we changed the string values in the Lehigh benchmark to include keywords randomly selected from a dictionary.

Figure 3 shows the impact of varying row blocking buffer size on the network throughput. As can be seen, throughput remains constant despite increasing the number of Index Managers servicing the index scan query. From this we can conclude that a bottleneck exists in the machine consuming results.

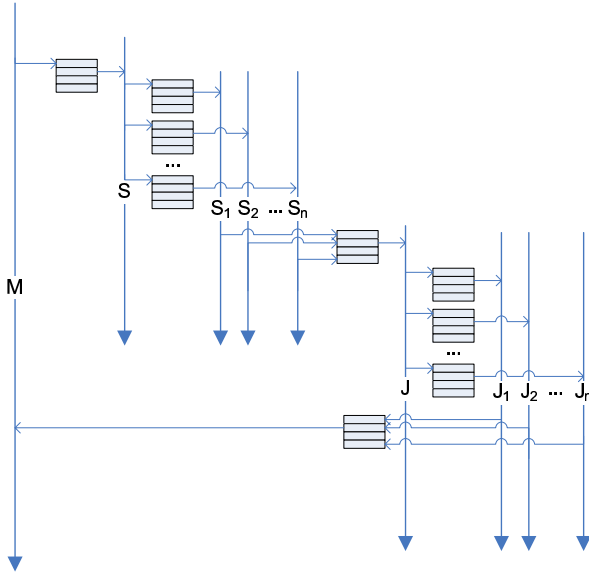
## 7.2 Join Processing

We begin our discussion of join processing by introducing the notions of variable bindings, join conditions, and join evaluation and continue by detailing our method of servicing queries which contain joins.

**Definition 4.** (*Variable bindings*) A variable binding is a function from the set of variables  $\mathcal{V}$  to the set of URI references  $\mathcal{R}$ , blank nodes  $\mathcal{B}$ , or literals  $\mathcal{L}$ .

**Definition 5.** (*Join Condition*) Given multiple quad patterns in a query, a join condition exists between two quad patterns  $\mathcal{Q}_j$  and  $\mathcal{Q}_k$  iff there exists one variable  $v \in \mathcal{V}, v \in \mathcal{Q}_j, v \in \mathcal{Q}_k$ . Joins are commutative. Variable  $v$  is termed the join variable.

In our query processing system, a query may consist either of one quad pattern (an atomic lookup) or may consist of multiple quad patterns where each pattern satisfies the join condition with at least one other pattern.



**Fig. 4.** Concurrent query execution with threads for exchanging intermediate results

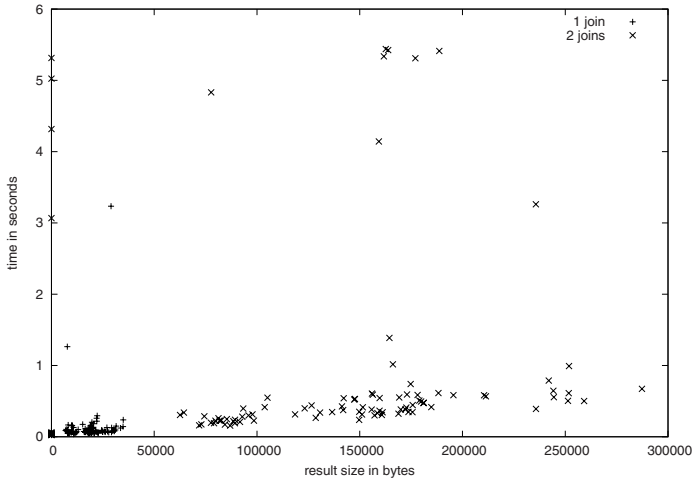
For joins we use a method called index nested loops join [11]. Multiple join operations can run concurrently in individual threads, with queues as coordination data structures for data exchange between the operators. Figure 4 illustrates the parallel execution of joins across remote Index Managers coordinated by the main thread  $M$ . Queues are represented as stack of boxes. Thread  $S$  represents a lookup operations of the first quad pattern in a query. The lookup is flooded to  $n$  Index Managers via threads  $S_{1\dots n}$ . The alternative would be to perform a directed lookup via the hash function. Intermediate results are passed to the join thread  $J$ , which in turn floods the lookups to  $n$  Index Managers via threads  $J_{1\dots n}$ . Threads  $J_{1\dots n}$  write final join evaluations to a blocking queue, which is accessed by the main thread  $M$ .

A necessary optimisation for joins requires that we carefully select which quad pattern will be serviced first to find initial valuations for the join variable. For join reordering, we can utilise a dynamic programming approach.

To evaluate the performance of distributed join processing, we deployed the 7 billion dataset over 16 Index Managers on 16 machines, and put the query processing component on a 17th machine. We tested 100 queries with a randomly chosen resource joined with one or two quad patterns. Figure 5 illustrates the correspondence between performance and result size.

## 8 Related Work

We employ variations of well-understood techniques from the fields of information retrieval, databases, and distributed systems. Inverted indices are discussed



**Fig. 5.** Distributed join evaluation performance depending on the result size

in Salton and McGill [20]. Our sparse index implementation for quads and supporting indices can be seen as a BTree index [3] with height 2, where the first level is entirely kept in memory. We optionally use compression, whose importance is well motivated by [23]. The idea of using multiple sorting order for keys to allow multidimensional lookups stems from [19]. Kowari [24] uses a similar complete quadruple index implemented using a hybrid of AVL trees and B-Trees. Semijoins, a method for performing joins in distributed databases has been introduced by Bernstein and Goodman [4]. Selinger et al. [21] introduced dynamic programming as a method for deriving query plans.

The WebBase [15] project describes in detail the architecture of a medium-sized Web repository, and various choices for implementing such a system. In contrast to documents, we deal with structured data. Swoogle [9] uses information retrieval methods to provide keyword searches over RDF data on a single machine. In contrast, we provide structured query processing capabilities on a distributed architecture.

Sesame [7] is one of the early RDF stores operating on one machine. Cai and Frank [8] propose a method to distributed RDF storage on a distributed hash table substrate. Stuckenschmidt et al. [22] investigate theoretically the use of global indices for distributed query processing for RDF. A treatment of RDF from a graph database perspective can be found in [1]. We have made a step towards unifying query processing with Web search; adding reasoning functionality to the mix [10] is the next step.

## 9 Conclusion

We have presented the architecture of a federated graph-structured data repository for use in a Semantic Web search engine, described various implementation

alternatives, and provided experimental and theoretical performance evaluation of the parallel system. To handle the complexity of a system involving a large number of machines, and to be able to optimise the performance of the individual operations, our data structures and methods have to exhibit good scale-up properties. We thus devised local data structures with constant seeks and linear throughput, optimised network data transfer, and multi-threaded query processing to achieve acceptable query performance on large data sets in a federated system.

## Acknowledgements

This work has been supported by Science Foundation Ireland under project Lion (SFI/02/CE1/I131) and by the European Commission under project TripCom (IST-4-0027324-STP).

## References

1. Angles, R., Gutiérrez, C.: Querying rdf data from a graph database perspective. In: Proceedings of the Second European Semantic Web Conference, pp. 346–360 (2005)
2. Battré, D., Heine, F., Höing, A., Kao, O.: Load-balancing in p2p based rdf stores. In: 2nd Workshop on Scalable Semantic Web Knowledge Base System (2006)
3. Bayer, R., McCreight, E.M.: Organization and maintenance of large ordered indices. *Acta Informatica* 1, 173–189 (1972)
4. Bernstein, P.A., Goodman, N.: Power of natural semijoins. *SIAM Journal on Computing* 10(4), 751–771 (1981)
5. Boncz, P.A., Zukowski, M., Nes, N.: MonetDB/X100: Hyper-Pipelining Query Execution. In: Proceedings of the Biennial Conference on Innovative Data Systems Research, pp. 225–237 (2005)
6. Brewer, E.A.: Combining Systems and Databases: A Search Engine Retrospective. In: Readings in Database Systems, 4th edn. (1998)
7. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In: Proceedings of the 2nd International Semantic Web Conference, pp. 54–68. Springer, Heidelberg (2002)
8. Cai, M., Frank, M.: Rdfpeers: a scalable distributed rdf repository based on a structured peer-to-peer network. In: Proceedings of the 13th International World Wide Web Conference, New York, NY, USA, pp. 650–657. ACM Press, New York (2004)
9. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V.C, Sachs, J.: Swoogle: A Search and Metadata Engine for the Semantic Web. In: Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management, ACM Press, New York (2004)
10. Fensel, D., van Harmelen, F.: Unifying reasoning and search to web scale. *IEEE Internet Computing* 11(2), 94–95 (2007)
11. Garcia-Molina, H., Widom, J., Ullman, J.D.: Database System Implementation. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1999)

12. Guo, Y., Pan, Z., Heflin, J.: An Evaluation of Knowledge Base Systems for Large OWL Datasets. In: Proceedings of the 3rd International Semantic Web Conference, Hiroshima, pp. 274–288. Springer, Heidelberg (2004)
13. Harth, A., Decker, S.: Optimized index structures for querying rdf from the web. In: Proceedings of the 3rd Latin American Web Congress, Buenos Aires, Argentina, pp. 71–80. IEEE Computer Society Press, Los Alamitos (2005)
14. Harth, A., Umbrich, J., Decker, S.: Multicrawler: A pipelined architecture for crawling and indexing semantic web data. In: Proceedings of the 5th International Semantic Web Conference, pp. 258–271 (2006)
15. Hirai, J., Raghavan, S., Garcia-Molina, H., Paepcke, A.: WebBase: a repository of Web pages. *Computer Networks* 33(1–6), 277–293 (2000)
16. Hogan, A., Harth, A., Decker, S.: ReConRank: A Scalable Ranking Method for Semantic Web Data with Context. In: 2nd Workshop on Scalable Semantic Web Knowledge Base Systems (2006)
17. Hogan, A., Harth, A., Decker, S.: Performing object consolidation on the semantic web data graph. In: Proceedings of 1st I3: Identity, Identifiers, Identification Workshop (2007)
18. Kossmann, D.: The state of the art in distributed query processing. *ACM Computing Surveys* 32(4), 422–469 (2000)
19. Lum, V.Y.: Multi-attribute retrieval with combined indexes. *Communications of the ACM* 13(11), 660–665 (1970)
20. Salton, G., McGill, M.: *Introduction to Modern Information Retrieval*. McGraw-Hill, New York (1984)
21. Selinger, P.G., Astrahan, M.M., Chamberlin, D.D., Lorie, R.A., Price, T.G.: Access path selection in a relational database management system. In: Proceedings of the 1979 International Conference on Management of Data, Boston, Massachusetts, pp. 23–34 (1979)
22. Stuckenschmidt, H., Vdovjak, R., Houben, G.-J., Broekstra, J.: Index Structures and Algorithms for Querying Distributed RDF Repositories. In: Proceedings of the 13th International World Wide Web Conference, pp. 631–639 (2004)
23. Witten, I.H., Moffat, A., Bell, T.C.: *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, San Francisco (1999)
24. Wood, D., Gearon, P., Adams, T.: Kowari: A platform for semantic web storage and analysis. In: XTech 2005 Conference (2005)



# Discovering Simple Mappings Between Relational Database Schemas and Ontologies

Wei Hu and Yuzhong Qu

School of Computer Science and Engineering, Southeast University,  
Nanjing 210096, P.R. China  
{whu, yzqu}@seu.edu.cn

**Abstract.** Ontologies proliferate with the growth of the Semantic Web. However, most of data on the Web are still stored in relational databases. Therefore, it is important to establish interoperability between relational databases and ontologies for creating a Web of data. An effective way to achieve interoperability is finding mappings between relational database schemas and ontologies. In this paper, we propose a new approach to discovering simple mappings between a relational database schema and an ontology. It exploits simple mappings based on virtual documents, and eliminates incorrect mappings via validating mapping consistency. Additionally, it also constructs a special type of semantic mappings, called contextual mappings, which is useful for practical applications. Experimental results demonstrate that our approach performs well on several data sets from real world domains.

## 1 Introduction

The popularity of ontologies is rapidly growing since the emergence of the Semantic Web. To date, the amount of available Web ontologies continues increasing at a phenomenal rate. For example, Swoogle [10] has collected more than 10,000 ontologies so far. However, most of the world's data today are still locked in data stores and are not published as an open Web of inter-referring resources [4]. In particular, as reported in [6], about 77.3% data on the current Web are stored in relational databases (the so-called “*Deep Web*”). Therefore, it is necessary to actualize interoperability between (Semantic) Web applications using relational databases and ontologies.

In order to achieve such interoperability, an effective way is to discover mappings between relational database schemas and ontologies. Although relational databases are based on closed-world assumption while ontologies use open-world semantics, there usually exist some approximate correspondences between them. For instance, an attribute in a relational database schema may correspond to a property in an OWL ontology. In fact, relational databases can be formalized by *First Order Logic* (FOL) [21]; while the logical foundation for OWL ontologies is *Description Logic* (DL) [2], which is a subset of FOL. Thereupon, it is feasible to construct mappings between relational database schemas and ontologies.

Discovering mappings between a relational database schema and an ontology usually employs a two-phase paradigm: (i) searching simple mappings between entities in the relational database schema and the ontology, and (ii) constructing complex compositions based on simple mappings. Finding simple mappings is an early and fundamental stage for constructing complex compositions [1]. In this paper, we focus on the first problem, i.e., discovering simple mappings between a relational database schema and an ontology.

Manually discovering such simple mappings is tedious and improbable at the Web scale. Although many (semi-)automatic approaches have been proposed to address this issue (e.g. [7][13][17]), pursuant to the results of our investigation, they have not well considered the characteristics of relational database schemas and ontologies, so the mappings they exploited are not accurate enough. Besides, most of the present approaches cannot construct semantic mappings, which are demonstrated to be useful in various practical applications (e.g. [8]).

In this paper, we propose a new approach to discovering simple mappings. It constructs virtual documents for the entities in a relational database schema as well as an ontology, so it can discover mappings from a semantic perspective, and it validates mapping consistency, so it can eliminate certain incorrect mappings. In addition, the approach constructs a special type of semantic mappings, called *contextual mappings* [5], between relations in the relational database schema and classes in the ontology. The contextual mappings can be transformed directly to view-based mappings with selection conditions, which are useful for applications from real world domains.

The remainder of this paper is organized as follows. Section 2 gives the definitions of discovering simple mappings between a relational database schema and an ontology. Section 3 sketches out our approach. Section 4 describes the approach in details. Section 5 evaluates the approach on several cases from real world domains. Section 6 discusses some related works. Finally, Section 7 concludes the paper with future work.

## 2 Problem Statement

Followed by [19], a *data model* is a collection of high-level data description constructs that hide many low-level storage details. A description of data in terms of a data model is called a *schema*.

**Definition 1.** *A relational database schema,  $\mathbb{R}$ , is a finite collection of relation schemas. A relation schema consists of the name of the relation, the names of the attributes in the relation along with their associated domains. A domain is referred to in a relation schema by the domain name and has a set of associated values. A relational database schema specifies a set of integrity constraints (ICs), which restrict the data instances that can be stored in the database.*

In our notation,  $\mathcal{R}$  denotes a relation, and  $\mathcal{A}$  denotes an attribute.  $type(\mathcal{A})$  gets the domain name of  $\mathcal{A}$ .  $rel(\mathcal{A})$  gets the relation which specifies  $\mathcal{A}$ .  $pk(\mathcal{R})$  returns the attributes appeared as the primary keys of  $\mathcal{R}$ .  $ref(\mathcal{A})$  returns the attributes



`hasAuthor` is an object property. Besides, `ConferencePaper` and `JournalPaper` are two subclasses of `Paper`. `Paper` and `hasID` are linked by a *restriction* construct. If we match the relational schema with the ontology, we possibly obtain some mappings (numbered 1–3 with dotted lines) in Fig. 1, where the first and the second ones are two mappings holding the equivalence relationships respectively, and the third is a mapping holding the subsumption relationship.

### 3 Overview of the Approach

The overview of our approach is illustrated in Fig. 2. In general, it starts with a relational schema and an ontology, and after four processing stages, it outputs a set of simple mappings.

- *Phase 1: Classifying entity types.* This phase is a preprocessing process. It heuristically classifies entities in the relational schema and the ontology into four different groups to limit the searching space of candidate mappings. Besides, this phase coordinates different characteristics between the relational schema and the ontology.
- *Phase 2: Discovering simple mappings.* This phase firstly constructs virtual documents for the entities in the relational schema and the ontology to capture their implicit semantic information. Then, it discovers simple mappings between entities by calculating the confidence measures between virtual documents via the TF/IDF model [22].
- *Phase 3: Validating mapping consistency.* This phase uses mappings between relations and classes to validate the consistency of mappings between attributes and properties. It considers the compatibility between data types of attributes and properties as well. In addition, some inference rules are also integrated in this process.

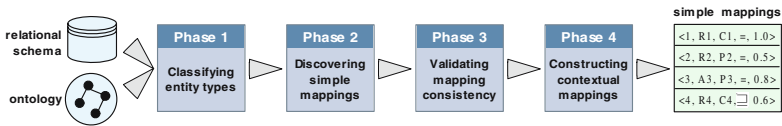


Fig. 2. Overview of the approach

- *Phase 4: Constructing contextual mappings.* This phase operates on mappings between relations and classes found in the previous phases, and supplies them with sample instances. It constructs a set of contextual mappings, which indicate the conditions how they could be transformed to view-based mappings with selection conditions.

## 4 Details of the Approach

In this section, we describe each of the four processing stages in details.

### 4.1 Classifying Entity Types

A relation in a relational schema can be classified into four disjoint types based on the properties of its primary keys: *Strong Entity Relation* ( $\mathcal{SER}$ ), *Weak Entity Relation* ( $\mathcal{WER}$ ), *Regular Relationship Relation* ( $\mathcal{RRR}$ ), and *Specific Relationship Relation* ( $\mathcal{SRR}$ ). An attribute can be distinguished into two categories by whether it is a foreign key: *Foreign Key Attribute* ( $\mathcal{FKA}$ ) and *Non Foreign Key Attribute* ( $\mathcal{NFKA}$ ). Please refer to [9] for formal definitions.

Generally, a strong (or weak) entity relation would heuristically match a class in an ontology; while a regular (or specific) relationship relation would heuristically match an object property. For example, the relation `Author` in Fig. 1 is a strong entity relation, and matches the class `Author`; while the relation `writes` is a regular relationship relation, and matches the object property `hasAuthor`.

Similarly, if an attribute is a foreign key attribute, it would match an object property; otherwise, it would match either a datatype or an object property. One exception needs to be noticed. If a relation is a regular (or specific) relationship relation, all its attributes appeared as primary keys as well as foreign keys are unnecessary to participate in the process of discovering mappings. For instance, in Fig. 1, the relation `writes` is a regular relationship relation, so the attributes `aid` and `pid` should not be considered anymore.

According to the heuristic classification, we partition entities in the relational schema and the ontology into the following four groups.

- Group 1:  $\{\{\mathcal{SER}\} \cup \{\mathcal{WER}\}\} \times \{\mathcal{C}\}$ .
- Group 2:  $\{\{\mathcal{RRR}\} \cup \{\mathcal{SRR}\}\} \times \{\mathcal{P}_O\}$ .
- Group 3:  $\{\mathcal{FKA}\} \times \{\mathcal{P}_O\}$ .
- Group 4:  $\{\mathcal{NFKA}\} \times \{\{\mathcal{P}_D\} \cup \{\mathcal{P}_O\}\}$ .

Besides, we consider some preprocessing steps to coordinate different characteristics between the relational schema and the ontology. As an example, a regular (or weak) relationship relation should be copied so that it could match two object properties holding the *inverseOf* construct. As another example, an  $n$ -arity relationship ( $n \geq 3$ ) should be reified as a group of binary relationships, because OWL ontologies can only express unary and binary relationships [2]. Please note that the heuristic rules above are not complete, but they are effective in a lot of application scenarios.

### 4.2 Discovering Simple Mappings

Inspired by [20], we present a method in this paper, which considers the structures of both the relational schema and the ontology to exploit their semantic information. The rationalities are as follows: the semantic information of a relational schema is characterized mainly by its ICs. For instance, a referential IC

involves two relations and associates a group of attributes in one relation to the keys of another relation. Likewise, an OWL ontology can be mapped to an RDF graph [18], which also indicates the semantic information in its structure.

We construct virtual documents, denoted by  $VD(\cdot)$ , for the entities in both the relational schema and the ontology to capture their structural information. A virtual document represents a collection of weighted tokens, which are derived not only from the description of the entity itself, but also from the descriptions of its neighbors. The weights of the tokens indicate their importance, and could then be viewed as a vector in the TF/IDF model [22].

The formulae in (1)–(2) build virtual documents for relations and attributes respectively. In general, for a relation, if it is a strong (or weak) entity relation, then the virtual document is from its local description; otherwise, it is a regular (or specific) relationship relation, then the virtual document is from its local description as well as the descriptions of the referenced relations. For an attribute, if it is a foreign key attribute, besides involving the description itself, we further consider the descriptions of its referenced relations; otherwise, we complement its data type into account.

$$VD(\mathcal{R}) = \begin{cases} Des(\mathcal{R}) & \mathcal{R} \in \{\mathcal{SER}\} \cup \{\mathcal{WER}\} \\ Des(\mathcal{R}) + \alpha \cdot \sum_{\substack{\mathcal{A}' \in ref(\mathcal{A}) \\ \mathcal{A} \in pk(\mathcal{R})}} Des(rel(\mathcal{A}')) & \mathcal{R} \in \{\mathcal{RRR}\} \cup \{\mathcal{SRR}\} \end{cases}, \quad (1)$$

$$VD(\mathcal{A}) = \begin{cases} Des(\mathcal{A}) + \alpha \cdot (Des(rel(\mathcal{A})) + \sum_{\mathcal{A}' \in ref(\mathcal{A})} Des(rel(\mathcal{A}'))) & \mathcal{A} \in \{\mathcal{FKA}\} \\ Des(\mathcal{A}) + \alpha \cdot Des(rel(\mathcal{A})) + \beta \cdot Des(type(\mathcal{A})) & \mathcal{A} \in \{\mathcal{NFKA}\} \end{cases}. \quad (2)$$

Analogously, the formulae in (3)–(4) construct virtual documents for classes and properties respectively. In brief, for a class, its virtual document is its local description. For a property, we consider not only its local description, but also the descriptions of its domain and range classes. Please note that if the property is a datatype property, its range is actually its data type.

$$VD(\mathcal{C}) = Des(\mathcal{C}), \quad (3)$$

$$VD(\mathcal{P}) = \begin{cases} Des(\mathcal{P}) + \alpha \cdot (\sum_{\mathcal{C} \in d(\mathcal{P})} Des(\mathcal{C}) + \sum_{\mathcal{C} \in r(\mathcal{P})} Des(\mathcal{C})) & \mathcal{P} \in \{\mathcal{PO}\} \\ Des(\mathcal{P}) + \alpha \cdot \sum_{\mathcal{C} \in d(\mathcal{P})} Des(\mathcal{C}) + \beta \cdot Des(r(\mathcal{P})) & \mathcal{P} \in \{\mathcal{PD}\} \end{cases}. \quad (4)$$

For simplicity, in this paper, we define  $Des(\cdot)$  merely returns the name of an entity as its local description.  $\alpha$  and  $\beta$  are fixed rational numbers in  $[0, 1]$ . The values should be configured with respect to practical cases. In our experience,  $\alpha$  should be a little larger than  $\beta$ .

As an example to explain the construction of virtual documents, let us see the regular relationship relation **writes** in Fig. 11. Its local description only includes “write”, and its two neighbors are **Paper** and **Author**. The virtual document of **writes** is  $VD(\text{writes}) = \{\text{“write”}, \alpha \cdot \text{“paper”}, \alpha \cdot \text{“author”}\}$ .

We discover simple mappings by calculating the confidence measures between entities. The confidence measure between any two entities is calculated by the cosine value between two vectors  $N_i$  and  $N_j$ , corresponding to two virtual documents  $VD_i$  and  $VD_j$  in the TF/IDF model:

$$\text{confidence}(VD_i, VD_j) = \text{cosine}(N_i, N_j) = \frac{\sum_{k=1}^l n_{ik}n_{jk}}{\sqrt{\sum_{k=1}^l n_{ik}^2 \sum_{k=1}^l n_{jk}^2}}, \quad (5)$$

where  $l$  is the dimension of the vector space, and  $n_{ik}$  ( $n_{jk}$ ) is the component of the vector. If the two virtual documents do not share any tokens, the confidence measure would be 0.0. If all the token scores equal completely, it would be 1.0.

### 4.3 Validating Mapping Consistency

In a relational database, relations are the unique building structures while attributes are defined by relations within their local scopes, i.e., attributes cannot stand alone without relations. In contrast, classes and properties in an OWL ontology are both first-class citizens. Nevertheless, the restriction construct in an OWL ontology provides a way of specifying local domain and range constraints on the classes [18].

In this paper, we use this kind of constraints between relations (classes) and attributes (properties) to check the consistency between mappings. We firstly assume that the mappings between relations and classes are correct, and then we utilize these mappings to validate the consistency of the candidate mappings between attributes and properties. Please note that some inference rules should be considered as well. For example, in Fig. 11 the domains of the datatype property `hasID` should include the classes `JournalPaper` and `ConferencePaper`. Besides, we also check the compatibility of data types between non foreign key attributes and datatype properties.

Considering the example shown in Fig. 11 again, we assume that the mapping between the relation `Paper` and the class `Paper` has been discovered, then we validate two candidate mappings: the mapping between the attribute `id` in `Paper` and the property `hasID`, and the mapping between the attribute `id` in `Author` and the property `hasID`. It is obvious that the latter one is inconsistent with the mapping between the relation `Paper` and the class `Paper`.

### 4.4 Constructing Contextual Mappings

Data integration is a traditional application for matching [23]. As an important infrastructure, query answering provides certain answers of queries over mappings. In [8], it has been proven that subsumption relationships are helpful for the optimization of query answering. It inspires us to construct mappings holding the subsumption relationships between a relational schema and an ontology.

A naive approach to construct such mappings holding the subsumption relationships is reusing the hierarchies of entities in a relational schema and an ontology. For the ontology, the hierarchy of entities is explicitly specified by the *subClassOf* constructs; while for the relational schema, the *Reverse Engineering* techniques (e.g. [9]) could also help us recover such hierarchy. But the naive approach would suffer from finding too many mappings holding the subsumption relationships, and most of them are not useful in practical applications.

In this paper, we focus on searching a special type of mappings holding the subsumption relationships, called *contextual mappings*. It can be directly translated to conditional mappings or view-based mappings [5]. Let us see the example illustrated in Fig. 3. In the example, when the value of the attribute `type` in the relation `Paper` equals to “1”, the relation `Paper` matches the class `JournalPaper`; and when the value equals to “2”, it matches the class `ConferencePaper`. So the contextual mappings not only hold the subsumption relationships, but also explain the conditions how they can be converted to the equivalence relationships.

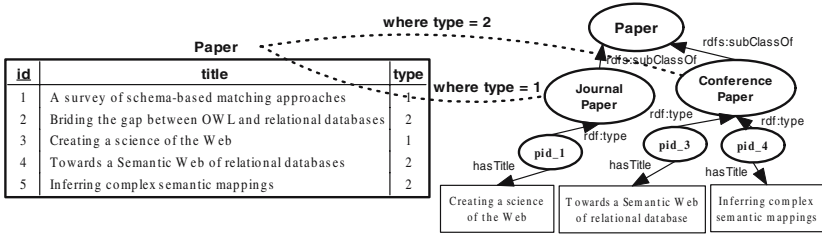


Fig. 3. The toy example with instances

Constructing contextual mappings requires two preconditions: (i) certain input mappings, and (ii) sample instances for both the relational schema and the ontology. To the first precondition, we have already found such mappings in the previous subsections. To the second one, a database is a collection of data [19], so it always contains instances; while according to the report by Swoogle [10], only 1.5% Semantic Web ontologies have few individuals. So it is possible to provide some overlapped instances for both the relational schema and the ontology.

The algorithm `ContextMatch` is shown in Table 1. The input of the algorithm is a relational schema with associated sample data, an ontology with associated sample individuals, and a set of simple mappings found previously. The goal of the algorithm is to assemble a collection of contextual mappings. To accomplish this, the algorithm considers each input mapping between a relation and a class in turn at line 1, and selects the data  $J_{\mathcal{R}}$  from the relation at line 2.

Next, in lines 3–5, the algorithm enumerates all the disjoint subclasses of the class, and for each subclass, it selects its individuals  $I_C^k$ , and finds the instances  $J_{\mathcal{R}}^k$  from  $J_{\mathcal{R}}$  which match  $I_C^k$  by `InstanceMatch`. As a result,  $J_{\mathcal{R}}$  is partitioned into some disjoint subsets corresponding to the subclasses. In our current implementation, `InstanceMatch` is developed by comparing the values of the instances through the input mappings between attributes and properties. Please note that `InstanceMatch` is the most time-consuming routine throughout the whole algorithm, and its complexity is relevant to the sizes of sample instances.

In lines 6–8, the algorithm repeatedly examines each attribute in the relation whether it is a categorical attribute or not. If so, `InformationGain` computes the information gain (IG) of the attribute on  $\{J_{\mathcal{R}}^k\}$ . The attribute  $\mathcal{A}_l$ , which has the



**Table 1.** An algorithm for constructing contextual mappings

---

**Algorithm.** ContextMatch( $\mathbb{S}, \mathbb{O}, J, I, M$ ).

---

**Input:** A relational schema  $\mathbb{S}$ , with associated sample data  $J$ ,  
an ontology  $\mathbb{O}$ , with associated sample individuals  $I$ ,  
and a set of simple mappings  $M$ .

**Output:** A set of contextual mappings  $M'$ .

1. for each  $m = \langle id, u, v, t, f \rangle$  in  $M$  s.t.  $u$  is a relation and  $v$  is a class
  2.      $J_{\mathcal{R}} := \text{InstanceSelect}(u, \mathbb{S}, J)$ ;
  3.     for each  $\mathcal{C}_k$  s.t.  $\mathcal{C}_k$  is a subclass of  $v$  and disjoint with any other  $\mathcal{C}_{k'}$
  4.          $I_{\mathcal{C}}^k := \text{InstanceSelect}(\mathcal{C}_k, \mathbb{O}, I)$ ;
  5.          $J_{\mathcal{R}}^k := \text{InstanceMatch}(J_{\mathcal{R}}, I_{\mathcal{C}}^k, M)$ ;
  6.     for each  $\mathcal{A}_i$  in  $u$  s.t.  $\mathcal{A}_i$  is a categorical attribute
  7.          $ig_i := \text{InformationGain}(\mathcal{A}_i, \{J_{\mathcal{R}}^k\})$ ;
  8.      $l := \arg \max(ig_i)$ ;
  9.     for each  $\mathcal{C}_k$  s.t.  $ig_l > \tau$
  10.          $M' := M' \cup \{\langle new\_id, u, \mathcal{C}_k, \mathcal{A}_l = \text{"xxx"}, ig_l \rangle\}$ ;
  11. return  $M'$ ;
- 

maximal IG, is chosen as the best splitting attribute. Please note that computing IG for classification (e.g. *decision tree*) has been widely studied in the fields of Machine Learning and Data Mining. In the end, in lines 9–11, if the IG of  $\mathcal{A}_l$  is larger than a given threshold  $\tau$ , then we construct a new contextual mapping, and add it to the set of contextual mappings as output.

## 5 Evaluation

We have implemented the proposed approach in Java, called MARSON (Mapping between relational schemas and ontologies). In this section, we report on some results of an experimental study. Please note that all the test cases and experimental results are available at our website<sup>1</sup>.

### 5.1 Case Study

In our evaluation, we choose the data sets used in [1], which can be downloaded from the website<sup>2</sup>. The data sets are obtained from a variety of real world domains, and the relational database schema and the ontology in each data set are developed independently. Volunteers are trained to set up reference mappings. The statistical data of the data sets are listed in Table 2.

<sup>1</sup> <http://iws.seu.edu.cn/infores/tools/falcon-ao/marson.zip>

<sup>2</sup> <http://www.cs.toronto.edu/~yuana/research/maponto/relational/testData.html>

**Table 2.** Characteristics of data sets

ID	$\mathbb{R}$	$\#\mathcal{R}$	$\#\mathcal{A}$	$\mathbb{O}$	$\#\mathcal{C}$	$\#\mathcal{P}$	Ref.
1	UTCS	8	32	Univ. CS	53	35	18
2	VLDB	9	38	Conference	18	29	27
3	DBLP	5	27	Bibliography	66	81	21
4	OBSERVER	8	115	Bibliography	66	81	72
5	Country	6	18	Factbook	43	209	22

## 5.2 Experimental Methodology

Two experiments are designed to evaluate MARSON. In the first experiment, we measure the performance of MARSON on discovering simple mappings (without contextual mappings) between a relational schema and an ontology. Four approaches are set up for comparison: (a) a simple approach, denoted by SIMPLE, which only utilizes the local descriptions of the entities (i.e.  $\alpha = 0, \beta = 0$  in (1)–(4)) for calculating the confidence measures in the TF/IDF model, and does not validate the consistency between mappings; (b) an approach, denoted by VDOC, which discovers simple mappings by constructing virtual documents, but does not check mapping consistency; (c) an approach, denoted by VALID, which only validates the consistency between mappings found by the simple approach; and (d) a simple version of an existing prototype RONTO [17]. We have implemented it based on I-SUB [24] as its elementary matcher for calculating the confidence measures between entities. Please refer to Section 6 for a detailed introduction. The parameters of VDOC and MARSON in (1)–(4) are uniformly set as follows:  $\alpha = 0.2, \beta = 0.1$ . Please note that our tests also show that MARSON is stable with slight changes on  $\alpha$  and  $\beta$ .

In the above experiment, we use the well known *F1-Measure* (a combination of *precision* and *recall*) to evaluate the performance of each approach. We have tested a variety of cutoffs or thresholds for each approach, and selected the best ones in our experiments. It seems fair to all the approaches.

In the second experiment, we evaluate the effectiveness of MARSON on constructing contextual mappings. Some real instances are collected from the Web corresponding to the relational schemas and ontologies in the first three data sets (more than 50 instances for each relation and class). We look into the contextual mappings found by our algorithm by comparing with the mappings established by experienced volunteers.

## 5.3 Discussion on Experimental Results

The results on measuring the F1-Measures of SIMPLE, VDOC, VALID and MARSON are illustrated in Fig. 4(a). It shows that either VDOC or VALID performs better than SIMPLE, and MARSON is dominant in most data sets. More specifically, VDOC improves SIMPLE in tests 1, 2, and 5, because it can discover the mappings between the entities having little commonality in their local descriptions. VALID enhances SIMPLE in almost all the data sets, since it often occurs

that the relational schema in each data set has some attributes in different relations owning the same names such as “id” or “name”. But the mappings found additionally by VDOC and VALID are not completely orthogonal, some of them are overlapped. Based on the experiment, MARSON is the best one on nearly all the data sets except for a slight lag in test 4. It demonstrates that it is feasible to integrate VDOC and VALID together and achieve a good result.

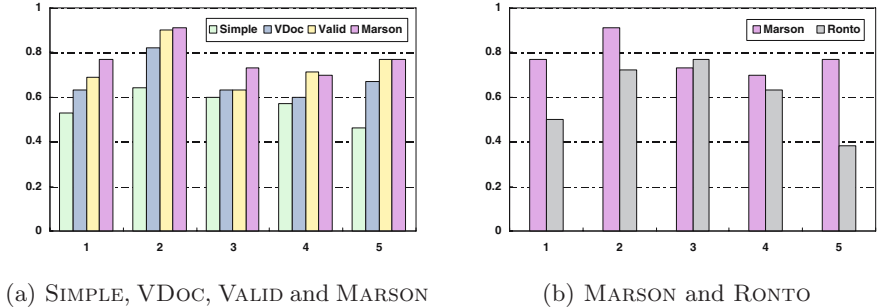


Fig. 4. Comparison of F1-Measure

The comparison results between MARSON and RONTO are shown in Fig. 4(b). It indicates that MARSON performs better than RONTO in average F1-Measure. The reason is that MARSON can find additional correct mappings by VDOC, and eliminate some inconsistent mappings by VALID.

Furthermore, it is valuable to mention that MARSON is quite efficient in the first experiment. Based on our environment (Intel Pentium IV 2.8GHz processor, 512MB memory, Windows XP Professional, and Java SE 6), it takes about 5 seconds to complete all the five tests (including the parsing time).

In the second experiment, the contextual mappings constructed by our algorithm are evaluated by experienced volunteers, and the results are exhibited in Table 3. MARSON constructs some interesting contextual mappings. For instance, in test 2, MARSON constructs a contextual mapping between the relation **Event** and the class **Conference**. It points out that when the values of the attribute **type** in **Event** equals to “Research Session” or “Industrial Session”, the subsumption relationship between **Event** and **Conference** can be converted to the equivalence relationship. In most tests, our algorithm finds all the possible contextual mappings. But in test 1, it misses the contextual mapping between the relation **academic\_staff** and the subclasses of **Faculty** (e.g. **Professor**), because without background knowledge, MARSON cannot discover the mapping between **academic\_staff** and **Faculty**.

## 6 Related Work

Discovering mappings between relational database schemas and ontologies is an interdisciplinary research in both Database and Semantic Web communities. At

**Table 3.** Evaluation of contextual mappings

ID	Found	Existing	Correct
1	2	3	2
2	1	1	1
3	1	1	1

an early stage, some works (e.g. [7]) try to implement visual toolkits in order to help users specify mappings manually. This kind of approaches may succeed in some specific scenarios, but they are impractical for the scale of the Web.

At present, many works focus on discovering mappings (semi-)automatically. For example, Dragut and Lawrence [13] transform relational schemas and ontologies into directed labeled graphs respectively, and reuse the schema matching tool COMA [11] to exploit simple mappings. Papapanagiotou et al. [17] develop a plug-in named RONTO, which introduces six different strategies to discover mappings by distinguishing the types of entities in relational schemas, and it is similar to the SIMPLE approach in this paper. However, all the approaches mentioned above disregard the structural differences in models, and do not validate the consistency between mappings.

Furthermore, to the best of our knowledge, no existing work raises the issue of constructing semantic mappings between relational schemas and ontologies. In both Database and Semantic Web communities, more and more researchers have been aware of the importance for constructing semantic mappings (e.g. [5,8]), and we believe it is also necessary to consider semantic mappings between relational schemas and ontologies. In this paper, we propose a novel algorithm to find a special type of semantic mappings, called contextual mappings, which can directly help query answering and data integration.

Besides, there exist some literatures addressing the problem from other directions. For example, Dou et al. [12] describe a general framework for integrating databases with ontologies via a first-order ontology language WEB-PDDL. Barasa et al. [3] design a language R2O for expressing complex mappings. Motik et al. [16] propose an extension of OWL with ICs that captures the intuition behind ICs in relational databases. An et al. [1] develop a prototype MAPONTO for inferring complex semantic mappings formalized by *Horn-Clauses* between relational tables and ontologies deriving from simple mappings. It is worthy of note that our approach can provide such initial mappings.

## 7 Summary and Future Work

In summary, the main contributions of this paper are listed as follows. Firstly, we have presented a new approach to discovering simple mappings between entities in a relational database schema and an ontology. It captures semantic information contained in the structures of the entities based on virtual documents, and eliminates incorrect mappings by validating mapping consistency.

Secondly, we have proposed a novel algorithm to construct contextual mappings. The algorithm reuse simple mappings and supplies additional sample instances for a relational database schemas and an ontology. Contextual mappings specify the conditions for converting to view-based mappings with selection conditions, which further help query answering and data integration.

Finally, we have experimentally evaluated our approach on several data sets from real world domains. The results demonstrate that our approach performs well as compared to some existing approaches in average F1-Measure. Besides, the results also show that the contextual mappings constructed by our approach are useful and meaningful.

In the future work, we look forward to comparing our approach with some intermediate approaches which firstly convert one data model to the other, and then reuse certain schema matching or ontology matching methods to discover simple mappings. We also hope to consider some machine learning techniques for mining some other interesting and useful semantic mappings. Finally, we would like to integrate our approach into some existing data integration tools in order to evaluate its effectiveness.

## Acknowledgements

The work is supported in part by the NSFC under Grant 60573083, and in part by the 973 Program of China under Grant 2003CB317004. We thank Dongdong Zheng and Yuanyuan Zhao for their work in the experiments. We also appreciate anonymous reviewers for their precious comments.

## References

1. An, Y., Borgida, A., Mylopoulos, J.: Inferring complex semantic mappings between relational tables and ontologies from simple correspondences. In: ODBASE 2005. Proceedings of International Conference on Ontologies, Databases and Applications of Semantics, pp. 1152–1169 (2005)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The description logic handbook: Theory, implementation and applications. Cambridge University Press, Cambridge (2003)
3. Barrasa, J., Corcho, O., Gomez-Perez, A.: R2O, an extensible and semantically based database-to-ontology mapping language. In: Bussler, C.J., Tannen, V., Fundulaki, I. (eds.) SWDB 2004. LNCS, vol. 3372, Springer, Heidelberg (2005)
4. Berners-Lee, T., Hall, W., Hendler, J., Shatbolt, N., Weitzner, D.J.: Creating a science of the Web. *Science* 313, 769–771 (2006)
5. Bohannon, P., Elnahrawy, E., Fan, W., Flaster, M.: Putting context into schema matching. In: VLDB 2006. Proceedings of the 32nd International Conference on Very Large Data Bases, pp. 307–318 (2006)
6. Chang, K.C., He, B., Li, C., Zhang, Z.: Structured databases on the Web: Observation and implications. *SIGMOD Record*. 33(3), 61–70 (2004)

7. Chen, H., Wang, Y., Wang, H., Mao, Y., Tang, J., Zhou, C., Yin, A., Wu, Z.: Towards a Semantic Web of relational databases: A practical semantic toolkit and an in-use case from traditional Chinese medicine. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 750–763. Springer, Heidelberg (2006)
8. Chen, H., Wu, Z., Wang, H., Mao, Y.: RDF/RDFS-based relational database integration. In: ICDE 2006. Proceedings of the 22nd International Conference on Data Engineering, p. 94 (2006)
9. Chiang, R.H.L., Barron, T.M., Storey, V.C.: Reverse engineering of relational databases: Extraction of an EER model from a relational database. *Data & Knowledge Engineering* 12, 107–142 (1994)
10. Ding, L., Pan, R., Finin, T., Joshi, A., Peng, Y., Kolari, P.: Finding and ranking knowledge on the Semantic Web. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 156–170. Springer, Heidelberg (2005)
11. Do, H.H., Rahm, E.: COMA – A system for flexible combination of schema matching approaches. In: Bressan, S., Chaudhri, A.B., Lee, M.L., Yu, J.X., Lacroix, Z. (eds.) CAiSE 2002 and VLDB 2002. LNCS, vol. 2590, pp. 610–621. Springer, Heidelberg (2003)
12. Dou, D., LePendu, P., Kim, S., Qi, P.: Integrating databases into the Semantic Web through an ontology-based framework. In: SWDB 2006. Proceedings of the 3rd International Workshop on Semantic Web and Databases, p. 54 (2006)
13. Dragut, E., Lawrence, R.: Composing mappings between schemas using a reference ontology. In: ODBASE 2004. Proceedings of International Conference on Ontologies, Databases and Applications of Semantics, pp. 783–800 (2004)
14. Gruber, T.R.: A translation approach to portable ontology specifications. *Knowledge Acquisition* 5(2), 199–220 (1993)
15. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: The state of the art. *The Knowledge Engineering Review* 18(1), 1–31 (2003)
16. Motik, B., Horrocks, I., Sattler, U.: Bridging the gap between OWL and relational databases. In: WWW 2007. Proceedings of the 16th International World Wide Web Conference (2007)
17. Papapanagioutou, P., Katsioui, P., Tsetsos, V., Anagnostopoulos, C., Hadjiefthymiades, S.: RONTO: Relational to ontology schema matching. *AIS SIGSEMIS BULLETIN* 3(3-4), 32–36 (2006)
18. Patel-Schneider, P.F., Hayes, P., Horrocks, I. (eds.): OWL Web ontology language semantics and abstract syntax. W3C Recommendation 10 (February 2004), <http://www.w3.org/TR/owl-semantics/>
19. Ramakrishnan, R., Gehrke, J.: Database management systems, 3rd edn. McGraw-Hill, New York (2002)
20. Qu, Y., Hu, W., Cheng, G.: Constructing virtual documents for ontology matching. In: WWW 2006. Proceedings of the 15th International World Wide Web Conference, pp. 23–31 (2006)
21. Rybiński, H.: On first-order-logic databases. *ACM Transaction on Database Systems* 12(3), 325–349 (1987)
22. Salton, G., McGill, M.H.: Introduction to modern information retrieval. McGraw-Hill, New York (1983)
23. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. *Journal on Data Semantics* IV, 146–171 (2005)
24. Stoilos, G., Stamou, G., Kollias, S.: A string metric for ontology alignment. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 623–637. Springer, Heidelberg (2005)

# Potluck: Data Mash-Up Tool for Casual Users

David F. Huynh, Robert C. Miller, and David R. Karger

MIT Computer Science and Artificial Intelligence Laboratory  
32 Vassar St., Cambridge, MA 02139, USA  
{dfhuynh, rcm, karger}@csail.mit.edu

**Abstract.** As more and more reusable structured data appears on the Web, casual users will want to take into their own hands the task of mashing up data rather than wait for mash-up sites to be built that address exactly their individually unique needs. In this paper, we present Potluck, a Web user interface that lets casual users—those without programming skills and data modeling expertise—mash up data themselves.

Potluck is novel in its use of drag and drop for merging fields, its integration and extension of the faceted browsing paradigm for focusing on subsets of data to align, and its application of simultaneous editing for cleaning up data syntactically. Potluck also lets the user construct rich visualizations of data in-place as the user aligns and cleans up the data. This iterative process of integrating the data while constructing useful visualizations is desirable when the user is unfamiliar with the data at the beginning—a common case—and wishes to get immediate value out of the data without having to spend the overhead of completely and perfectly integrating the data first.

A user study on Potluck indicated that it was usable and learnable, and elicited excitement from programmers who, even with their programming skills, previously had great difficulties performing data integration.

**Keywords:** mash up, drag and drop, faceted browsing, simultaneous editing, ontology alignment, end-user programming, semantic web, RDF.

## 1 Introduction

The construction of a Web 2.0 mash-up site is typically done by programmers. In this paper, we introduce Potluck, a tool that lets casual users—non-programmers—make mash-ups by themselves:

- Potluck allows the user to merge fields from different data sources, so that they are treated identically for sorting, filtering, and visualization. Fields are merged using simple drag and drop of field names.
- Potluck provides an efficient means for the user to clean up data syntactically, homogenize data formats, and extract fields syntactically embedded within existing fields, all through the application of simultaneous editing [9].
- Potluck supports faceted browsing [19] to let users explore and identify subsets of data of interest or subsets of data that need alignment and clean up.

We conducted a user study of Potluck and report the results here, which show that Potluck is a viable mash up solution for casual users and that it even has features desired by programmers.

In contrast, today's mash-up construction can only be done by programmers using complex technologies as it involves many technical challenges, particularly:

- scraping data from the original sites, where it may be hidden behind complex queries and web templates;
- aligning the original sites' data into a single coherent data model; and
- creating an effective visualization of the merged data.

These challenges are only worth to overcome for mash-ups that will appeal to many people, and they prevent the construction of mash-ups that are personal or narrow in appeal, serving only a few users and giving little return on investment of efforts and resources. For example, a high-school student writing a term report on the knowledge and use of mycology (mushrooms) among Polynesian tribes will be unlikely to find a mash-up site containing data on both mycology as well as Polynesians. She will also unlikely find enough resources (money and programming skills) to get such a site built quickly enough to meet her deadline, if ever built at all. The long tail of mash-up needs is thus left unanswered.

## 2 Scenario

Before describing the user interface of Potluck, we motivate it with a scenario that illustrates various idiosyncrasies of personal mash-up construction. Let us be optimistic that within a decade, the Semantic Web will be prevalent and RDF data will be everywhere. This scenario argues that even in this future world, users will *still* face problems making mash-ups between data sources.

In 2017, a historian named Henry is documenting the first cases of a rare genetic disease called GD726. These first cases occurred in the Valentine family in the 1820s. He wants to include in his final report a genealogical tree of the Valentine family, annotated with the disease's infliction, as well as a comprehensive table of the Valentines' data in an appendix.

Like most historians, Henry is not a programmer but he is experienced in collecting and managing data in his professional work. The proliferation of RDF means that Henry does not need programming skills to scrape HTML himself: all the information needed for his research has been converted into RDF by various independent organizations and individuals, both professionals and enthusiasts. Henry thinks it would be trivial to simply pool the RDF together and call it done.

Henry tracks down various birth certificate issuing offices and death certificate issuing offices where the Valentines lived for their RDF data. He notes that some offices use `dc:date` in their data to mean "birth date," some to mean "death date," and some "certificate issuing date." It would be disastrous to consider all the `dc:dates` the same even if the same predicate URI is used.

Henry also tracks down hospital records, which contain `hospital:tod` (short for "time of death"). Hence, `hospital:tod` is equivalent to some of the `dc:dates`. It would be hard to match `hospital:tod` with `dc:date` based on string analysis alone, yet match for some of the cases only.

The records all have geographical location names, but these names are not fully qualified. Those responsible for digitizing them thought that since all locations were within their country, there was no need to include the country name. As a consequence, Henry needs to append the country name to the many location names in order to map them.



People’s names are encoded in two different forms: “first-name last-name” in some data sets and “last-name, first-name” in others. Nick names are also present (e.g., “Bill” instead of “William”, and “Vicky” instead of “Victoria”).

The hospital records also pose problems. While most of their admittance dates are in ISO 8601 format, a few are of the kind “Easter Day 1824.” Such sloppiness has been observed in industrial and institutional databases, and should be expected on the Semantic Web.

Despite all these problems, there is one good thing about the data: Henry can reliably get the mother and father of each Valentine through the **gen:mother** and **gen:father** predicates, which seem to be very widely adopted. This helps Henry construct a genealogical tree visualization.

However, as males and females both have equal chance of passing on GD726, Henry wants to treat **gen:mother** and **gen:father** the same while tracing the disease through the family. Unfortunately, adding an **owl:sameAs** equivalence between those two predicates will break his genealogical tree.

While all parties involved in this scenario acted logically and responsibly, Henry still ends up with a mess of RDF. To fix up the data, Henry must be able to:

- Merge **dc:dates** into *several* groups (the birth dates and the death dates) even though they all use the same predicate URI. This requires distinguishing the fields by their origins rather than just by their URIs.
- Merge **gen:mother** and **gen:father** together in some situations while keeping them separate in other situations. This precludes the simple approach of adding **owl:sameAs** statements in the data model to implement equivalences.
- Edit the data efficiently to unify its syntax.
- Fix up the data iteratively as he learns more and more about the data.

### 3 User Interface

We now describe Potluck’s user interface, showing how it addresses the problems in the scenario above. The reader is encouraged to view a screencast to understand Potluck’s interactivity: <http://simile.mit.edu/potluck/>.

Figure 1 shows the starting screen of Potluck where the user can paste in several URLs of Exhibit-powered web pages and click Mix Data. This results in Figure 2, which lists data records from the original web pages. The records are interleaved by origins—the pages from which they have been extracted—to ensure that some records of each data set are always visible.

Fields are rendered as *field tags*: **label**, **position**, and **title**. Field tags are color-coded to indicate their origins: blue from one source and pink from another in Figure 2. Three core fields, **label**, **type**, and **origin**, are automatically assigned to all records and their tags are colored gray. Fields from different origins having the same name are considered different. For example, while **phone** means office phone, **phone** might mean secretary’s phone. Or more dangerously, **dc:date** in the scenario (in section 2) has several distinct meanings. These semantic differences, subtle or significant, might or might not be important to one particular user at one particular moment in time. Keeping the fields apart rather than

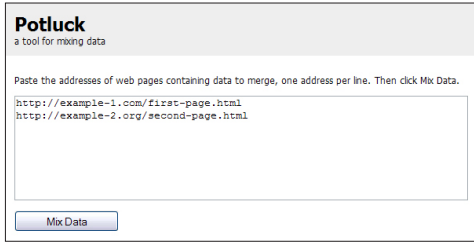


Fig. 1. The starting screen of Potluck takes URLs to Exhibit-powered web pages. Clicking Mix Data yields the mixed data in a screen like Figure 2.

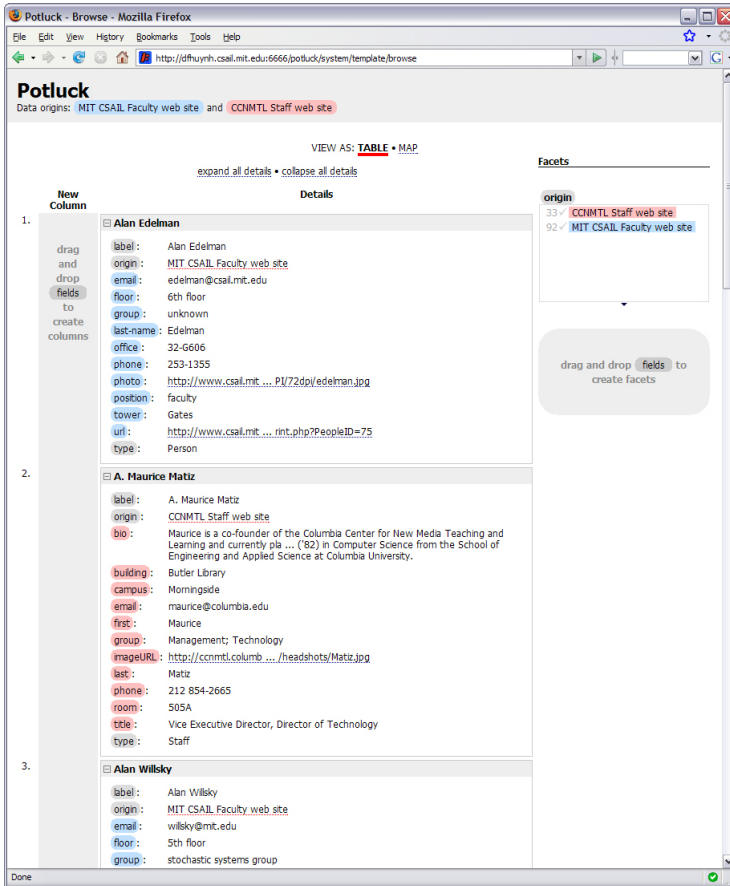


Fig. 2. Potluck’s user interface shows data that has just been mixed together but not yet processed by the user. Fields are rendered as draggable “field tags,” color-coded to indicate their origins. There are two drop target areas for creating columns and facets.

automatically merging them together allows the user to make the decision whether or not to merge.

**Creating columns and facets.** A field tag can be dragged and dropped onto the gray column to the left (Figure 2) to create a new column listing that field, or onto the gray box to the right to create a facet for filtering by that field. Figure 3 shows a newly created column. A column or facet can be moved by dragging its field tag and dropping the tag between other columns or facets. Deleting a column or facet (by clicking its ✕) removes the column or facet from the display but does not delete the corresponding field’s data.

**Merging fields.** A field tag can be dropped onto an existing column or facet in order to make that column or facet contain data for both the original field and the newly dropped field. Such an operation creates a *merged field*, whose field tag is rendered as a visual juxtaposition of the original tags, taking on a pill-shaped form `position title`. Figure 4 shows several columns and facets of merged fields. Merged field tags can be dragged and dropped just like elemental field tags can in order to create new columns and facets, or to merge into other existing columns and facets.

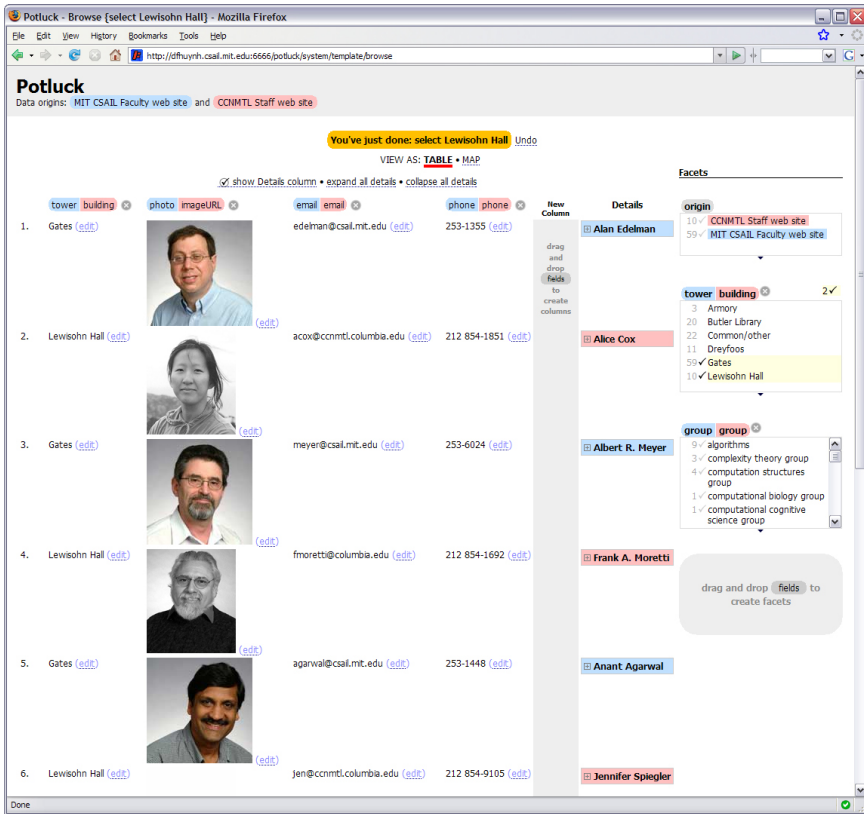
Creating a merged field does not disturb the elemental fields. Thus, in the scenario, it would be easy to have `gen:mother` and `gen:father` merged together for one purpose while keeping them separate for another purpose, all at the same time. Furthermore, the merging operation is *not* transitive, so that, say, merging fields `mother` and `father` together (to mean `parent`) and then `mother` and `grandmother` together (to mean `female ancestor`) does *not* force all three fields to be merged into `mother/father/grandmother`.

**Simultaneous editing.** The edit link next to each field value opens up the Simultaneous Editing dialog box where the values of that field can be edited *en masse* (Figure 5). The concept of simultaneous editing originated from LAPIS [9], a text editor that displays several keyboard cursors simultaneously on a text document, generalizes the user’s editing actions at one cursor, and applies them to the text at the rest of the

The screenshot shows the Potluck interface. On the left, there is a 'New Column' drop target with a 'tower' field tag and an 'edit' link. Below it, three records are listed:

1. Gates (edit) - Alan Edelman
  - label: Alan Edelman
  - origin: MIT CSAIL Faculty web
  - email: edelman@csail.mit.edu
  - floor: 6th floor
  - group: unknown
  - last-name: Edelman
  - office: 32-G606
  - phone: 253-1355
  - photo: http://www.csail.mit...
  - position: faculty
  - tower: Gates
  - url: http://www.csail.mit...
  - type: Person
2. missing - A. Maurice Matz
  - label: A. Maurice Matz
  - origin: CCNMTL Staff web site
  - bio: Maurice is a co-founder Engineering and Apple
  - building: Butler Library
  - campus: Morningside
  - email: maurice@columbia.edu
  - first: Maurice
  - group: Management; Technol
  - imageURL: http://ccnmtl.columb...
  - last: Matz
  - phone: 212 854-2665
  - room: 505A
  - title: Vice Executive Director
  - type: Staff
3. Dreyfoos (edit) - Alan Wilsky
  - label: Alan Wilsky
  - origin: MIT CSAIL Faculty web
  - email: wilsky@mit.edu
  - floor: 5th floor
  - group: stochastic systems gro
  - last-name: Wilsky
  - office: 32-D582
  - phone: 253-2356
  - photo: http://www.csail.mit...
  - position: faculty

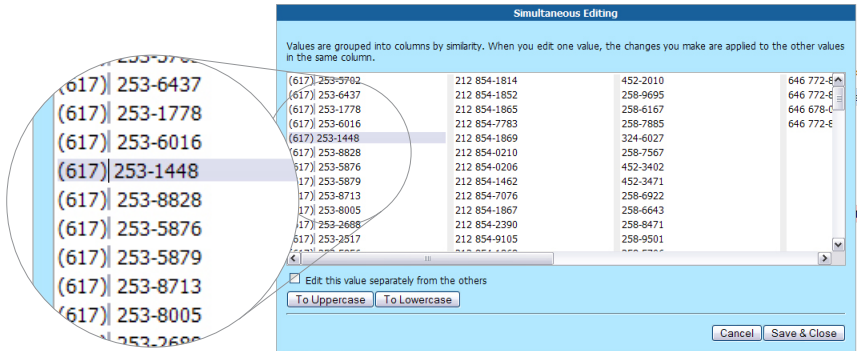
**Fig. 3.** Potluck renders a new column to the left when `tower` is dropped into the New Column drop target. Since the second record is not from the same origin as the dropped field, its cell in that column shows `missing`.



**Fig. 4.** A screen shot of Potluck showing several columns and facets of merged fields. The records’ details have been collapsed to make space for the columns.

ursors. Based on the user’s mouse clicks, LAPIS guesses how to divide the text document into records (often into lines or paragraphs) and where the cursors should be placed within those records (e.g., after the second word of the third sentence in each paragraph). Whereas LAPIS has to guess what a record is for the purpose of simultaneous editing, Potluck already has the field values conveniently separate. Potluck groups field values into columns by structural similarity, e.g., the phone numbers in the second column all have area code 212. These columns serve to visually separate out values of different forms, call out outliers (such as “Easter Day 1824” in the scenario), and let the user edit different forms differently. The user can click on any field value to give it keyboard focus, and editing changes made to it are applied to other values in the same column in a similar fashion. The multiple cursors in Figure 5 give visual feedback of the simultaneous editing operations in progress.

If a value appears in several records it is shown in only one entry in the dialog box. In the scenario, if the nickname “Bill” appears in three records, the user can click on its single entry in the dialog box, set the checkbox Edit this value separately from the others, and change it to “William” to correct all three records.



**Fig. 5.** Potluck’s Simultaneous Editing dialog box lets the user change several similar values simultaneously by editing any one of them. Multiple keyboard cursors are shown and any editing change to the focused value is immediately reflected in the other values.

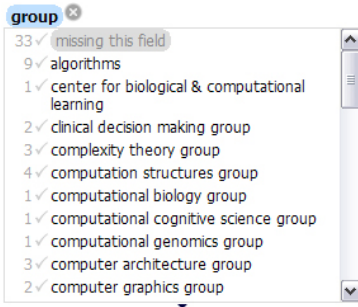
Simultaneous editing is useful for correcting inconsistencies between data sets that occur many times, such as prefixing area codes to phone numbers and wrapping existing area codes in parentheses. It is also useful for reformatting a field, such as changing “first-name last-name” into “last-name, first-name”, and for making a new field out of an existing field, such as extracting building numbers (32) from within office numbers (32-582).

**Faceted browsing** [19] is a browsing paradigm in which a set of records can be filtered progressively along several dimensions in any arbitrary order. For example, a set of recipes can be filtered by picking an ingredient first, a cooking method second, and a cuisine finally, or by picking a cuisine first, then an ingredient, and a cooking method finally depending on which order suits the user best. Because the data Potluck handles is often multidimensional, faceted browsing is useful in Potluck as it is designed for exploring multidimensional data in flexible, user-controllable ways. Exploration is needed for identifying and selecting out just the subset of data that is useful as well as for isolating on records that need cleaning up. All faceted browsers so far assume that they are fed data sets that have already been cleaned and made coherent using other tools. In Potluck, we extended faceted browsing for the mash-up task in which data arrives from many sources.

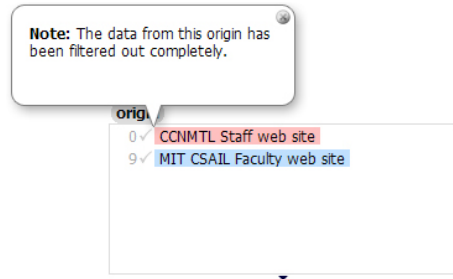
If within a facet there are records for which the corresponding field is missing, the facet explicitly shows a choice for filtering to those records (Figure 6). This visual element, not present in conventional faceted browsing interfaces, also serves to remind the user that, if that field is an elemental field instead of a merged field, the field is not present for records in other data sets.

While working with multiple data sets at the same time, it can be easy to forget that an elemental field from one data set does not exist in the others. Whenever a facet choice causes all records from an origin to be filtered out completely, that origin remains in the origin facet and a message is popped up drawing the user’s attention to it (Figure 7).

**Visualizations.** Potluck currently provides two visualizations: a tabular view and a map view. We have discussed the tabular view extensively while the map



**Fig. 6.** If inside a facet there are records for which the corresponding field is missing, the facet shows `missing this field` as a choice so that the user can get to those records



**Fig. 7.** The origin facet does not remove choices for which there are no records. Moreover, it pops up messages to call the user’s attention to those filtered out origins.

view can be seen in the screencast accompanying this paper. Any field containing street address or as latitude/longitude pair can be dropped onto the map view to plot the records without doing any programming.

**Miscellany.** Potluck provides drop down menus on left clicks as alternatives to drag and drop in order to increase the likelihood that the user succeeds at finding some way to accomplish a task. The browser’s Back and Forward buttons can be used to redo and undo user actions. Like contemporary highly interactive web interfaces, Potluck also shows the most recently done or undone action and provides a link to undo or redo it.

## 4 Implementation

Potluck consists of two components: a server-side component implemented as a Java servlet, responsible for retrieving the data within the Exhibit-embedding web pages to mix; and a client-side component implemented in Javascript on top of the Exhibit API [5], responsible for all the user interface interactivity.

Merged fields are implemented as query unions: when the values of a merged field are requested, the values of each elemental field in that merged field are returned in a single result set. No equivalence is added into the data model so that merging operations will not be transitive and so that the original elemental fields can still be used in isolation even after they have been merged.

Simultaneous editing is implemented in Javascript. Each field value is parsed into a sequence of features. Features are runs of digits, of letters, or of white spaces, or individual punctuation marks and symbols. For example, “733-3647” is broken down into three features: the run of digits “733”, the symbol “-”, and the run of digits “3647”. Field values are then clustered into columns by greedily aligning these sequences of features.

As the user moves the keyboard cursor, makes selections, and edits the text of one value, the cursor positions are generalized to be relative to the features of the field value being edited (e.g., “second character from the beginning of the third

last feature”), and then those generalized cursor positions are turned into absolute cursor positions of each of the other field values in the same cluster and used to apply the edit. Secondary cursors are rendered using colored `<span>` elements.

As the clipboard Cut and Paste operations cannot be reliably detected in web browsers, we must support cut-and-paste in the simultaneous editing paradigm using a trick. When some text is inserted, we check if that same piece of text has previously been deleted in one edit action and assume that what has taken place is a cut-and-paste operation. Note that this trick works only for cut-and-paste, not copy-and-paste.

## 5 Evaluation

We conducted a user study on Potluck to ascertain whether people could learn how to use Potluck as well as to discover usability problems. We also wanted to observe how people use Potluck in an open-ended task using their own judgement about which fields to merge and edit, and how to display them.

### 5.1 Design and Procedure

This study consists of two tasks: a structured task during which the subjects performed simple steps to familiarize themselves with Potluck, and an unstructured task during which the subjects performed an open ended task based on the skills they had just acquired.

In Task #1, subjects browsed two web pages containing information about 92 people in a lab and 33 people in another lab, and answered questions about these people in ways that required the pages’ faceted browsing features (e.g., “how many people are in the Gates tower?”). This warm-up exercise let the subjects learn about the data and about faceted browsing. Then the subjects were asked to use Potluck to mix the data in those pages and to achieve the following goals:

- create a column listing the buildings where people work and make sure the column is filled in with information for people from both labs;
- create a column listing people’s phone numbers and edit them to have the form (xxx) xxx-xxxx, using 617 for phone numbers without area code;
- create a column listing people’s job titles;
- create a facet of people’s job titles, use it to filter for people in directing positions (directors and co-directors), and determine how many such people there are in each lab; and
- create a column of people’s last names and sort it in ascending order.

These instructions were *not* worded in low-level details (e.g., click this button) so to allow the subjects the opportunities to learn how to use Potluck’s user interface by themselves and to allow us the chance to discover usability problems.

In Task #2, the subjects were asked to use Potluck to mix data from two Exhibit-powered web pages of 40 + 55 publications and then mock up a single web page where hypothetical visitors could conveniently sort and filter through all of those publications as if the data came from a single source. The subjects were left to their own discretion to decide which columns and facets to create, although some examples were given in case the subjects were not familiar with the domain.



## 5.2 Participants and Apparatus

Six subjects (2 male, 4 female) from a university community were recruited by sending an e-mail message to a mailing list and posting paper ads around our college campus. Four were younger than 30, and two older than 30. They were two students (mechanical engineering and computer science), two researchers (applied math and brain and cognitive science), a lawyer, and an applied math consultant.

We also recruited five subjects (1 male, 4 female) from our campus' libraries, who worked with data in their daily job. Two were in their 20s, one 30s, and two 40s. We wanted to observe if librarians, who have more experience working with data, would use Potluck differently.

There were a total of 11 subjects, referred to as G1 to G6 from the general university population and L1 to L5 from the libraries. All browsed the Web at least a few times a day and used Firefox as one of their primary browsers.

Subjects received \$10 each for participating in a 30 – 45 minute study session. All sessions were conducted by one investigator on a single computer (Pentium 4 2.53GHz, 1.00GB) with an 18" LCD flat panel at 1600x1200 resolution in 32-bit color and a Dell two-button mouse with wheel, running Microsoft Windows XP. The study facilitator observed the subjects and took written notes.

## 5.4 Results

All subjects were able to learn Potluck's user interface with little guidance and to complete the user study's tasks within 45 minutes. We now report the results in more details and point out usability issues to address in the future.

**Columns.** Nine subjects out of 11 used only drag and drop to create columns. This indicates that the relevant visual cues may be sufficiently strong. One of the other two subjects, G5, used the Create Column menu command at first but adopted drag and drop later. L1 used only the menu command.

G5 and L5 had difficulty understanding that dragging a field tag to create a column automatically filled up the whole column with data wherever the field was available. They continued to drag the same field tag out again and again for each row, paying no attention to the data already shown in the column. We can improve the drag feedback to better indicate that the whole field is being dragged, such as showing ghosted images of several field values near the mouse pointer.

All except one subject merged columns using drag and drop; G2 used the corresponding menu command. G3 and G4 expected the phone fields from both sources in Task #1 to be merged automatically. We can make Potluck suggest such merging if the field names match precisely.

Most subjects merged **position** and **title** together into one column, but one subject also included **group** to more fully qualify **position**. This was because most **title** values were more specific than most **position** values (e.g., "Codirector of Marketing" vs. "professor"). This operation was actually not what the subject intended (as he verbalized): the operation performed a set union of two fields instead of a string concatenation. But as Potluck rendered the **group** value after the **position** value for each record (e.g., "professor, computer architecture"), the visual outcome looked right and the subject was contented. However,



sorting on this merged field would produce random orders and a facet created out of this merged field would list the **group** and **position** values separately, not paired together. Potluck should support string concatenation and suggest it as an alternative to merging whenever the two fields involved come from the same source. Note that in the scenario in section 2, concatenation is probably not the desired choice when the **gen:mother** field is dropped onto the **gen:father** field even though both come from the same source.

**Facets.** All subjects used drag and drop to create facets. Two subjects initially created facets using the corresponding menu command, but they discovered the drag and drop alternative and did not revert to the menu. Merging facets was done solely using drag and drop. We note that the field tags on facets do not offer any menu (an oversight in our implementation); only field tags in the details column and in the column headers support menus.

Some subjects tended to drag already merged field tags from columns to create facets while the others dragged elemental field tags from the Details column to create merged facets. The latter behavior forced the user to re-merge fields she has already merged; this is both inefficient and error-prone as some subjects did forget to re-merge fields. Potluck should have automatically suggested or defaulted to the merged field whenever an elemental field that has been merged is used.

G4 did not initially merge facets in Task #1 to filter for people in directing positions. Instead, he created two facets, **position** and **title**, from the two sources separately and used `missing this field` to achieve the goal. In either facet, he selected directing positions as well as `missing this field` so that records in the other source were not excluded. This required on his part deeper understanding of how faceted browsing worked. When asked to achieve the goal without using `missing this field`, he discovered that he could merge facets.

**Simultaneous editing.** All subjects were able to edit several phone numbers using the simultaneous editing feature. G1 anticipated this feature even before clicking edit, asking out loud, “can I edit them all together?” She later used the feature to delete first names from people’s full names to get a field of last names. This action properly utilized the simultaneous editing feature’s power but destroyed data (the first names). We can make Potluck alert the user of this loss and offer a convenient way to apply the edit on a copy of the original field instead.

G4 tried to move the leading “A” from publication titles to the end (e.g., “Tale of Two Cities, A”) using simultaneous editing (a reasonable goal) but the facilitator explained that the feature did not support that case. L2 and G6 tried to swap first names and last names so that publications could be sorted by their authors’ last names. L2 selected a last name in the simultaneous editing dialog box and dragged it to the front of the corresponding first name; unfortunately, a bug prevented this from working. G6 used keyboard shortcuts for cut-and-paste and succeeded. These subjects’ actions indicated some intuitiveness in using cut and paste and drag-and-drop for simultaneous editing.

G3 expressed that she did not want to see all phone numbers in the simultaneous editing dialog box but only their templates. G5 and L3 edited only the first group of phone numbers, and L4 edited only the first and third groups, neglecting

the groups that were not scrolled into view. To avoid such oversight, we will need to make it more apparent which pieces of data an edit does and does not affect.

**Librarians vs. general subjects.** Among the five librarians, four were catalogers (who characterize physical artifacts such as books and enter their metadata into databases), and one was a programmer responsible for integrating large data sets. While the catalogers showed no significant difference with the general subjects in their use of Potluck, the programmer, L1, was clearly an outlier: he created 10 columns and 7 facets in total. He was very excited about the user interface of Potluck and described his data integration work, consisting of manual data entry and Perl scripting, to be tedious and painful.

G6, who also needed programming skills to deal with some data for his work, expressed equal enthusiasm for Potluck. He used simultaneous editing to swap first name and last name. Thus, while there was no noticeable difference between the subjects from the general population and the librarians, who purportedly work with data and metadata on a daily basis, there was a difference between programmers and non-programmers in how much they appreciated Potluck. Programmers, who have encountered difficulties in dealing with data even with their programming skills, appreciated Potluck more. Non-programmers accomplished the tasks in the study equally well, but were not equally excited perhaps because there was not enough reusable data on the Web for them to feel the need to mash up data themselves. However, when there will be more reusable data in the future, interfaces like that of Potluck have the potential to level the playing field for non-programmers, making them as effective as programmers for the task of mashing up data.

## 6 Related Work

Piggy Bank [6] and Tabulator [3] illustrate how personal mash-up tools with generic functionality can let individual Web users satisfy their own unique combinations of mash-up needs. While Tabulator consumes only RDF data, Piggy Bank can also run screen scrapers to extract data from HTML pages—a design to overcome the current shortage of RDF data on the Web. While the proliferation of structured data on the Web will hopefully eliminate the need to scrape fragile HTML, the data still has to be cleaned up and aligned before it can appear coherent to the user and thus become useful. The amount of broken HTML code at the present forebodes messy real-world RDF in the future, broken perhaps not just in syntax but also in semantics. Personal mash-up tools like Piggy Bank and Tabulator have largely ignored this problem, assuming that once data is in RDF, conventional visualizations and browsing techniques previously designed to work on individual coherent data sets can be applied readily on mashed up data.

An exception to these tools is WebScripter [18], which lets casual users create coherent reports out of data collected from several sources, offering data alignment features for that purpose. Although the desired target audience is casual users, WebScripter's interface is still expert-oriented, full of jargon such as "DAML", "class", "instance", etc. WebScripter offers no feature for fixing data at the syntactic level (e.g., swapping first name and last name) and it has not been formally evaluated on actual users.

There are also web applications such as Dapper [1] that let users scrape existing web sites for data and serve that data up in structured formats as “feeds,” or make use of data already scraped by other people. These web applications still offer very limited capabilities for cleaning up data, integrating data in different schemas, and constructing rich visualizations. Those that offer more capabilities, such as Ning [2], require programming.

In research, data alignment tools have been built mostly for experts and research has focused primarily on data modeling theories and automated agents for ontology alignment [7, 8, 10, 11, 12, 13, 14, 16] rather than on user interfaces for making practical use of aggregated data. Because they specialize only in data alignment, they implicitly assume that users work with the data in delineated stages, first aligning the data and cleaning it up, and then making use of that data in some other tools. We believe that users actually work iteratively on data, switching from aligning and clean up the data to using the data, and back, as they get to know the data better over time. Furthermore, these tools tend to work on ontological abstractions, basing their interface interactions on concepts such as classes. Casual users have little knowledge about data modeling and ontological abstractions, and little interest in learning.

Faceted browsing was pioneered by Yee et. al. [19] and recently adopted by the Semantic Web research community [4, 15, 17]. We extended faceted browsing to allow selection of records that are missing data. Simultaneous editing has been researched previously [9] and we extended it with the automatic clustering of field values into columns.

## 7 Conclusion

In this paper, we presented Potluck, a tool for casual users—those without programming skills and data modeling expertise—to mash up data by themselves. Potluck is novel in its use of drag and drop for merging fields, its integration and extension of the faceted browsing paradigm for focusing on subsets of data to align, and its application of the simultaneous editing technique for cleaning up data syntactically. Potluck also lets the user construct rich visualizations of data in-place as the user aligns and cleans up the data. This iterative process of integrating the data while constructing useful visualizations is desirable when the user is unfamiliar with the data at the beginning—a common case—and wishes to get immediate value out of the data without having to spend the overhead of completely and perfectly integrating the data first. A user study on Potluck indicated that it was usable and learnable, and solicited excitement from programmers who, even with their programming skills, had great difficulties in integrating data.

## Acknowledgements

We are grateful to the user study subjects for their time and their feedback. We thank the members of the Haystack group, the User Interface Design group, and the Simile team for their insightful comments on our work. This work was supported by the National Science Foundation (under award number IIS-0447800), by Nokia, and by the Biomedical Informatics Research Network. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the funders.

## References

- [1] Dapper: The Data Mapper, <http://www.dapper.net/>
- [2] Ning - Create your own Social Networks! <http://www.ning.com/>
- [3] Berners-Lee, T., et al.: Tabulator: Exploring and Analyzing linked data on the Semantic Web. In: SWUI 2006 (2006)
- [4] Hildebrand, M., Hardman, L.: Facet: A Browser for Heterogeneous Semantic Web Repositories. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
- [5] Huynh, D., Karger, D., Miller, R.: Exhibit: Lightweight Structured Data Publishing. In: WWW 2007 (2007)
- [6] Huynh, D., Mazzocchi, S., Karger, D.: Piggy Bank: Experience the Semantic Web Inside Your Web Browser. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
- [7] Kalfoglou, Y., Schorlemmer, M.: Ontology Mapping: The State of the Art. *The Knowledge Engineering Review* 18(1) (2003)
- [8] Laera, L., Tamma, V., Euzenat, J., Bench-Capon, T., Payne, T.: Reaching Agreement over Ontology Alignments. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
- [9] Miller, R., Myers, B.: Multiple Selections in Smart Text Editing. In: IUI 2002 (2002)
- [10] Mitra, P., Noy, N.F., Jaiswal, A.R.: Ontology Mapping Discovery with Uncertainty. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, Springer, Heidelberg (2004)
- [11] Mocan, A., Cimpian, E., Kerrigan, M.: Formal Model for Ontology Mapping Creation. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
- [12] Noy, N.F., Musen, M.A.: SMART: Automated Support for Ontology Merging and Alignment. In: Workshop on Knowledge Acquisition, Modeling and Management (1999)
- [13] Noy, N.F.: Semantic Integration: A Survey Of Ontology-Based Approaches. SIGMOD Record, Special Issue on Semantic Integration 33(4) (December 2004)
- [14] Noy, N.F., Musen, M.A.: The PROMPT Suite: Interactive Tools For Ontology Merging And Mapping. *International Journal of Human-Computer Studies* (2003)
- [15] Oren, E., Delbru, R., Decker, S.: Extending faceted navigation for RDF data. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
- [16] Scharffe, F.: Schema Mappings for the Web. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
- [17] Schraefel, M.C., Smith, D., Russel, A., Owens, A., Harris, C., Wilson, M.: The mSpace Classical Music Explorer: Improving Access to Classical Music for Real People. In: MusicNetwork Open Workshop, Integration of Music in Multimedia Applications (2005)
- [18] Yan, B., Frank, M., Szekely, P., Neches, R., Lopez, J.: WebScripter: Grass-roots Ontology Alignment via End-User Report Creation. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, Springer, Heidelberg (2003)
- [19] Yee, P., Swearingen, K., Li, K., Hearst, M.: Faceted Metadata for Image Search and Browsing. In: CHI 2003 (2003)

# An Empirical Study of Instance-Based Ontology Matching

Antoine Isaac<sup>1,2</sup>, Lourens van der Meij<sup>1,2</sup>, Stefan Schlobach<sup>1</sup>,  
and Shenghui Wang<sup>1,2</sup>

<sup>1</sup> Vrije Universiteit Amsterdam

<sup>2</sup> Koninklijke Bibliotheek, Den Haag

{aisaac,lourens,schlobac,swang}@few.vu.nl

**Abstract.** Instance-based ontology mapping is a promising family of solutions to a class of ontology alignment problems. It crucially depends on measuring the similarity between sets of annotated instances. In this paper we study how the choice of co-occurrence measures affects the performance of instance-based mapping.

To this end, we have implemented a number of different statistical co-occurrence measures. We have prepared an extensive test case using vocabularies of thousands of terms, millions of instances, and hundreds of thousands of co-annotated items. We have obtained a human Gold Standard judgement for part of the mapping-space. We then study how the different co-occurrence measures and a number of algorithmic variations perform on our benchmark dataset as compared against the Gold Standard.

Our systematic study shows excellent results of instance-based matching in general, where the more simple measures often outperform more sophisticated statistical co-occurrence measures.

## 1 Introduction

Dating as far back as the problems of record and database schema integration, studied for well over 40 years now, the semantic heterogeneity problem is probably the single-most urgent problem to be solved to realize a web-scale Semantic Web. A huge number of ontologies is now available [1]. This makes automatic ontology mapping [2] as the anticipated solution to semantic heterogeneity, is therefore a research issue of paramount importance. To address it the Semantic Web community has invested significant efforts over the past few years. This has led to the development of a plethora of high-quality matching software, whose potential has been proven in specific applications in a variety of domains.

Ontology mapping techniques are commonly divided into 4 broad categories [1]: lexical (detecting similarities between labels of concepts), structural (using

---

<sup>1</sup> Swoogle has indexed over 10,000 of them, cf <http://swoogle.umbc.edu/>.

<sup>2</sup> *Ontology mapping* is the task of determining relations such as equivalence or subsumption between concepts of two separate ontologies.

the structure of the ontologies), based on background knowledge, and instance-based mapping (using classified instance data). Among these, there are surprisingly few systematic studies of instance-based ontology mapping, *i.e.* the construction of links between concepts based on the co-occurrence of instances. In Chapter 6.2 of [1] a number of systems are discussed which make use of extensional information. However, to the best of our knowledge there has been no systematic evaluation yet over the use of different similarity measures for instance-based mapping, and this paper attempts to close this gap.

The basic idea of instance-based mapping is that the more significant the overlap of common instances of two concepts is, the more related these concepts are. The difficult question is how to define the notion of significance for such *extension* overlap. Previous investigations on instance-based mapping [2,3] have shown that there are some crucial decisions to be made with this respect. We propose a systematic approach considering the following dimensions:

- **Measures:** the most simple idea is to calculate the common factor of two concepts  $C$  and  $D$  as the proportion of jointly annotated books over the sum of books annotated by  $C$  and  $D$ , as done by the Jaccard measure. In statistics and Information Theory a number of other measures have been developed, such as Pointwise Mutual Information, Information Gain or Log-likelihood ratio.
- **Thresholds:** often the above mentioned measures are vulnerable for data-sparseness: if there are too few instances, the common factor measure ranks mappings high when the two concepts involved can only be found in one single book's annotation. The solution to dealing with this issue is to consider thresholds in the measures.
- **Hierarchy:** following the semantics of ontologies we can use the hierarchy, *i.e.* including the instances of descendants in the extension of a concept.

In this paper we will study the effect of these choices on a critical application in which it is considered to combine two thesauri. We implemented a system that calculates ranked lists of mappings according to 5 measures and different thresholds. It also allows us to include instances from a concept's descendants into its extension. We evaluated the resulting mappings against a Gold Standard built manually.

Based on this case-study we will answer the following research questions

1. Is instance-based mapping a reliable technology to be applied in practical, possibly critical applications?
2. Which combination of measures, thresholds and information inclusion works best, possibly depending on circumstances such as whether precision or recall is considered more important?

The first question can be answered easily: our results show an excellent level of quality. The second, more technical question will be answered in the course of the paper.

It is worth emphasising that we make the non-trivial assumption that doubly annotated instances exist. Furthermore, note that we evaluate the quality of

the similarity measures rather than compare performances of (existing or new) mapping systems. For a discussion on the use of different measures and methods for possible application scenarios for mappings we refer to [4].

The paper is structured as follows. In Section 2 we introduce our application. In Section 3 we describe the methodology of our mapper, including the different measures and parameters. In the remaining sections we will describe our experiments and the results before Section 6 sums up our findings, and presents perspectives on future work.

## 2 Use Case Scenario

The National Library of the Netherlands<sup>3</sup> maintains a large number of collections. Two of them are the *Deposit Collection*, containing all the Dutch printed publications (one million items), and the *Scientific Collection*, with about 1.4 million books mainly about the history, language and culture of the Netherlands. Each collection is described according to its own indexing system. On the one hand, the Scientific Collection is described using the GTT, a huge vocabulary containing 35,000 general terms ranging from *Wolkenkrabbers* (Skyscrapers) to *Verzorging* (Care). On the other hand, the books contained in the Deposit Collection are mainly indexed against the *Brinkman thesaurus*, containing a large set of headings (more than 5,000) that are expected to serve as global subjects of books. Both thesauri have similar coverage but differ in granularity. Also, for each concept, they provide the usual lexical and semantic information found in thesauri: synonyms and notes, broader and related concepts, etc.

The co-existence of these different systems, even if historically and practically justified, is not satisfactory from the point of view of interoperability. The KB is therefore investigating ways to combine the two thesauri, trying to enhance integration while ensuring compatibility with legacy data of both systems. For this reason, mapping GTT concepts with Brinkman concepts is crucially needed.

Finally, it is important to mention that around 250,000 books are common to the depot and scientific collections, and have therefore been manually annotated with both GTT and Brinkman vocabularies. This makes the KB use case especially suitable for studying instance-based mapping techniques.

## 3 A Framework for Instance-Based Mapping

We will now describe our formal framework for instance-based mappings, slightly adapting the one presented in [5]. Given two ontologies  $\mathcal{S}$  (for source) and  $\mathcal{T}$  (target) we see a mapping as a triple  $(S, T, R)$ , where  $R$  is a relation between concepts  $S \in \mathcal{S}$  and  $T \in \mathcal{T}$ . Often, the relation  $R$  is taken from the set  $\{\equiv, \sqsubseteq, \sqcap, \perp\}$ , resp. for equivalence, subsumption, overlap and disjointness. In an application about thesauri [6], relations similar to *broader than*, *narrower than*, and even the *related to* relation might also be considered.

<sup>3</sup> Koninklijke Bibliotheek (KB), <http://www.kb.nl>



In instance-based mapping semantic relations between concepts of two ontologies are determined based on the overlap of their instance sets. This is a very natural approach, as in most ontology formalisms the semantics of the relations between concepts is defined via the set of their instances. The idea for mapping is then simply that the higher the ratio of co-occurring instances for two concepts, the more related they are.

As instance-based mapping is closely depending on the meaning of a concept in an ontology formalism, **different ways of interpreting concepts** have to be taken into account. The most prominent question is whether a concept is interpreted as the collection of instances annotated by itself alone, or whether the instances of its descendants in the hierarchy also belong to its extension.

Unfortunately, in the real world we also have to deal with incorrectly annotated instances, data sparseness and ambiguous concepts, so that basic statistical measures of co-occurrence, such as the Jaccard measure, might be inappropriate if applied in a naive way.

We deal with this problem in two ways: first **we use other measures for calculating relatedness** of sets based on their elements, such as Pointwise Mutual Information, Information Gain or Log-Likelihood ratio, which have been developed in information theory and statistics. Finally, we **consider statistical thresholds** which explicitly exclude statistically unreliable information.

This analysis immediately suggests a systematic study of different instance-based mapping paradigms according to three dimensions:

Measures — Hierarchy — Thresholds

We will use these in the following sections to answer the research questions we outlined in the Introduction, based on a set of systematic experiments. First however, let us briefly fix some technical terms we use later in the paper.

### 3.1 Measures

We use similarity measures to order pairs of proposed mappings according to the strength of their relatedness, and in our experiments we assess the ranking rather than the objective values. Therefore, we do not need any special normalisation of measures, nor require them to be within the 0-1 interval.

In the following we will call the set of instances annotated by a concept  $C$  its extension, and abbreviate by  $C^i$ . As usual the cardinality of a set  $S$  is denoted by  $|S|$ .

**Jaccard Measures.** The first candidates are functions that measure the fraction of instances annotated by both concepts relative to the set of instances annotated by either one of the concepts.

*Jaccard.* The first measure

$$JC(S, T) = \frac{|S^i \cap T^i|}{|S^i \cup T^i|} \quad (1)$$



is known as the Jaccard measure. If there is a perfect correlation between two concepts, the measure will have a value of 1; if there is no co-occurrence, the measure will be 0. An evident problem with this measure is that it does not distinguish between two matches  $(S, T), (S', T')$  where the first tuple co-occurs in 100 instances while the second is based on a single book containing  $(T'_1, T'_2)$  in both cases with no other occurrences of both concepts. Yet, a mapping based on one instance gives intuitively less evidence for equivalence than the case based on 100 different books.

*Corrected Jaccard.* To correct this, we define *corrected Jaccard* with the goal of assigning a smaller score to less frequently co-occurring annotations. We (relatively arbitrary) choose a factor of 0.8 so that evidence based on one co-occurring instance is weighed as much as mapping two concepts would get when a large number of concepts have 20% in their intersection.

$$JC_{corr}(S, T) = \frac{\sqrt{|S^i \cap T^i| \times (|S^i \cap T^i| - 0.8)}}{|S^i \cup T^i|} \tag{2}$$

**Standard Information-Theory Measures.** Similarity measures for concepts based on annotations is not new, and often standard statistical measures have been applied to extract semantics from natural language texts based on co-occurrence of terms. As the problem is closely related to mapping concepts, we consider three of those measures: Pointwise Mutual Information, Log-Likelihood ratio and Information Gain.

*Pointwise Mutual Information.* Pointwise Mutual Information measures the reduction of uncertainty that the annotation of one concept yields for the annotation with the other. For mapping we use co-occurrence counts to estimate probabilities:

$$PMI(S, T) = \log_2 \frac{|S^i \cap T^i| \times N}{|S^i| \times |T^i|} \tag{3}$$

where  $N$  is the number of annotated instances.

*Log Likelihood ratio.* In the context of word co-occurrence in corpora it was noticed that PMI is inadequate to deal with sparse data [7]. Data sparseness is also a problem in our case, because the set of annotated objects is often small as compared to the size of the ontologies.

For the likelihood ratio, we compare the hypothesis that  $p_1$  is the maximum likelihood estimate of the probability  $P(i_2|i_1) = \frac{k_1}{n_1}$  and that  $p_2$  is the maximum likelihood estimate of the probability  $P(i_2|\neg i_1) = \frac{k_2}{n_2}$ , with the hypothesis that  $p_1 = p_2 = P(i_2) = \frac{k_1+k_2}{n_1+n_2}$ , which is just the maximum likelihood estimate of the probability of  $i_2$ .

In order to scale this ratio to make comparison possible, we use the log-likelihood form  $-2 \log \lambda$ . Thus, for our particular situation, we compute:

$$-2[\log L(p_0, k_1, n_1) + \log L(p_0, k_2, n_2) - \log L(p_1, k_1, n_1) - \log L(p_2, k_2, n_2)]$$

$$\begin{aligned} \text{where } \log L(p, k, n) &= k \log p + (n - k) \log(1 - p) \\ \text{and } k_1 &= |S^i \cap T^i| & k_2 &= |S^i| - |S^i \cap T^i| & n_1 &= |T^i| \\ \text{and } n_2 &= N - |T^i| & p_1 &= k_1/n_1 & p_2 &= k_2/n_2 & p_0 &= |S^i|/N. \end{aligned}$$

*Information Gain.* Information gain is the difference in entropy, i.e. the amount of information we can gain about a hypothesis by observing, and is used in decision trees learning to determine the attribute that distinguishes best between positive and negative examples.

In ontology mapping the analogy is the following: Information Gain describes the in- or decrease of the difficulty of assigning a concept to an instance if it has already been annotated with a concept from the other ontology. Formally, the entropy of assigning a concept  $T$  to an instance  $i$  can be estimated by  $e_1 = -\frac{|T^i|}{N} \times \log_2\left(\frac{|T^i|}{N}\right)$ , where  $N$  is again the number of instances. After assigning a concept  $S$  from the source ontology the entropy  $e_2 = -\frac{|S^i \cap T^i|}{|S^i|} \times \log_2\left(\frac{|S^i \cap T^i|}{|S^i|}\right)$ . The information gain is then  $IG = e_1 - e_2$ .

For Information Gain the order of source and target are of crucial importance. For mapping targeting equivalence and relatedness, however, we do not have to take symmetry information into account. The version used in our experiments is a combination of two IG measures:  $IGB(S, T) = \max\{IG(S, T), IG(T, S)\}$ .

### 3.2 Enforcing Thresholds to Guarantee Statistical Relevance

Both Log-likelihood and Information Gain take the number of instances of a concept into account to ensure statistical viability of its results. An alternative approach is to set a threshold for discarding computation of measures if the extension of one of the concepts is too small. The aim of this study is not to find the ideal threshold for statistical relevance, because this will probably too strongly depend on the collection of instances. However, we want to empirically show that there is a difference between using a threshold, and not using a threshold. Therefore, we only consider values 1 and 10 for cut-off, and denote a measure  $M$  with cut-off as  $M_{10}$ .

### 3.3 Hierarchical Information

For all the measures we previously defined, we used as the extension of a concept  $C$  the set  $C^i$  of its direct instances, i.e. the set of books explicitly annotated with it. However, semantically, this is not the only option, as one could also take more information from the ontology into account. Especially, a *broader than* relation could imply that the instances of the more specific concept are also instances of the more general one. The alternative definition of the extension  $C_{alt}^i$  of a concept  $C$  is then defined as  $\bigcup_{D \sqsubseteq C} D^i$ . We will refer to a measure  $M$  based alternative extension as *Mhier*.

### 3.4 Calculating Mappings from Rankings

Once decided on a suitable measurement we order mappings according to their degree of relatedness. From such an ordering we can derive all sorts of mappings,

such as 1-1 or 1-n mappings.<sup>4</sup> In practise one also has to choose a cut-off point, *i.e.* a value of the measure, below which a mapping of two instances will be considered too unreliable.

As both the choice of cut-off and 1-n mappings is strongly application-specific, in our experiments we evaluate more generally. Instead of evaluating a particular mapping based on a particular setting we assess the quality of the ranking, *i.e.* we calculate whether a mapping suggested in a particular position in the ordering induced by the measure is correct or not.<sup>5</sup>

## 4 Experimental Setup

In our experiments we used the 5 measures described in the previous section: Jaccard, corrected Jaccard, PMI, LLR, and IGB, as well as hierarchical and non-hierarchical extensions, and two alternative thresholds (1 and 10) to deal with statistically insignificant information. Having calculated the similarity between all pairs of concepts from GTT and Brinkman we then rank these pairs of concepts based on their similarity measure in a descent order. In Section 5, we will give comparison of precision and recall of our experiments with respect to these different options.

### 4.1 Dataset and Types of Mappings

In our dataset, there are 243,886 books which were doubly annotated with concepts from GTT and Brinkman. In total 24,061 GTT concepts and 4,990 Brinkman concepts have been used for annotation. For each GTT and Brinkman concept, we treated the books annotated by this concept as its instances. As both ontologies are thesauri, we expect our target mapping relations, beyond the expected “equivalent to”, to be the usual thesaurus semantic relations “broader than”, “narrower than” and/or “related to”.

### 4.2 Evaluation Methods

To be able to estimate the quality of a mapping, we need an evaluation procedure. For each measure we calculate four ordered lists, two taking the hierarchy into account, two not, of which one is based on a threshold of 1, and one on a threshold of 10. To get a better understanding on the difference between the measures, we calculate the overlap of the different lists. Table 1 shows the percentage of shared mappings between the ranked lists generated by all similarity measures up to the first 10,000 mappings.

<sup>4</sup> A 1-n mapping can be obtained as follows: for a source concept  $S$  let  $(S, T)$  be the first pair in the ordering. Then all pairs  $(S', T)$  for  $S \neq S'$  are deleted from the list. 1-1 mappings can be created by deleting all  $(S, T')$  for  $T \neq T'$  as well. Other cardinality choices are possible, including m-1 and m-n (“many-to-many”) mappings.

<sup>5</sup> As an indication, we will sometimes use specific cut-offs of 100, 1,000, and 10,000 mappings, which makes the comparison of different measures easier. These numbers are relatively arbitrary, though.

**Table 1.** Comparison between top 10,000 mappings generated by our original measures

JC				
$JC_{corr}$	80%			
LLR	39%	46%		
IGB	15%	15%	9%	
MI	37%	28%	10%	10%
	JC	$JC_{corr}$	LLR	IGB

Table 1 shows a surprisingly big difference in the lists of mappings found using the different measures. This shows that there are indeed significantly different, and a systematic evaluation will be of crucial importance.

Due to the size and complexity of the task a complete evaluation of the correctness of the calculated mappings by domain experts is out of the question. As an alternative, we have developed an evaluation procedure consisting of three steps: **producing a Gold Standard**, calculating **average precision** and **recall approximation**. Part of this procedure is based on the simple, admittedly simplistic assumption that concepts with identical labels are equivalent.

**Producing a Gold Standard.** In order to evaluate the precision of the mappings generated by different measures, we first sampled the generated mappings to a reasonable size for human evaluation to produce a Gold Standard manually. For each list of mappings, we selected the top 100 mappings, every tenth mapping from the 101st to 1,000th mapping, and every 100th from 1,001st to 10,000th mappings. We filtered out all lexically equivalent mappings, since we already consider them to be valid. This produces 1,600 mappings for human evaluation.

The selected mappings were presented in random order to 3 Dutch native speakers who assigned relations “equivalent to,” “broader than,” “narrower than,” “related to,” “no link” and “do not know” to all pairs. The (online) evaluation we set out allowed evaluators to access, for the concepts involved in a mapping, both thesaurus information (*e.g.* broader concepts) and the books annotated with them.

Ordering mappings by similarity measure does not necessarily suggest an interpretation in terms of the target mapping relations “equivalent to,” “broader than,” “narrower than” and/or “related to.” Our evaluation allows us to consider that three different types of mapping are correct: we can consider the highly ranked mappings to be

1. equivalences only (ONLYEQ),
2. equivalent, broader or narrower relations, but not related-to (NOTREL)
3. all relations except explicit non-relatedness. (ALL)

Each way of *interpreting the nature of a found mapping* will have its use in practical applications<sup>6</sup> and conceptually we do not prefer any one over any

<sup>6</sup> Equivalence might be used, for instance, in a data translation application, while broader and narrower links can be exploited for hierarchical browsing, as in [8].

other. However, we will have to study the effect of choosing a particular semantic assumption in our experiments.

**Average Precision.** Since the mapping set for human judgement is only a sample of the whole generated mappings, we use the following equation to calculate the average precision up to the  $i$ th mapping:

$$P_i = \frac{N_{\text{good},i}}{N_i} \quad (4)$$

where  $N_i$  is the number of mappings which are evaluated up to  $i$ th mapping, while  $N_{\text{good},i}$  is the that of mappings which are evaluated as good ones.

**Recall Approximation.** A preliminary experiment using string comparison over concept labels shows 2,895 exact lexical matches (2,895) between GTT and Brinkman concepts, meaning that 8.2% of GTT concepts and 55.4% of Brinkman ones have a direct equivalent form in the other thesaurus.

This is quite a significant number, especially regarding the Brinkman thesaurus. As in our case lexically equivalent concepts are considered a perfect match, we argue that the recall value on lexically equivalent concepts can be used to approximate the absolute recall. Our approximation for recall, at the  $i$ th mapping, is thus  $R_i = \frac{N_{\text{lex},i}}{N_{\text{lex}}}$  where  $N_{\text{lex},i}$  is the number of lexically equivalent mappings among these top  $i$  mappings, and  $N_{\text{lex}}$  is the number of all lexical equivalences between these two thesauri.

Once precision and recall are calculated, the F-measure up to  $i$ th mapping is calculated as

$$F_i = \frac{2(P_i \times R_i)}{P_i + R_i}. \quad (5)$$

### 4.3 Goals of the Experiments

The overall goal of our study is to improve the understanding on the role of different measures and tunings on the process of instance-based mapping. This means first and foremost answering the question whether there is a **best** combination of measure, threshold and hierarchy, which outperforms all other combinations. Furthermore, we want to better understand the influence of the choice of measure and other parameters on the mapping. All this might depend on the interpretation of the found mappings, and we will have to study the effect of the assumptions made on the nature of the relations considered to be correct.

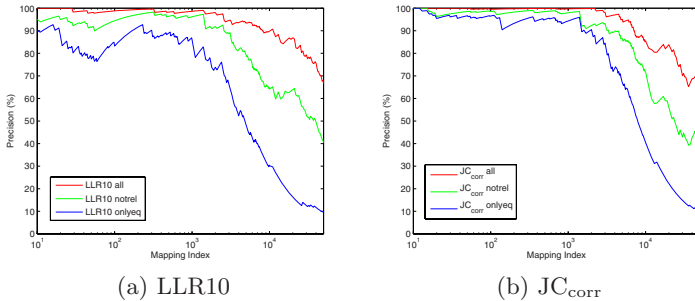
1. How does interpreting the nature of a found mapping influence results?
2. What is the influence of the choice of threshold?
3. What is the influence of using hierarchy information?
4. What is the best measure and setting for instance-based mapping?

## 5 Experimental Results

To answer the research questions mentioned in the previous section we performed a number of experiments in which we calculated precision, recall and f-measure.

## 5.1 The Influence of the Nature of a Mapping on the Results

Figure 1 shows the precision results when we use different criteria for “correctness” of mappings, *i.e.*, apart from the explicit “equivalent to” relation, whether we also count “broader than,” “narrower than” or “related to” as correct mappings. As mentioned above, ONLYEQ means only those mappings which were judged “equivalent to” are counted; NOTREL counts three kinds of relation but not “related to” relation; ALL counts every relation except “no link” nor “do not know” as correct.



**Fig. 1.** Comparing “ALL” “NOTREL” and “ONLYEQ” for LLR10 and JC<sub>corr</sub>

We give the results for two measures LLR10, which is the version of LLR with threshold 10, and JC<sub>corr</sub>, defining precision on the Y-axis in relation to the position in the ranking in logarithmic scale (X-axis), starting from 10 to filter the initial noise. The results for the other measures are comparable to these results. As there is a set inclusion between the sets of correct mappings, it is natural that the lines do not cross and that the top line describes the ALL-, the middle one the NOTREL- and the lower one the ONLYEQ relation.

What is more interesting is the differences between the figures. First, although LLR10 performs slightly better than JC<sub>corr</sub> on the ALL counts, the precision of LLR10 is worse than that of JC<sub>corr</sub> for ONLYEQ. What does this mean? It indicates that the LLR10 measure is more suitable to recognise related terms, whereas the stricter measure JC<sub>corr</sub> is better at recognising proper equivalences.

This indicates that the choice of measure has to depend on the application, and should be based on the interpretation of the nature of a found mapping. Despite the slight differences in the outcome mentioned above, we will in the following only present the results based on ONLYEQ for lack of space.

## 5.2 What Is the Influence of the Choice of Threshold?

An important problem in instance-based mapping is how to deal with sparse data. We have been discussing two approaches: using a threshold to exclude unreliable mappings, and statistical measures that can deal with uncertainty.

To study the effect of such a threshold, we ranked mappings according to our measures with and without a threshold. In Figure 2 we show the results for 2 measures JC and LLR, where the two dashed lines with dots are the versions with threshold, and the continuous line the Jaccard measure.

The following figures all have the same structure: the three graphs depict precision, recall and f-measure on the Y-axis versus the index of the mapping on the X-axis (which is given in a logarithmic scale).

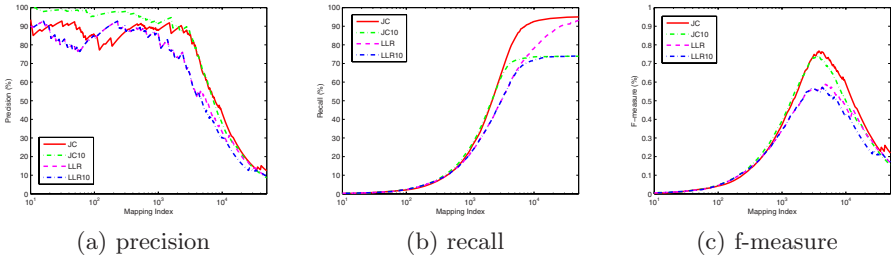


Fig. 2. Comparison with respect to the threshold

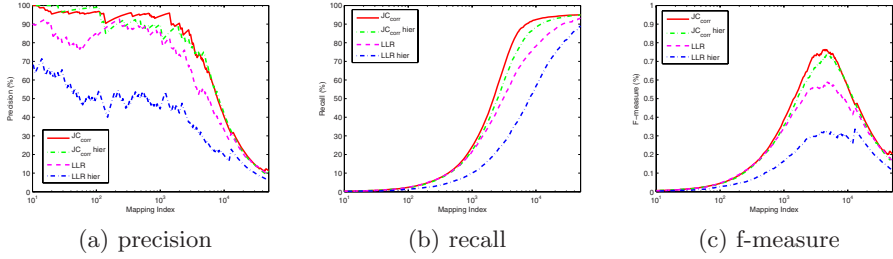
The results are in line with our expectation. As LLR has been developed for dealing with sparseness, we expect the difference between the version with and without the threshold to be more similar than this is the case for Jaccard. This shows clearly in the precision, which is almost the same for LLR and LLR10. What is also expected is the significant drop in recall for both measures with threshold at around 5000 mappings. Remember that choosing a threshold simply excluded the mappings from consideration, which will also exclude many correct mappings. Also, it is interesting to notice that when considering the ALL interpretation for mappings, the gain in precision is less significant for Jaccard. This shows that using threshold rather discards related concepts, for which co-occurrence evidence, even in a small number of items, is very often reliable.

The general lesson is that including a threshold generally improves precision but that there is a price in recall to be paid.

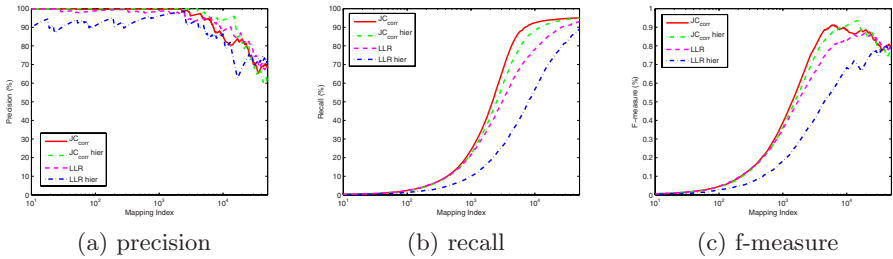
### 5.3 What Is the Influence of Using Hierarchy Information?

Ontology mapping is different from co-occurrence in texts in that the concepts are hierarchically organised. To find out what effect including this hierarchical information has on instance-based mapping we compared the four most promising measures with and without instances of the descendants in calculating the mappings. We also performed this experiment on different interpretations of the nature of a found mapping, and found quite diverse results.

Figure 3 shows the results of comparing JC<sub>corr</sub> and LLR with their versions JC<sub>corr</sub>hier and LLRhier, where we consider the ONLYEQ interpretation, *i.e.* we only consider those mappings to be correct that the evaluators have marked as equivalent. The most striking result is the gigantic drop in precision for LLRhier, as compared to JC<sub>corr</sub>, for which the results are very competitive when considering



**Fig. 3.** Comparison with respect to the hierarchical information (ONLYEQ)



**Fig. 4.** Precision comparison with respect to the hierarchical information (ALL)

hierarchy. Given that we only consider equivalence statements, this shows that including instances from descendants of concepts weakens the strength of equivalent concepts in the LLR measure.

To validate this assumption, we considered the same experiments with the ALL interpretation, *i.e.* we also accept related-terms and broader/narrower-than as correct mappings.

Figure 4 shows that this assumption is correct, as the drop in precision is much smaller given the more lenient interpretation of what a mapping is. Our general conclusion regarding hierarchical information: there is no significant improvement, and in most cases even a decrease in performance. A practical reason for this can also be found in the data itself. First, GTT and Brinkman include only few hierarchical links: almost 20,000 GTT terms have no parents. Second, GTT and Brinkman are thesauri, and as such their hierarchy can be interpreted as part-whole or as domain-object links. Examples for this would be “Bible” and “Gospel according to Luke.”

### 5.4 The Best Measure and Setting for Instance-Based Mapping

We can now finally answer the question which measure and tuning is best for instance-based mapping on our dataset. For this we considered the five measures with their ideal tuning, *i.e.* JC,  $JC_{corr}$ , LLR, PMI10 and IG10.

Figure 5 shows that the most simple measures JC and  $JC_{corr}$  have highest precision and recall at almost any mapping index, which is also reflected in the overall highest f-measures.



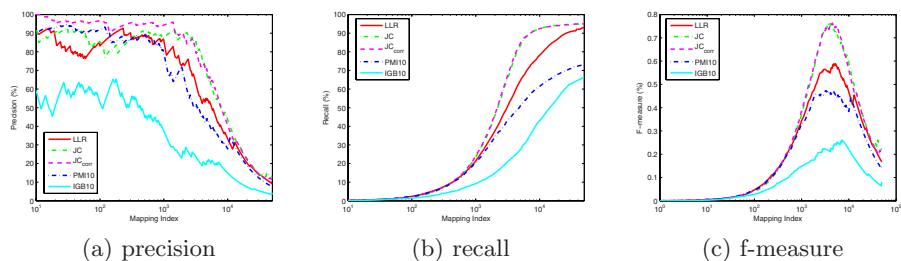


Fig. 5. Final comparison of the 5 best measures

We would like to finish the overview over our results with a general remark on the quality of our mappings. In general, apart from IGB, the results are surprisingly good, as compared to results from other ontology matching evaluations [9]. This indicates that instance-based matching is probably an easier task than structure-based or label-based mapping, but also indicates that our techniques will be suitable even in critical applications.

## 6 Conclusion

In this paper we presented an empirical study of instance-based matching based on a number of experiments performed on an application in the National Library of the Netherlands. We produced a Gold Standard for good mappings, and evaluated 5 different well-studied similarity measures, as well as two different ways to fine-tune them. All representations are, or course, based on Semantic Web standards.

We have to note that the complicated and very time consuming issue of evaluation was only touched marginally in the paper. Producing a gold standard is difficult and took us a long time, but the results remain sometimes controversial among domain experts. We will address the issue in more detail in future work.

The general results are very encouraging. For the first 1000 mappings the best available measure has a precision of over 90%, and at an estimated recall level of 70% we still have a precision of over 70%. Interestingly enough these results were not achieved by the complex statistical measures, but by an adapted version of the simple Jaccard measure.

The use of thresholds and hierarchical information had little influence in general, though the latter needs more study. The question here, and one that will probably apply to a number of our results, is how dependent the results are on our particular collection, and our ontologies.

For this reason we intend to conduct a similar analysis on other corpora, *e.g.* web directories or music classifications. We are confident, however, that general results are domain independent, and that instance-based mapping is a reliable and high-performing approach to ontology mapping.

## Acknowledgements

STITCH is funded by NWO, the Dutch Organisation for Scientific Research. We would like to thank Henk Matthezing for his constant advice at the KB, and Claus Zinn for his valuable comments on the manuscript.

## References

1. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer, Heidelberg (2007)
2. Vizine-Goetz, D.: Popular LCSH with Dewey Numbers: Subject headings for everyone. *Annual Review of OCLC Research* (1997)
3. Avesani, P., Giunchiglia, F., Yatskevich, M.: A large scale taxonomy mapping evaluation. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005*. LNCS, vol. 3729, Springer, Heidelberg (2005)
4. Isaac, A., Matthezing, H., van der Meij, L., Schlobach, S., Wang, S., Zinn, C.: The value of usage scenarios for thesaurus alignment in cultural heritage context. Under submission
5. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. *Journal of data semantics* 4, 146–171 (2005)
6. Doerr, M.: Semantic problems of thesaurus mapping. *Journal of Digital Information* 1(8) (2004)
7. Manning, C., Schütze, H.: *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA (1999)
8. van Gendt, M., Isaac, A., van der Meij, L., Schlobach, S.: Semantic Web Techniques for Multiple Views on Heterogeneous Collections: a Case Study. In: *10th European Conference on Digital Libraries (ECDL)*, Alicante, Spain (2006)
9. Euzenat, J., Mochol, M., Shvaiko, P., Stuckenschmidt, H., Šváb, O., Svátek, V., van Hage, W.R., Yatskevich, M.: Results of the ontology alignment evaluation initiative 2006. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006*. LNCS, vol. 4273, Springer, Heidelberg (2006)

# Finding All Justifications of OWL DL Entailments

Aditya Kalyanpur<sup>1</sup>, Bijan Parsia<sup>2</sup>, Matthew Horridge<sup>2</sup>, and Evren Sirin<sup>3</sup>

<sup>1</sup> IBM Watson Research Center, 19 Skyline Drive, Hawthorne NY 10532, USA  
adityakal@us.ibm.com

<sup>2</sup> School of Computer Science, University of Manchester, UK  
{bparsia,matthew.horridge}@cs.man.ac.uk

<sup>3</sup> Clark & Parsia LLC, Washington DC, USA  
evren@clarkparsia.com

**Abstract.** Finding the justifications of an entailment (that is, all the minimal set of axioms sufficient to produce an entailment) has emerged as a key inference service for the Web Ontology Language (OWL). Justifications are essential for debugging unsatisfiable classes and contradictions. The availability of justifications as explanations of entailments improves the understandability of large and complex ontologies. In this paper, we present several algorithms for computing all the justifications of an entailment in an OWL-DL Ontology and show, by an empirical evaluation, that even a reasoner independent approach works well on real ontologies.

**Keywords:** OWL Ontology Explanation, Debugging, Justifications.

## 1 Introduction

Since OWL became a W3C standard, there has been a notable increase in the number of people that are attempting to build, extend and use ontologies. To some extent, the provision of editing environments, visualization tools and reasoners has helped catalyse this. In traditional ontology editing environments, users are typically able to create ontologies and use reasoners to compute unsatisfiable classes, subsumption hierarchies and types for individuals. However, as ontologies have begun to be used in real world applications, and a broader audience of users and developers have been introduced to Ontology Engineering, it has become evident that there is a significant demand for editing environments which provide more sophisticated services.

In particular, the generation of explanations, or *justifications*, for inferences computed by a reasoner is now recognized as highly desirable functionality for both ontology development and ontology reuse. A clear demonstration of the need for *practical* explanation services manifested itself in the observations of the switching of users from Protege 3.2 [1] to Swoop [2] purely for the benefits of automatic explanation facilities [3].

As an example of explanation of entailments using Swoop, see Figure [4]. The left part of the Figure shows a justification for the entailed subsumption

Explanation	Explanation
<b>Axioms causing the inference</b> <b>Tears <math>\sqsubseteq</math> SecretedSubstance:</b> 1) (Tears $\sqsubseteq$ ( $\exists$ isActedOnSpecificallyBy . (( $\exists$ isFunctionOf . LachrymalGland) $\cap$ Secretion))) 2) $\perp$ (isActedOnSpecificallyBy $\sqsubseteq$ isActedOnBy) 3) (Tears $\sqsubseteq$ NAMEDBodySubstance) 4) $\perp$ (NAMEDBodySubstance $\sqsubseteq$ BodySubstance) 5) $\perp$ (BodySubstance $\sqsubseteq$ Substance) 6) (SecretedSubstance $\equiv$ (Substance $\cap$ ( $\exists$ isActedOnBy . Secretion)))	<b>Axioms causing the inference</b> <b>CorbansPrivateBinSauvignonBlanc rdf:type FullBodiedWine:</b> 1) (CorbansPrivateBinSauvignonBlanc rdf:type SauvignonBlanc) 2) $\perp$ ((SauvignonBlanc $\equiv$ (( $\exists$ madeFromGrape . {SauvignonBlancGrape}) $\cap$ $\leq$ 1 madeFromGrape) $\cap$ SemillonOrSauvignonBlanc)) 3) $\perp$ (madeFromGrape domain Wine) 4) (CorbansPrivateBinSauvignonBlanc hasBody Full) 5) (FullBodiedWine $\equiv$ (( $\exists$ hasBody . {Full}) $\cap$ Wine))

Fig. 1. Justifications in GALEN and Wine

*Tears*  $\sqsubseteq$  *SecretedSubstance* in the medical ontology GALEN<sup>1</sup>. The right part of the Figure shows a justification for why a *CorbansPrivateBinSauvignonBlanc* is entailed to be an instance of the concept *FullBodiedWine* in the Wine<sup>2</sup> Ontology. Both GALEN and Wine are expressive OWL-DL ontologies with non-trivial entailments, and understanding how these entailments arise becomes much easier using Swoop’s UI for displaying justifications (i.e., axioms responsible for the inference).

In general, the algorithms for finding justifications come in two flavors: *Black-box* and *Glass-box*:

- *Black-box (reasoner independent)* algorithms use the reasoner solely as a *sub-routine* and the internals of the reasoner do not need to be modified. The reasoner behaves as a “Black-box” that typically accepts as input, an ontology and a specific entailment test, and returns an affirmative or a negative answer, depending on whether the entailment holds in the ontology. In order to obtain justifications, the algorithm selects the appropriate inputs to the reasoner and interprets its output accordingly. While Black-box algorithms typically require many satisfiability tests, they can be easily and robustly implemented – they only rely on the availability of a sound and complete reasoner for the logic in question, and thus can be implemented on reasoners based on techniques other than tableaux, such as resolution.
- *Glass-box (reasoner dependent)* algorithms are built on existing tableau-based decision procedures for expressive Description Logics. Their implementation requires a thorough and non-trivial modification of the internals of the reasoner. For a tableau based system, these involve some form of tracing through the tableau. Tracing techniques have been used to find all justifications for the description logic (DL) *ALC* [4] and also single justifications with rather low overhead for *SHIF* (in our previous work [5]).

In this paper, we present a *practical* two stage Black-box technique for effectively finding *all* justifications for an entailment, and demonstrate its significance on a set of expressive, realistic OWL Ontologies. In addition, we describe and evaluate a faster hybrid solution that combines a Glass-box and Black-box approach to computing all justifications.

<sup>1</sup> <http://www.cs.man.ac.uk/~horrocks/OWL/Ontologies/galen.owl>

<sup>2</sup> <http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine.rdf>

## 1.1 Justification of Entailments

Informally, a justification is simply the precise set of axioms in an ontology responsible for a particular entailment. For example, if an ontology  $\mathcal{O}$  contains concepts  $A$ ,  $B$ , and  $A$  is inferred to be a subclass of  $B$ , i.e.,  $\mathcal{O} \models A \sqsubseteq B$ , then the justification for this concept subsumption entailment is simply the smallest set of axioms in  $\mathcal{O}$  responsible for it. It is important to realize that there may be more than one justification for a given entailment in the ontology. If *at least one* of the axioms in *each* of the justifications for an entailment is removed from the ontology, then the corresponding entailment no longer holds.

Justifications are formally defined as follows:

**Definition 1.** (*JUSTIFICATION*)

Let  $\mathcal{O} \models \alpha$  where  $\alpha$  is an axiom and  $\mathcal{O}$  is a consistent ontology. A fragment  $\mathcal{O}' \subseteq \mathcal{O}$  is a justification for  $\alpha$  in  $\mathcal{O}$ , denoted by  $\text{JUST}(\alpha, \mathcal{O})$ , if  $\mathcal{O}' \models \alpha$ , and  $\mathcal{O}'' \not\models \alpha$  for every  $\mathcal{O}'' \subset \mathcal{O}'$ .

We denote by  $\text{ALL\_JUST}(\alpha, \mathcal{O})$  the set of all the justifications for  $\alpha$  in  $\mathcal{O}$ .

While our goal, in general, is to compute all justifications for any arbitrary entailment in an OWL-DL ontology, we specifically focus on a particular type of entailment – concept unsatisfiability. This is because an OWL-DL ontology corresponds to a *SHOIN*( $\mathcal{D}$ ) knowledge base, and [6] shows that for every sentence (axiom or assertion)  $\alpha$  entailed by a *SHOIN*( $\mathcal{D}$ ) KB  $\mathcal{O}$ , there is always a class  $C_\alpha$  that is unsatisfiable w.r.t  $\mathcal{O}$ . Conversely, given any class  $C$  that is unsatisfiable w.r.t.  $\mathcal{O}$ , there is always a sentence  $\alpha_C$  that is entailed by  $\mathcal{O}$ .

As an example,  $\mathcal{O} \models A \sqsubseteq B$  iff the class  $A \sqcap \neg B$  is unsatisfiable in  $\mathcal{O}$ . Thus, the axioms responsible for the subsumption entailment  $\mathcal{O} \models A \sqsubseteq B$  are precisely the same set of axioms responsible for the concept unsatisfiability entailment  $\mathcal{O} \models A \sqcap \neg B \sqsubseteq \perp$ . Consequently, given an OWL-DL Ontology (*SHOIN*( $\mathcal{D}$ ) KB), the problem of finding all justifications for an arbitrary entailment reduces to the problem of finding all justifications for *some* unsatisfiable concept<sup>3</sup>.

Therefore, in the remainder of this paper, we shall restrict our attention, without loss of generality, to the problem of finding all justifications for an unsatisfiable concept w.r.t to a consistent *SHOIN*( $\mathcal{D}$ ) KB<sup>4</sup>.

## 2 Finding Justifications

In this section, we investigate the problem of computing justifications. First, we focus on the problem of finding *one* justification and then we discuss how to compute *all* of them. For such a purpose, we explore and discuss different techniques to tackle the problem. In any case, the process of finding justifications

<sup>3</sup> [6] describes the translation of a particular *SHOIN*( $\mathcal{D}$ ) entailment test to a concept satisfiability check.

<sup>4</sup> It should be emphasized that this reduction, in logics with nominals, such as *SHOIN*, holds for all standard entailments, including ABox statements. Thus our techniques extend to explaining inconsistent ontologies as well.

for an entailment consists of transforming the entailment to an unsatisfiable class (as described in the previous section) and then computing the justifications for that unsatisfiable class.

## 2.1 Computing a Single Justification

**A Black-box Technique.** The intuition behind our black box approach is simple: given a concept  $C$  unsatisfiable relative to  $\mathcal{O}$ , add axioms from  $\mathcal{O}$  to a freshly generated ontology  $\mathcal{O}'$  until  $C$  is found unsatisfiable with respect to  $\mathcal{O}'$ . We then prune extraneous axioms in  $\mathcal{O}'$  until we arrive at a single minimal justification. Thus, the algorithm consists of two stages: (i) “expand”  $\mathcal{O}'$  to find a superset of a justification and (ii) “shrink” to find the final justification. Each stage involves various satisfiability-check calls to the reasoner and the main aim of optimizations should be minimizing the number of satisfiability tests.

**Table 1.** Black Box Algorithm to find a Single Justification

<b>Algorithm:</b> SINGLE_JUST_ALG <sub>Black-Box</sub> <b>Input:</b> Ontology $\mathcal{O}$ , Unsatisfiable concept $C$ <b>Output:</b> Ontology $\mathcal{O}'$
(1) $\mathcal{O}' \leftarrow \emptyset$ (2) <b>while</b> ( $C$ is satisfiable w.r.t $\mathcal{O}'$ ) (3)     select a set of axioms $s \subseteq \mathcal{O}/\mathcal{O}'$ (4) $\mathcal{O}' \leftarrow \mathcal{O}' \cup s$ (5) perform fast pruning of $\mathcal{O}'$ using a sliding window technique (6) <b>for each</b> axiom $k' \in \mathcal{O}'$ (7) $\mathcal{O}' \leftarrow \mathcal{O}' - \{k'\}$ (8) <b>if</b> ( $C$ is satisfiable w.r.t. $\mathcal{O}'$ ) (9) $\mathcal{O}' \leftarrow \mathcal{O}' \cup \{k'\}$

This algorithm, which we refer to as SINGLE\_JUST\_ALG<sub>Black-Box</sub>( $C$ ,  $\mathcal{O}$ ), shown in Table 1, is composed of two main parts: in the first loop, the algorithm generates an empty ontology  $\mathcal{O}'$  and inserts into it axioms from  $\mathcal{O}$  in each iteration, until the input concept  $C$  becomes unsatisfiable w.r.t  $\mathcal{O}'$ . In the second loop, the algorithm removes an axiom from  $\mathcal{O}'$  in each iteration and checks whether the concept  $C$  turns satisfiable w.r.t.  $\mathcal{O}'$ , in which case the axiom is reinserted into  $\mathcal{O}'$ . The process continues until all axioms in  $\mathcal{O}'$  have been tested.

In between the two loops, lies a key optimization stage of fast pruning of  $\mathcal{O}'$  that is output at the end of the first loop. The idea here is to use a window of  $n$  axioms, slide this window across the axioms in  $\mathcal{O}'$ , remove axioms from  $\mathcal{O}'$  that lie within the window and determine if the concept is still unsatisfiable in the new  $\mathcal{O}'$ . If the concept turns satisfiable, we can conclude that at least one of the  $n$  axioms removed from  $\mathcal{O}'$  is responsible for the unsatisfiability and hence we insert the  $n$  axioms back into  $\mathcal{O}'$ . However, if the concept still remains unsatisfiable, we can conclude that all  $n$  axioms are irrelevant and we remove them from  $\mathcal{O}'$ .

**A Glass-box Technique.** In our previous work [5], it was shown that for the logic  $SHIF(\mathcal{D})$ , the computational overhead of tracking axioms internally within the tableau reasoner as they contribute to an inconsistency (a technique known as ‘tableau tracing’) is negligible, especially for time. That is, a satisfiability test while computing the justification (if the test is negative) is pragmatically as easy as performing the satisfiability test alone. Thus, if you have tracing implemented in your reasoner, it makes sense to use it, indeed, to leave it on in all but the most resource intensive situations.

We have improved on our tracing solution described in [5] by extending it to approximately cover  $SHOIN(\mathcal{D})$ . We say “approximately” since the final output of the tracing algorithm is not ensured to be the justification itself, but may include a few extraneous axioms due to non-deterministic merge operations caused by max-cardinality restrictions. These extraneous axioms are pruned out using the same algorithm which is used in the final stage of the Black-Box approach. In the remainder of this paper, we refer to our  $SHOIN(\mathcal{D})$ -extended Glass-box tracing algorithm as `SINGLE_JUST_ALGGlass-Box`.

## 2.2 Computing All Justifications

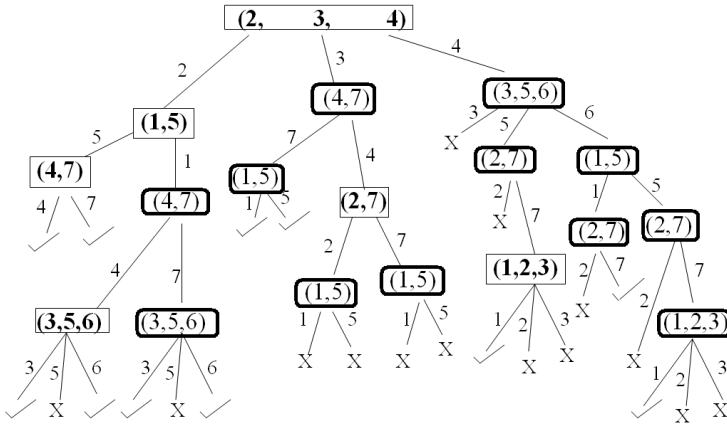
Given an initial justification, we can use other techniques to compute the remaining ones. A plausible one is to employ a variation of the classical Hitting Set Tree (HST) algorithm [7]. This technique is both reasoner independent (Black-box) and, perhaps surprisingly, practically effective.

**A Black-box Approach Using Reiter’s HST.** We first provide a short background on Reiter’s HST Algorithm and then describe how it can be used to compute all justifications.

Reiter’s general theory of diagnosis considers a system of components described by a *universal set*  $U$ , and a set  $S \subseteq \mathcal{P}U$  of *conflict sets* (each conflict set is a subset of the system components responsible for the error), where  $\mathcal{P}$  denotes the powerset operator. The set  $T \subseteq U$  is a *hitting set* for  $S$  if each  $s_i \in S$  contains at least one element of  $T$ , i.e. if  $s_i \cap T \neq \emptyset$  for all  $1 \leq i \leq n$  (in other words,  $T$  ‘hits’ or intersects each set in  $S$ )<sup>5</sup>. We say that  $T$  is a *minimal hitting set* for  $S$  if  $T$  is a hitting set for  $S$  no  $T' \subset T$  is a hitting set for  $S$ . The *Hitting Set Problem* with input  $S, U$  is to compute all the minimal hitting sets for  $S$ . The problem is of interest to many kinds of *diagnosis* tasks and has found numerous applications.

Given a collection  $S$  of conflict sets, Reiter’s algorithm constructs a labeled tree called *Hitting Set Tree* (HST). Nodes in an HST are labeled with a set  $s \in S$ , and edges are labeled with elements  $\sigma \in \bigcup_{s \in S} s$ . If  $H(v)$  is the set of edge labels on the path from the root of the HST to the node  $v$ , then the label for  $v$

<sup>5</sup> The significance of a hitting set for the collection of conflict sets is that in order to repair the system fully, at least one element in each of the conflict sets, i.e., its hitting set, needs to be removed from the ontology. Moreover, to keep system changes to a minimum, hitting sets should be as small as possible.



**Fig. 2. Finding all Justifications using HST:** Each distinct node is outlined in a rectangular box and represents a set in  $\text{ALL\_JUST}(C, \mathcal{O})$ . Total number of satisfiability tests is no. of distinct nodes (6) + nodes marked with '✓' (11) = 17.

is any  $s \in S$  such that  $s \cap H(v) = \emptyset$ , if such a set exists. If  $s$  is the label of  $v$ , then for each element  $\sigma \in s$ ,  $v$  has a successor  $w$  connected to  $v$  by an edge with  $\sigma$  in its label. If the label of  $v$  is the empty set, then  $H(v)$  is a hitting set for  $S$ .

In our case, the universal set describing the system corresponds to the total set of axioms in the ontology, and a justification (for a particular concept unsatisfiability) corresponds to a single conflict set. While Reiter’s algorithm is typically used to find all minimal hitting sets given a collection of conflict sets, it can also be used to dynamically find all conflict sets (justifications in our case), given the duality of the algorithm.

The idea is that given an algorithm to find a single justification for concept unsatisfiability, call it `SINGLE_JUST_ALG` (it could be either Black-box or Glass-box based as described in the previous section), we find any one justification and set it as the root node of the HST. We then remove each of the axioms in the justification individually, thereby creating new branches of the HST, and find new justifications along these branches on the fly (using `SINGLE_JUST_ALG`) in the modified ontology. This process needs to be exhaustively done in order to compute all justifications. The advantage of drawing parallels to Reiter’s algorithm is that we can make use of all the optimizations presented in the latter to speed up the search.

The following example illustrates the approach.

Consider an ontology  $\mathcal{O}$  with ten axioms and some unsatisfiable concept  $C$ . For the purpose of this example, we denote the axioms in  $\mathcal{O}$  as natural numbers. We now show how to combine HST and `SINGLE_JUST_ALG` to compute  $\text{ALL\_JUST}(C \sqsubseteq \perp, \mathcal{O})$ . Figure 2 illustrates the whole process for our example. We anticipate that the expected outcome is the following:

$$\text{ALL\_JUST}(C \sqsubseteq \perp, \mathcal{O}) = \{\{1, 2, 3\}, \{1, 5\}, \{2, 3, 4\}, \{4, 7\}, \{3, 5, 6\}, \{2, 7\}\}.$$



The algorithm starts by executing `SINGLE_JUST_ALG(C, O)` and let us assume that we obtain the set  $S = \{2, 3, 4\}$  as an output. The next step is to initialize a Hitting Set Tree  $\mathbf{T} = (V, E, \mathcal{L})$  with  $S$  in the label of its root, i.e.  $V = \{v_0\}$ ,  $E = \emptyset$ ,  $\mathcal{L}(v_0) = S$ . Then, it selects an arbitrary axiom in  $S$ , say 2, generates a new node  $w$  with an empty label in the tree and a new edge  $\langle v_0, w \rangle$  with axiom 2 in its label. Then, the algorithm invokes `SINGLE_JUST_ALG` with arguments  $C$  and  $O - \{2\}$ . In this case, it obtains a new justification for  $C \models \perp$  w.r.t.  $O - \{2\}$ , say  $\{1, 5\}$ . We add this set to  $S$  and also insert it in the label of the new node  $w$ .

The algorithm repeats this process, namely removing an axiom and executing the `SINGLE_JUST_ALG` algorithm to add a new node, until the concept turns satisfiable, in which case we mark the new node with a checkmark ‘ $\surd$ ’.

There are two critical optimizations in Reiter’s HST algorithm that help reduce the number of calls to `SINGLE_JUST_ALG`:

- **Early path termination:** Once a hitting set path is found, any superset of that path is guaranteed to be a hitting set as well, and thus no additional satisfiability tests are needed for that path, as indicated by a ‘ $X$ ’ in the label of the node. Moreover, if all possible paths starting with the current edge path have been considered in a *previous* branch of the HST, the current path can be terminated early. For example, in Figure 2 the first path in the right-most branch of the root node is 4,3 and is terminated early since the algorithm has already considered all possible paths (hitting sets) containing axioms  $\{3,4\}$  in an earlier branch.
- **Justification reuse:** If the current edge path in any branch of the HST does not intersect with a previously found justification, then that justification is directly added as the new node to the branch. This is because the edge path represents axioms removed from the ontology, and if none of these removed axioms are present in a particular justification, that justification is guaranteed to exist in the ontology. Thus, we do not need to call `SINGLE_JUST_ALG` again to re-compute this justification. In Figure 2, *oval-bordered* nodes of the HST represent reused justifications.

Using the above optimizations, the total no. of calls to `SINGLE_JUST_ALG`, as shown in Figure 2 is reduced from 47 (in the exhaustive case) to only 17.

When the HST is fully built, the distinct nodes of the tree collectively represent the complete set of justifications of the unsatisfiable concept.

**Definition of the Algorithm.** The main component of the algorithm, which we refer to as `ALL_JUST_ALG`, is the recursive procedure `SEARCH_HST`, that effectively traverses a hitting set tree. The algorithm proceeds using a *seed* justification which is obtained by the `SINGLE_JUST_ALG` algorithm. This seed is then used to construct a Hitting Set Tree (HST), which ultimately yields all justifications.

The correctness and completeness of the algorithm is given by Theorem 1.

**Theorem 1.** *Let  $C$  be unsatisfiable concept w.r.t  $O$ . Then,  $\text{ALL\_JUST\_ALG}(C, O) = \text{ALL\_JUST}(C, O)$ .*

ALL\_JUST\_ALG( $C, \mathcal{O}$ )

**Input:** Concept  $C$  and ontology  $\mathcal{O}$

**Output:** Set  $S$  of justifications

- (1)     **Globals:**  $S \leftarrow HS \leftarrow \emptyset$
- (2)      $just \leftarrow \text{SINGLE\_JUST\_ALG}(C, \mathcal{O})$
- (3)      $S \leftarrow S \cup \{just\}$
- (4)      $\alpha \leftarrow \text{select some } i \in just$
- (5)      $path \leftarrow \emptyset$
- (6)     SEARCH-HST( $C, \mathcal{O} \setminus \{\alpha\}, \alpha, path$ )
- (7)     **return**  $S$

SEARCH-HST( $C, \mathcal{O}, \alpha, path$ )

**Input:**  $C$  and  $\mathcal{O}$

$\alpha$  is the axiom that was removed

$path$  is a set of axioms

**Output:** *none* — modifies globals  $S, HS$

- (1)
- if**  $path \cup \{\alpha\} \subseteq h$  for some  $h \in HS$
- (2)     **or** there exists a prefix-path  $p$  for some  $h \in HS$  s.t.  
 $p = path$
- (3)     **return** (i.e., early path termination)
- (4)     **if** there exists  $just \in S$  s.t.  $path \cap just = \emptyset$
- (5)      $new\_just \leftarrow just$  (i.e., justifications reuse)
- (6)     **else**
- (7)      $new\_just \leftarrow \text{SINGLE\_JUST\_ALG}(C, \mathcal{O})$
- (8)     **if**  $new\_just \neq \emptyset$  (i.e.,  $C$  is satisfiable relative to  $\mathcal{O}$ )
- (9)      $S \leftarrow S \cup \{new\_just\}$
- (10)     $new\_path \leftarrow path \cup \{\alpha\}$
- (11)    **foreach**  $\beta \in new\_just$
- (12)     SEARCH-HST( $C, \mathcal{O} \setminus \{\beta\}, \beta, new\_path$ )
- (13)    **else**
- (14)     $HS \leftarrow HS \cup path$

**Proof 1.** ( $\subseteq$ ): Let  $S \in \text{ALL\_JUST\_ALG}(C, \mathcal{O})$ , then  $S$  belongs to the label of some non-leaf node  $w$  in the HST generated by the algorithm. In this case,  $\mathcal{L}(w) \in \text{ALL\_JUST}(C, \mathcal{O}')$ , for some  $\mathcal{O}' \subseteq \mathcal{O}$ . Therefore,  $S \in \text{ALL\_JUST}(C, \mathcal{O})$ .

( $\supseteq$ ): We prove by contradiction. Suppose there exists a set  $M \in \text{ALL\_JUST}(C, \mathcal{O})$ , but  $M \notin \text{ALL\_JUST\_ALG}(C, \mathcal{O})$ . In this case,  $M$  does not coincide with the label of any node in the HST. Let  $v_0$  be the root of the tree, with  $\mathcal{L}(v_0) = \{\alpha_1, \dots, \alpha_n\}$ . As a direct consequence of the completeness of Reiter's search strategy, the algorithm generates all the minimal Hitting Sets containing  $\alpha_i$  for each  $i \in \{1, \dots, n\}$ . Every minimal hitting set  $U$  is s.t.  $U \cap M \neq \emptyset$ . This follows from the fact that in order to get rid of an entailment, at least one element from each of its justifications must be removed from the ontology, and hence the hitting set must intersect each justification. This implies that  $\alpha_i \in M$  for  $1 \leq i \leq n$ , and therefore,  $\mathcal{L}(v_0) \subseteq M$ . However, since  $\mathcal{L}(v_0) \in \text{ALL\_JUST}(C, \mathcal{O})$  and  $\mathcal{L}(v_0) \subseteq M$ , then  $M \notin \text{ALL\_JUST}(C, \mathcal{O})$ , as it is a superset of an existing justification.

**Alternate Glass-box Approach to Computing All Justifications.** While the Glass-box approach (tableau tracing) mentioned in Section 2.1 is used to find a *single* justification of an unsatisfiable concept, extending it to compute all the justifications is *not* straightforward. This is because computing all justifications amounts to saturating the completion graph generated by the DL reasoner (when testing the concept satisfiability) in order to explore *all* possible clashes. This, in effect, requires us to “turn off” many of the key optimizations in the reasoner. Since the excellent performance of current OWL reasoners critically depends on these optimization techniques, having to disable them renders this technique (currently) impractical. The optimizations (such as early clash detection or back-jumping) need to be reworked (if possible) to handle the fact that finding a single clash is no longer useful (in that it stops the search). For this reason our approach to finding all justifications uses the SEARCH\_HST algorithm in combination with the SINGLE\_JUST\_ALG<sub>Black-Box</sub> or SINGLE\_JUST\_ALG<sub>Glass-Box</sub> algorithm.

### 3 Implementation and Evaluation

#### 3.1 Implementation Details

We implemented SINGLE\_JUST\_ALG<sub>Black-Box</sub> and ALL\_JUST\_ALG using the latest version of the OWL API<sup>6</sup>. This version has excellent support for manipulating axioms and has fairly direct, low level wrappers for Pellet 1.4<sup>8</sup>, and FaCT++ 1.1.7<sup>9</sup>. Such access is important for our experiments since the overhead of a remote access protocol can easily dominate processing time.<sup>7</sup>

**Implementation of SINGLE\_JUST\_ALG<sub>Black-Box</sub>.** A critical piece in the “expand” stage of the SINGLE\_JUST\_ALG<sub>Black-Box</sub> algorithm is selecting *which* axioms to copy over from  $\mathcal{O}$  into  $\mathcal{O}'$ . In our implementation, we run a loop that starts by inserting the concept definition axioms into  $\mathcal{O}'$  and slowly expands  $\mathcal{O}'$  to include axioms of structurally connected concepts, roles, and individuals (i.e., axioms which share terms in their signature). We vary the pace with which the fragment  $\mathcal{O}'$  is expanded, initially considering few axioms to keep the size of  $\mathcal{O}'$  bounded, and later allowing a large number of axioms into  $\mathcal{O}'$  (at each iteration of the loop) if the concept continues to remain satisfiable in  $\mathcal{O}'$ .

In the fast pruning stage of SINGLE\_JUST\_ALG<sub>Black-Box</sub>, we start with window size  $n$  being either one tenth of the size of the number of axioms to prune or just ten (whichever is greater). As pruning is repeated, we shrink the window size by some factor (currently, by 0.5). Pruning continues until the window size

<sup>6</sup> <http://sourceforge.net/projects/owlapi>

<sup>7</sup> We did not test with Racer Pro (<http://www.racer-systems.com/>) since, as of this writing, we only had DIG access to Racer Pro from the new OWL API and the overhead of DIG was proving unworkable. Given that Racer Pro often does better than the other reasoners and is usually in the rough ball park for normal reasoning, it is safe to extrapolate our results.

**Table 2.** Sample OWL Data used in our experiments. C=Classes, P=Properties, and I=Individuals in the ontology. Entailed are all the non-explicit subsumptions, including unsatisfiabilities, found during classification. These ontologies are available upon request.

Ontology	Expressivity	Axioms	C/P/I	Entailed	Domain
1. Generations	$ALCIF$	335	22/4/0	24	Family tree
2. DOLCE-Lite	$SHOIN(\mathcal{D})$	1417	200/299/39	3	Foundational
3. Economy	$ALCH(\mathcal{D})$	1704	338/53/481	51	Mid-level
4. MadCow	$ALCHOIN(\mathcal{D})$	105	54/17/13	32	Tutorial
5. Tambis	$SHIN$	800	395/100/0	65	Biological science
6. Sweet-JPL	$ALCHO(\mathcal{D})$	3833	1537/121/150	183	Earthscience
7. Chemical	$ALCH(\mathcal{D})$	254	48/20/0	43	Chemical elements
8. Transport	$ALCH(\mathcal{D})$	2051	444/93/183	52	Mid-level
9. MyGrid	$SHOIN$	8179	550/69/13	297	Bioinformatics services
10. University	$STOF(\mathcal{D})$	169	30/12/4	23	Training
11. AminoAcids	$ALCF$	2077	47/5/3	64	Classifies proteins
12. Sequence Ontology	$AL\mathcal{E}HI+$	1754	1248/17/9	179	The OBO (xp) sequence
13. Gene Ontology	$AL\mathcal{E}HI+$	1759	759/16/0	100	The OBO (xp) gene
14. MGED Ontology	$ALCF(\mathcal{D})$	236	236/88/0	100	Microarray experiment

is 1 (i.e. equivalent to slow pruning), or until the size of the candidate justification stays constant between each round of pruning. Thereafter, slow pruning is performed until the candidate justification is verified as minimal.

**Implementation of ALL\_JUST\_ALG.** We implemented one additional optimization in the SEARCH\_HST procedure to speed up the search. Since early path termination in the algorithm relies on the presence of previously detected hitting sets, it makes sense to find hitting sets as soon as possible during the Reiter search. Since a hitting set is basically a set that intersects all justifications, chances of getting a hitting set earlier in the search are higher if you give a higher priority to exploring axioms that are common to many justifications. For this purpose, we order axioms in a justification based on the commonality (or frequency) of the axiom across all the justifications currently found.

Our test data consists of 14 publicly available OWL ontologies that varied greatly in size, complexity, number of interesting entailments, and expressivity. See Table 2 for details. We classified each ontology to determine the unsatisfiable classes and the inferred atomic subsumptions. Since unsatisfiable classes and atomic subsumptions are the standard entailments exposed by ontology development environments, these are the natural explananda of interest. Note that 185 of the entailments are detecting unsatisfiable classes, whereas the remaining 587 represent “coherent” subsumptions. This is a bit distorted as several of the ontologies have “deliberate” bugs for tutorial purposes. But Chemical, e.g., is a production ontology, wherein (in this version) there are 37 unsatisfiable classes.

Since our sample reasoners do not handle precisely the same logic (FaCT++ has difficulty with certain datatypes), for the cross reasoner test we stripped out the problematic axioms. Since these constructs were not particularly heavily or complexly used, this seems harmless for the purpose of establishing the feasibility of these techniques.

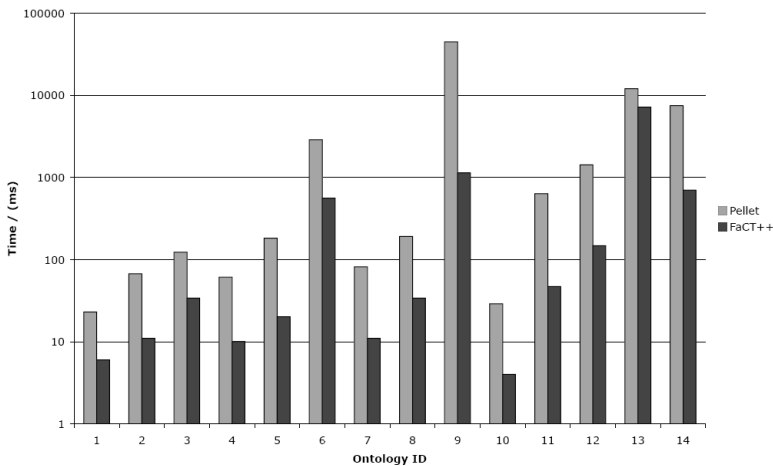
All experiments have been performed on a MacBook Pro (Intel) 2.16 GHz Intel Core Due, with 2GB RAM, and 1.5GB (max) memory allotted to Java.

### 3.2 Experimental Results

First, we recorded the base classification time for each reasoner on all the ontologies (**Fig. 3a**). Then, for each entailment, we compared the performance of generating a *single* justification (**Fig. 3b**), and then *all* justifications (**Fig. 3c**). In both cases, Pellet and FaCT++ were used in the `SINGLE_JUST_ALGBlack-Box` algorithm and Pellet with tableau tracing enabled for `SINGLE_JUST_ALGGlass-Box`.

The first striking thing to notice is that the time to compute a justification for each entailment is in the sub second range on our setup. The second thing to notice is the excellent performance of `SINGLE_JUST_ALGGlass-Box` – in many cases (ontologies 4, 5, 6 for example) the time to compute a justification was so small that it was difficult to measure (around 1 ms). It should be noted that the times in **Fig. 3b** do not include the time to classify in the first place, which is, of course, needed to find the entailments. But, it does show that there is relatively little overhead for finding a single justification even with `SINGLE_JUST_ALGBlack-Box`. We believe that these times are perfectly acceptable for generating explanations on a practical basis in ontology development environments. In such situations, ontologies have typically been classified and users generate justifications on demand as they need them.

Similarly, as seen in **Fig. 3c**, the time to compute all justifications for all entailments in these ontologies using both blackbox and glassbox implementations of `SINGLE_JUST_ALG` as input to `ALL_JUST` is impressive. We are no longer solidly in second and sub-second range across the board (ontologies 7, 9 and 14 have dramatically lengthened experiment times), but for a wide range of purposes it is quite acceptable. Again, as would be expected from the results presented in **Fig. 3b**, `ALL_JUST` with `SINGLE_JUST_ALGGlass-Box` generally beat the black box technique (in certain cases by several orders of magnitude).



**Fig. 3a.** Times for each reasoner to Classify Ontologies

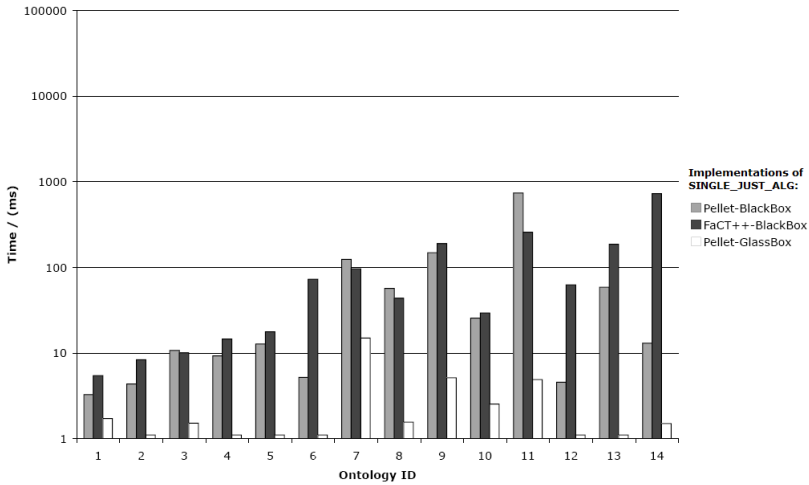


Fig. 3b. Times to Compute Single Justifications

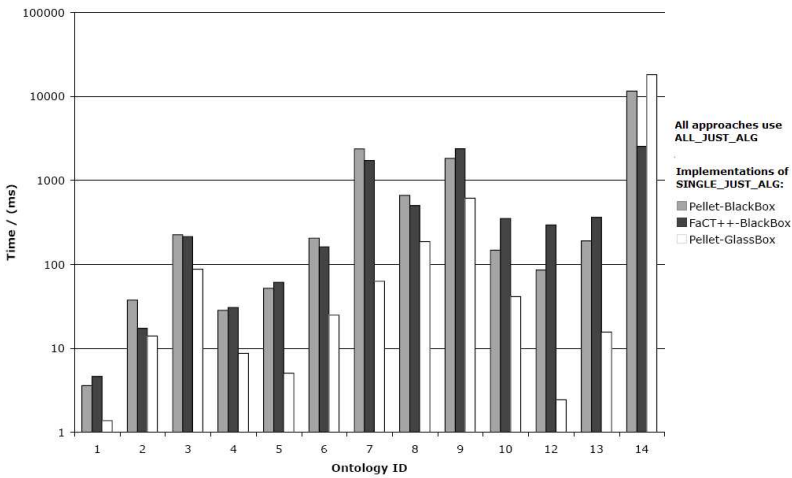


Fig. 3c. Times to Compute All Justifications

However, for ontology 14 (MGED), the reverse is true – ALL\_JUST fared worse with SINGLE\_JUST\_ALG<sub>Glass-Box</sub> than with SINGLE\_JUST\_ALG<sub>Black-Box</sub>, and, in any case, the time required is much greater than the corresponding time require to compute a single justification. The reason for this discrepancy is that in general, when computing a single justification using the blackbox technique, the reasoner only has to operate on a small fragment of the ontology. However, in the case of the glassbox approach a satisfiability check has to be performed on the whole ontology – when axioms are added or removed from the ontology this means that the satisfiability check must be rerun. While this does not pose

**Table 3.** Found justification statistics. Notice some very large justifications and some entailments with high numbers of justifications. In contrast, many ontologies had relatively small and few justifications.

Onto	No. of Justifications		Justification Size	
	Mean	Max	Mean	Max
1	1.00	1	1.79	2
2	1.00	1	1.00	1
3	1.29	2	3.61	6
4	1.09	2	2.33	6
5	1.00	1	4.12	13
6	1.12	3	2.04	7
7	9.86	26	7.99	12
8	2.33	5	5.53	9
9	1.39	8	5.39	40
10	1.65	5	5.04	10
12	1.00	1	1.50	7
13	1.04	3	1.24	7
14	1.00	1	2.00	2

a problem for many of the ontologies used in these tests, the time for Pellet to perform a satisfiability check on the MGED ontology was around 5 seconds – with multiple satisfiability checks the overall effect was noticeable.

In summary, the rather high performance of the black box case is notable. While typically beat by glass box justification generation, the total time generally stayed in the sub second range – it is arguable this performance is more than acceptable for use in ontology browsers and editors. From a pure performance perspective, it suggests that glass box techniques might not be necessary even as an optimization, at least in a wide range of cases. It should be noted that these constant factors can significantly add up if one is finding justifications for a large number of entailments (e.g., when caching a justification for each of the entailments found during classification). However, for debugging and on demand explanation it is clear that black box is sufficient for current ontologies, and has the great advantage of reasoner independence.

## 4 Related Work

There has been a lot of recent work done in capturing justifications for inconsistent ontologies or unsatisfiable concepts in relatively inexpressive description logic KBs using reasoner-dependent approaches. [10] describes a technique to find minimal sets of axioms responsible for an entailment (in this case, minimal inconsistent ABoxes) by labeling assertions, tracking labels through the tableau expansion process and using the labels of the clashes to arrive at a solution. The technique is limited to the logic  $\mathcal{ALCF}$ . Similar ideas can be seen in [11], where the motivation is debugging unsatisfiable concepts in the DICE terminology. [11] formalizes the problem including the specification of terms such as the MUPS (Minimal Unsatisfiability Preserving sub-TBox), which is the justification for the unsatisfiability entailment. [11] also describes an algorithm, restricted to  $\mathcal{ALC}$  without general TBoxes, that relies on tableau saturation in order to find all

the MUPS for an unsatisfiable  $\mathcal{ALC}$  concept and later compares this approach with a black-box implementation (in their system DION) not guaranteed to find all MUPS in [4]. On the other hand, we focus on a much more expressive logic OWL-DL (or  $\mathit{SHOIN}(\mathcal{D})$ ), provide a general definition of justification for any arbitrary OWL-DL entailment, present a sound, complete and highly optimized Glass-box and Black-box solution to finding all justifications, and demonstrate its feasibility on realistic expressive OWL-DL ontologies.

## 5 Conclusion

There is clearly room for further optimizations of all the algorithms presented in this paper, e.g., using more sophisticated analytical techniques for axiom selection in the ‘expand’ stage of the algorithm `SINGLE_JUST_ALGBlack-Box`. However, the overall message is clear: *reasoner independent techniques for finding all justifications for an OWL-DL entailment are practical*. We believe our implementation could be easily deployed in current OWL editors to offer explanation services that have been demonstrated to be useful [5] and are in high demand. In our own experience, we find having justifications completely change the way we work with ontologies for the enormously better. Indeed, we believe that no respectable OWL environment need lack for explanation support.

## References

1. Noy, N., Sintek, M., Decker, S., Crubezy, M., Fergerson, R., Musen, M.: Creating semantic web contents with Protégé-2000. IEEE Intelligent Systems (2001)
2. Kalyanpur, A., Parsia, B., Sirin, E., Cuenca-Grau, B., Hendler, J.: Swoop: A web ontology editing browser. Journal of Web Semantics 4(2) (2006)
3. Kalyanpur, A., Parsia, B., Sirin, E., Grau, B.: Repairing unsatisfiable concepts in owl ontologies. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)
4. Schlobach, S., Huang, C.R., Van-Harmelen, F.: Debugging incoherent terminologies. In: Journal of Automated Reasoning (JAL) (in press, 2007)
5. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging unsatisfiable classes in OWL ontologies. Journal of Web Semantics 3(4) (2005)
6. Horrocks, I., Patel-Schneider, P.: Reducing OWL entailment to description logic satisfiability. In: International Semantic Web Conference (2003)
7. Reiter, R.: A theory of diagnosis from first principles. Artificial Intelligence 32, 57–95 (1987)
8. Sirin, E., Parsia, B.: Pellet system description. In: Description Logics (DL) (2004)
9. Horrocks, I.: FaCT and iFaCT. In: Description Logics (1999)
10. Baader, F., Hollunder, B.: Embedding defaults into terminological knowledge representation formalisms (Technical Report RR-93-20)
11. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Proc. of IJCAI 2003 (2003)



# How Useful Are Natural Language Interfaces to the Semantic Web for Casual End-Users?

Esther Kaufmann and Abraham Bernstein

Department of Informatics, University of Zurich, Switzerland  
{kaufmann,bernstein}@ifi.uzh.ch

**Abstract.** Natural language interfaces offer end-users a familiar and convenient option for querying ontology-based knowledge bases. Several studies have shown that they can achieve high retrieval performance as well as domain independence. This paper focuses on usability and investigates if NLI are useful from an end-user's point of view. To that end, we introduce four interfaces each allowing a different query language and present a usability study benchmarking these interfaces. The results of the study reveal a clear preference for full sentences as query language and confirm that NLI are useful for querying Semantic Web data.

## 1 Introduction

The need to make the contents of the Semantic Web accessible to end-users becomes increasingly pressing as the amount of information stored in ontology-based knowledge bases steadily increases. Natural language interfaces (NLI) provide a means of querying access to casual end-users without having them to learn RDF, OWL, SPARQL, or other logic-based languages. While NLI hide the formality of ontologies and query languages from end-users by offering them a familiar and intuitive way of query formulation, the realization of NLI involves various problems:

- Due to linguistic variability and ambiguities, for which natural languages (NL) are infamous, the development of accurate NLI is a *very complex and time-consuming task* that requires extraordinary design and implementation efforts. Nevertheless, by restricting and controlling the query language such that the end-user has to follow it or engage the user in query formulation dialogues, we can eliminate linguistic variability [6,22]. Moreover, the semantics that is contained in ontologies can provide the context needed to overcome ambiguities.
- Good NLI are often domain- or application-tailored, which makes them *hardly adaptable and portable*. However, if we extract the necessary information to analyse a user's NL query from a knowledge base, NLI become domain-independent or, at least, easily adaptable to new domains [3,9].
- The retrieval performance (in terms of precision and recall) of a NLI is directly linked to the portability problem. *The more a system is tailored to a domain, the better its retrieval performance is.* The goal, however, is to

build portable NLI without sacrificing retrieval quality because end-users would not accept unreliable and inaccurate interfaces.

- Repeatedly held discussions in the NLI literature raise the *issue of the usefulness of NLIs*. Even if we design a well-performing and domain-independent NLI, it remains unclear if it is approved and adopted by end-users. In the time of Google and graphical user interfaces, where people are used to formulating their information needs with keywords and then browse through dozens of answers to find the appropriate one or to clicking through menus and graphically displayed functions, full-fledged NLIs may be redundant.

Though we have identified four problem dimensions regarding NLIs—and there may be other—we think that NLIs are a promising option for casual end-users to interact with logic-based knowledge bases. Several projects have shown that NLIs can perform well in retrieval tasks [12,20,24] and be portable as well as domain-independent [9,18,26] without being unnecessarily complex. This paper now attempts to shed some light on the problem dimension of usability and usefulness of NLIs (i.e., the last of the four issues raised above). To that end, we have implemented four interfaces, which are portable, domain-independent, and exhibit good performance, to conduct a comprehensive usability study. The four interfaces are simple in design, avoid complex configurations, and extract the knowledge needed to analyse input queries from OWL knowledge bases.

Each interface supports a different query language with a different degree of restriction and formality. We benchmarked the four systems against each other in a usability study with 48 subjects providing us with an answer to the question about the usefulness of NLIs from an end-user’s point of view. Consequently, our contribution is that *we investigate if NLIs to Semantic Web data are in fact useful for and approved by casual end-users*. Note that we refer to casual end-users as defined in [4].

The remainder of the paper is structured as follows. First, we introduce each of the four interfaces and explain their major characteristics. We, then, describe the usability study in section 3, in which the four systems are benchmarked against each other, and discuss the results, which leads to the discussion of some limitations of our approach as well as future work in section 4. The paper closes with a section on related work and conclusions.

## 2 Four Different Query Interfaces to the Semantic Web

Given our premise that NLIs are only useful for casual end-users if they are actually approved and, therefore, used by them, we conducted a usability study with four query interfaces implemented for that purpose: *Ginseng*, *NLP-Reduce*, *Querix*, and *Semantic Crystal*. Each interface requires a different query language regarding its freedom, naturalness, and formality: ranging from keywords to complete English sentences, from menu-based options to a graphically displayed query language. In the following, we describe each of the four systems beginning with the interface that has the least restrictive and most natural query language, then continuing with the systems that feature more restricted query languages, and closing with the system requiring a formal, graphical query language.

## 2.1 NLP-Reduce

NLP-Reduce is a “naïve” and completely domain-independent NLI for querying Semantic Web knowledge bases [16]. It is called *naïve* because the approach is simple and processes NL queries as bag of words only employing a reduced set of NL processing techniques, such as stemming and synonym expansion (hence its name *NLP-Reduce*). The interface allows users to enter keywords (e.g., “Chinese restaurant San Francisco”), sentence fragments (e.g., “Chinese restaurants that are in San Francisco”), or full English sentences (e.g., “Which Chinese restaurants are in San Francisco?”).

A query is first reduced by removing stopwords as well as punctuation marks and stemming the rest of the words. The system then tries to identify triple structures in the rest of the query words and match them to the synonym-enhanced triple store that is generated from an OWL knowledge base when loaded into NLP-Reduce. The identified triples are joined and translated into SPARQL statements. To execute the SPARQL query, NLP-Reduce uses Jena<sup>1</sup> and the Pellet Reasoner<sup>2</sup>. After executing the query, the results (including the URIs) and some execution statistics are displayed to the user (see Fig. 1).<sup>3</sup>

When generating the triple store from a knowledge base, NLP-Reduce also obtains synonyms from WordNet providing the users with a larger vocabulary that can be deployed when querying. This leads to better usability and eases the interface’s limitation of being dependent on the quality and choice of the vocabulary used in knowledge bases. The weakness, however, is also the interface’s major strength, as it does not need any adaption for new knowledge bases and is completely portable. From an end-user’s point of view, the major advantage of the system is that it is robust to ungrammatical and deficient input.

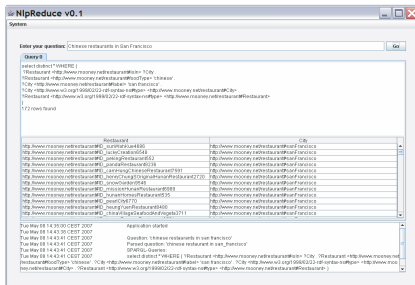


Fig. 1. The NLP-Reduce user interface

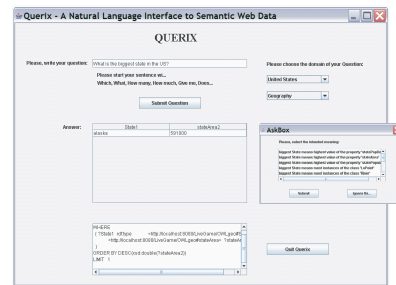


Fig. 2. The Querix user interface

## 2.2 Querix

Querix is a domain-independent NLI that requires full English questions as query language [17]. Compared to a logic-based NLI, Querix does not try to resolve

<sup>1</sup> <http://jena.sourceforge.net/>

<sup>2</sup> <http://pellet.owld.com/>

<sup>3</sup> Larger images of each interface can be found at:

<http://www.ifi.uzh.ch/ddis/research/semweb/talking-to-the-semantic-web/>

NL ambiguities, but asks the user for clarification in a dialog window if an ambiguity occurs in the input query. The user acts the role of the druid Getafix (hence the name *Querix*) who is consulted by Asterix, Obelix and the other villagers whenever anything strange occurs. A strange event within Querix is an ambiguity. The person composing a query benefits from the clarification dialog by better retrieval results.

The system uses a parser to analyse the input query. From the parser's syntax tree, a query skeleton is extracted, in which triple patterns are identified. Based on pattern matching algorithms that rely on the relationships that exist between the elements in a knowledge base, the triple patterns are then matched to the resources in the knowledge base. The matching and joining of the triples is controlled by domain and range information. From the joined triples, a SPARQL query is generated that can be executed by Jena. Using WordNet, synonyms of the words in the query and the IDs in the knowledge base are included, providing an enhanced query language vocabulary and a better matching.

If Querix encounters an ambiguity in a query, i.e., several semantically different SPARQL queries could be generated for a single NL query, the clarification dialog of the interface pops up showing the different meanings for the ambiguous element in a menu (Fig. 2). The user can now choose the intended meaning, and the interface executes the corresponding SPARQL query. Consider, for example, the query "What is the biggest state in the US?", in which the word "biggest" can refer to the properties `statePopulation`, `statePopulationDensity`, and `stateArea` of a knowledge base containing geographical information. If the user selects `statePopulation`, the answer to the query is "California;" if `stateArea` is selected, the answer Querix returns is different, namely "Alaska."

### 2.3 Ginseng

Ginseng - a *guided input natural language search engine* allows users to query OWL knowledge bases using a controlled input language akin to English [51]. Basing on a grammar, the system's incremental parser offers the possible completions of a user's entry by presenting the user with choice pop-up boxes (as shown in Fig. 3). These pop-up menus offer suggestions on how to complete a current word or what the next word might be. The possible choices get reduced as the user continues typing.

Entries that are not in the pop-up list are ungrammatical and not accepted by the system. In this way, Ginseng guides the user through the set of possible questions preventing those unacceptable by the grammar. Once a query is completed, Ginseng translates the entry to SPARQL statements, executes them against the ontology model using Jena, and displays the SPARQL query as well as the answer to the user.

When starting Ginseng, all knowledge bases in a predefined search path are loaded and the grammar compiler generates a dynamic grammar rule for every class, property, and instance. These dynamic rules enable the display of the IDs used in the ontology in the pop-up boxes. While the static grammar rules provide the basic sentence structures for questions, the dynamic rules allow that certain



ones. The interface incrementally generates textual SPARQL query statements for the current state of the graphically constructed query; the SPARQL statements are exhibited on the bottom of the right side of the user interface. In the case of datatype properties (the green properties in the menu list), a user can additionally specify whether the property’s value should be used as restriction or as output. If the output is specified, the query can be executed and the result is shown to the user in a new tab. Jena is again applied for query execution. On the dashboard in Fig. 4, we see a complete graphical representation of the query: “Give me the titles of the movies that have an actor with the family name ‘Depp’ and that were distributed in the year 2000.”

### 3 The Usability Study

The goal of the usability study was to investigate how useful NLI were to find data in Semantic Web knowledge bases in comparison with each other and a formal query language. Casual end-users should test and assess the usability of each of the four systems and, in particular, their query languages. As such, we let casual end-users perform the same retrieval tasks with each tool and find out which query language they liked best, which query language they liked least, and why. Furthermore, we examined the time they spent to perform the tasks and how successful they were with each system. To recall the range of query languages and their features provided by our interfaces, we summarize them here:

- NLP-Reduce: keywords, sentence fragments, and full sentences
- Querix: full sentences
- Ginseng: predetermined and menu-based words/sentences
- Semantic Crystal: graphically displayed, clickable, formal query language

#### 3.1 Experimental Setup

To benchmark the four interfaces in a controlled experiment, we promoted the usability study on the websites of our department and university. We, additionally, promoted the study by billboard advertisements, which we distributed in Zurich. We ended up with 48 subjects almost evenly distributed over a wide range of backgrounds and professions: bankers, biologists, computer scientists, economists, game programmers, housewives, journalists, language teachers, mechanical engineers, musicians, pedagogues, psychologists, secretaries, sociologists, veterinarians, video artists, unemployed persons to name some of them (in alphabetical order). There was a normal distribution of age ranging from 19 to 52 years with a mean of 27.6 years. With 48 users we were able to cover each possible order of the four systems not just once but twice, a fact that increases the overall statistical significance of the results (see below).

For each interface the users were asked to perform the same tasks: They had to reformulate four questions presented to them as sentence fragments into the respective query language required by the four systems and enter the questions

into the interfaces. The four questions were principally the same for each system, but we slightly changed them to make the overall experiment more interesting for the users. For example, one question was “area of Alaska?” given for NLP-Reduce and “area of Georgia?” for Querix etc. The four question templates were:

- area of Alaska?
- number of lakes in Florida?
- states that have city named Springfield?
- rivers run through state that has largest city in US?

In principle, each interface is able to answer all four queries. Each system does, however, “stumble” across one of the queries such that, for example, more than one query is needed to retrieve the correct result. For every user, we changed the order in which the interfaces were presented as well as the order of the queries for each system to prevent any learning effects from influencing the results.

After completing the questions with each interface, users were asked to answer the *SUS* questionnaire. *SUS* is a standardized usability test [8] containing ten standardized questions (e.g., “I think that the interface was easy to use.”), each answered on a 5-point Likert scale establishing a person’s impression regarding a user interface. The test covers a variety of usability aspects, such as the need for support, training, and complexity. The result of the questionnaire is a value between 1 and 100, where 1 signifies that a user found a system absolutely useless and 100 that a user found a system optimally useful.

After testing and judging all interfaces, users were explicitly asked to fill in a comparison questionnaire in which they were asked which NLI they liked best and which one they liked least; they were asked the analogous questions regarding the query languages. We also asked them about the motivations for their choices. At the end of the overall experiment, people were requested to answer a number of demographic questions such as age, gender, profession, knowledge of informatics, knowledge of linguistics, knowledge of formal query languages, and knowledge of English.

To provide an introduction to the query languages of the interfaces, users were given 1-page instructions for each system. Hence, the procedure of the experiment for each user was the following: (1) read some introductory notes on the overall experiment, (2) read instructions on the query language of the first interface, (3) reformulate, enter, and execute four queries with the first interface, (4) fill in the *SUS* questionnaire for the first interface, (5) proceed by repeating steps 2 to 4 with the second, third, and fourth interface, (6) fill in the comparison questionnaire, (6) and finally provide answers to the demographic questions. The overall experiment took about 45 to 60 minutes. Using the *Morae* Software [4] we were able to remotely record the desktop of the users as well as log and time each of their key entries and mouse clicks.

The experiment was based on the *Mooney Natural Language Learning Data* [24]. Its geography database consists of a knowledge base that contains geographical information about the US and their logical representations. We chose the

<sup>4</sup> <http://www.techsmith.com/morae.asp>



data set because it covers a domain that can easily be understood by casual users and does not demand expert knowledge. To make the knowledge base accessible to our interfaces, we translated it to OWL and designed a simple class structure as meta model.

### 3.2 Results of the Experiment

The data we collected in the experiment was analysed quantitatively as well as qualitatively. For the quantitative analysis, we used *ANOVA* and *Mixed Linear Regression Models* as available in the R-Software<sup>5</sup> and its *lme4*-package.<sup>6</sup> The first part of the results is summarized in Table 1.

**Table 1.** Average time, number of queries, and success/failure rate results

	avg time for all 4 queries	avg time per query	avg number of queries	avg success rate	avg failure rate
NLP-Reduce	2 min 39 sec	23.54 sec	7.94	69.27 %	30.73 %
Querix	4 min 11 sec	29.31 sec	7.75	77.08 %	22.92 %
Ginseng	6 min 06 sec	34.82 sec	11.06	63.54 %	36.46 %
Semantic Crystal	9 min 43 sec	89.53 sec	7.02	54.86 %	45.14 %
p-value (single factor) ANOVA with 4 levels)	1.56e-26	4.91e-40	3.92e-06	1.06e-05	2.54e-05

Most strikingly, our results are more than highly significant, which is due to the high number of users and the double coverage of every possible interface as well as query order. The first column shows that users were significantly fastest when entering the four queries with NLP-Reduce ( $p = 1.56e-26$ ). This outcome is obvious as the query language of NLP-Reduce imposes least constraints on the user and allows entering the queries with least words. Users spent most time when working with Semantic Crystal demonstrating that the intellectual burden of composing semantically and syntactically appropriate formal queries lies exclusively with the user, whereas the other three systems carry the burden to some extent. The average time that was spent per query nicely mirrors the increasing degree of formality and restrictiveness of the interfaces' query languages.

We can see in column 3 that it took users on average 7.02 queries to find answers to the four questions given in the experiment with Semantic Crystal and 11.06 query trials with Ginseng. NLP-Reduce and Querix lie in between and close to each other. The high number of query trials in Ginseng is a result of its query language's restrictiveness causing users to repeatedly reformulate and execute their queries in a kind of backtracking behavior. The log files revealed that the lowest number of query trials in Semantic Crystal emerged from users giving up and not willing to keep trying until an appropriate query was composed.

The average success and failure rates indicate how many of the four queries retrieved a satisfying answer from the users' perspective (i.e., the user thought

<sup>5</sup> <http://www.r-project.org/>

<sup>6</sup> <http://stat.ethz.ch/CRAN/>



**Table 2.** Results of SUS and comparison questionnaires

	ave SUS score	interface liked best	interface liked least	QL liked best	QL liked least
NLP-Reduce	56.72	12.50 %	25.00 %	18.75 %	25.00 %
Querix	75.73	66.67 %	2.08 %	60.42 %	4.17 %
Ginseng	55.10	6.25 %	12.50 %	16.67 %	12.50 %
Semantic Crystal	36.09	14.58 %	60.42 %	4.17 %	58.33 %
p-value (single factor ANOVA with four levels)	7.36e-17	0.297	0.297	0.0075	0.0075

that she/he had found the correct answer). Though Semantic Crystal in fact provides more precise answers than its competitors, the success rate of only 54.86% is due to inappropriate and invalid query formulations. The best success rate achieved by Querix from the users' point of view is due to Querix's answer display. For example, if a user enters a query "How many rivers run through Colorado?", the answer of Querix is: "There are 10.", while the other three interfaces show a list with the names of ten rivers and the number of results found. Some users specifically pointed out in the questionnaires that they trusted the answers of Querix more because the NL answer created the impression that the system "understood" the query.

Using the regression model, we found that the order of the four queries and the knowledge of informatics, linguistics, formal query languages, and English did not significantly affect the time. While there was no correlation between the variable *gender* and the average time spent per query either, the variable *age* influenced the average time: With every year a user's age grows, the average time to reformulate a query increases by 3.30 seconds ( $p = 0.010$ ).

Table 2 contains the results of the SUS and the comparison questionnaires. Querix achieved the highest average SUS score of 75.73 and significantly outperformed the other three interfaces ( $p = 7.36e-17$ ). The graphical query interface Semantic Crystal did not get much appreciation, which is reflected in the average SUS score of 36.09. NLP-Reduce and Ginseng achieved similar SUS scores somewhere in the middle of the other two NLIs; their scores do not significantly differ from each other (paired, one-tailed t-Test:  $p = 0.356$ ). It is no surprise that 66.67% of the users liked the Querix interface best and only 2.08% liked it least, even if this result is not significant (columns 2 and 3 in Table 2). Querix obtained almost the same feedback for its query language (QL), this time reaching statistical significance (columns 4 and 5). Even though 60.42% of the users disliked Semantic Crystal as query interface when comparing it to the other three NLIs, a surprising portion of 14.58% assessed Semantic Crystal as favorite interface. The graphically displayed knowledge base was found useful by five users. Only 12.50% liked NLP-Reduce best and 6.25% Ginseng. With respect to the query language, the results show the same ranking as the SUS scores except for the query language liked least. Here, the keywords provided by NLP-Reduce were more disliked (25.00%) than the restricted query language of Ginseng (12.50%).

We can, therefore, hypothesize that the full freedom of keyword-based query languages is less suitable for casual end-users, since it does not support the

user in the process of query formulation. The overall preference for Querix may further reflect this query language tradeoff between freedom that can produce confusion and restrictiveness that can enable guidance.

The regression analysis showed that with each second spent more with a system, the SUS score dropped by 0.06 ( $p = 1.79e-09$ ), whereas the number of queries used, the success/failure rate, and the order of the queries did not influence the SUS ratings. The order in which the interfaces were presented to the user, however, made an impact: The system that was tested last always obtained a higher SUS score ( $p = 0.0025$ ), i.e., an increase by 5.3. *Knowledge of informatics* was the only additional variable that also influenced the SUS ratings: The better the knowledge of informatics of a user was, the higher the SUS score turned out for each interface ( $p = 0.0029$ ).

When categorizing and counting the comments that users gave in the comparison questionnaire, the most often-named comments for each interface were the following:

- NLP-Reduce: +easy, +no training necessary, –QL not apparent, –QL too relaxed
- Querix: +obvious and non-constraining QL, +full sentences possible, +asks for clarification, –full sentences too restrictive
- Ginseng: +easy, +supports the user, –QL too restrictive, –few sentence structure possibilities
- Semantic Crystal: +graphical display of data, –too complex, –too laborious, –much training necessary

### 3.3 Discussion of the Most Remarkable Results

The results of the usability study with 48 users clearly show that *Querix and its query language requiring full English questions was judged to be the most useful and best-liked query interface*. This finding contradicts another usability study investigating different query languages and showing that students generally preferred keyword-based search over full-questions search [21]. The users in that study declared that they would only accept full query sentences if the retrieval results were better. In contrast, our results exhibit a highly significant preference for full-sentence queries independent of the retrieval performance.

One of the most prominent qualitative results was that several users, who rated Querix as best interface, explicitly stated that *they appreciated the “freedom of the query language.”* Nevertheless, full sentences are more restrictive than keywords and sentence fragments meaning that the query language of NLP-Reduce actually offers more freedom and less restriction than Querix. There may be two reasons for the comment: (1) With full-sentence questions, users can communicate their information need in a familiar and natural way without having to think of appropriate keywords in order to find what they are looking for. (2) People can express more semantics when they use full sentences and not just keywords. Using verbs and prepositions to link loosely listed nouns enables semantic associations, which users may experience as more freedom in query formulation.

Though Semantic Crystal was assessed as difficult and laborious to use, some users pointed out *the advantage of graphically displayed knowledge bases and queries*. Consequently, we should consider interfaces to Semantic Web data that offer a combination of graphically displayed and NL query languages. A user could then choose between different querying possibilities. Furthermore, we might have to think of adequate *NL answer generation components* [2], which seems to increase a user's trust in a system and the overall user satisfaction.

## 4 Limitations and Future Work

We are well aware that our usability study does not provide a definitive answer to the discussion of the usefulness of NLI. We deliberately omitted both a retrieval performance evaluation and a portability evaluation of our systems concentrating only on the dimension of usability. The former two evaluations have partially been done for some of the systems and their completion is part of our future work.

Concerning valid conclusions to be drawn from a usability study, we would still need a more comprehensive usability study with more users to cover more precisely distinguished degrees of query languages along a well-defined formality continuum. To prevent influences from variables that are not directly linked to the query languages, the NLIs should be the same except for the query languages. In our study the appearance of the interfaces was different.

We limited ourselves to four interfaces and four queries for several reasons. First, we wanted to cover each possible tool order; consider that a usability study with five different interfaces requires 120 users to cover each order of the interfaces. Second, we preferred to not overload the users in an exhaustive experiment risking to taint the results due to fatigue. Last, our users should not be students (like in most usability studies), but people representing a general public. Finding such users is a difficult, time-consuming, and also expensive endeavor, since we offered our users a small monetary reward for taking part.

We still believe that our usability study provides a substantial contribution to the discussion of how useful NLIs are for casual end-users. Motivated by the work of [27], we will, therefore, develop and implement a combined interface as described above and conduct further usability studies in the future.

## 5 Related Work

NLIs have been developed since the 70s, but oftentimes with moderate success [3,25], which resulted in a decreasing interest in the topic in the 90s. The necessity for robust and applicable NLIs has become more acute in recent years as the amount of information has grown steadily and immensely. A number of well-performing NLIs to databases emerged [1,10,13,14,19,20]. Considering the difficulties with full NL, it seems comprehensible that restricted NL or menu-guided interfaces have been proposed by some approaches [14,22,25]. The popularity of the Semantic Web created a number of NLIs that provide access to

ontology-based knowledge bases [9,11,12,15,18,26]. Most of the evaluations of NLI mainly focus on retrieval performance and/or the portability dimension. As our work concentrates on usability, we will only discuss three closely related projects that conducted a usability study: ORAKEL, Squirrel, and CHESt.

ORAKEL by Cimiano and colleagues [9] is a portable NLI to knowledge bases that is ontology-based in two ways. First, it uses an ontology in the inference process to answer users' queries. Second, the system employs an ontology in the process of adapting the system to a domain and a specific knowledge base. This adaptation is performed by domain experts and has been evaluated in a user study. It was shown that people without any NLI expertise could adapt ORAKEL by generating a domain-specific lexicon in an iterative process. The controlled study involved 26 users from both academic and industrial institutions. Results were reported in terms of recall and precision showing that the iterative methodology to lexicon customization was indeed successful. A second experiment was performed to determine the linguistic coverage of 454 questions asked by end-users. They report an excellent coverage of 93%, but did not investigate the usefulness from the end-users' point of view.

The Squirrel system presented by Duke et al. [11] is a search and browse interface to semantically annotated data. It allows combined search facilities consisting of keyword-based and semantic search in order to balance between the convenience for end-users and the power of semantic search. Users can enter free text terms, see immediate results, and follow with a refinement of their query by selecting from a set of matching entities that are associated with the result set and returned by the system on the basis of an ontology. Squirrel has been evaluated in three steps: (1) in a heuristic evaluation, in which usability experts judged the interface according to a list of usability heuristics, (2) in a walk-through evaluation, where users were asked to complete a number of tasks, while their actions were recorded, and (3) in a set of field tests giving users information seeking tasks and collecting feedback. Promising results obtained from 20 users are reported: Squirrel achieved an average perceived information quality of 4.47 on a 7-point scale. It was rated positively regarding its properties but skeptically in terms of performance and speed. Regrettably, the authors provide neither a detailed description of the evaluations nor explicit results.

The core of the work by Reichert and her colleagues [21] lies in a usability study, making it most closely related to our work. They investigate how students assess the possibility of querying a multimedia knowledge base by entering full questions instead of just keywords. For this purpose, two versions of the e-learning question-answering tool CHESt were implemented. The first version offers a keyword-based search; the second version allows a semantic search with full sentences as query input. They conducted three task-oriented experiment sessions with 18, 18, and 14 students and benchmarked the two versions of CHESt. The outcome of the three sessions is that the students generally preferred the keyword-based search to the full questions search (76% on average). This was found to be independent of the appropriateness of the results. The students reported that they would use the option of complete questions if this yielded in better results. Nonetheless, the authors conclude that the intellectual

task of thinking and formulating full-sentence queries must not necessarily be considered as a burden compared to entering loose keywords. We can confirm this conclusion from our usability study, which presents a wider choice of query languages, and draw even more detailed conclusions.

The approaches in the field of NLIs nicely show that such interfaces can successfully tackle the performance and transportability dimension. As such, they complement our findings, which focuses on the usability dimension. However, more work is needed regarding NLIs to Semantic Web data and further comprehensive usability studies to investigate the end-users's perspective.

## 6 Conclusions

This paper attempted to answer the question *if NLIs are actually useful for casual end-users*. While most studies concerning NLIs to structured data aim at high-quality retrieval performance and transportability, we focused on the usability dimension. Our usability study with 48 users and four interfaces featuring four different query languages showed that the full-sentence query option was significantly preferred to keywords, a menu-guided, and a graphical query language. NLIs offering an adequate query language can, therefore, be considered to be indeed useful for casual end-users. We believe that our study generally shows the potential of NLIs for end-user access to the Semantic Web, providing a chance to offer the Semantic Web's capabilities to the general public.

**Acknowledgement.** This work was partially supported by the Swiss National Science Foundation (200021-100149/1).

## References

1. Andreasen, T.: An approach to knowledge-based query evaluation. *Fuzzy Sets and Systems* 140(1), 75–91 (2003)
2. Androutsopoulos, I., Kallonis, S., Karkaletsis, V.: Exploiting owl ontologies in the multilingual generation of object descriptions. In: 10th Europ. Workshop on Natural Language Generation, pp. 150–155, Aberdeen, UK (2005)
3. Androutsopoulos, I., Ritchie, G.D., Thanisch, P.: Natural language interfaces to databases - an introduction. *Natural Language Engineering* 1(1), 29–81 (1995)
4. Battle, L.: Preliminary inventory of users and tasks for the semantic web. In: 3rd Intl. Semantic Web User Interaction Workshop, Athens, GA (2006)
5. Bernstein, A., Kaufmann, E.: Gino - a guided input natural language ontology editor. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 144–157. Springer, Heidelberg (2006)
6. Bernstein, A., Kaufmann, E., Göhring, A., Kiefer, C.: Querying ontologies: A controlled english interface for end-users. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 112–126. Springer, Heidelberg (2005)
7. Bernstein, A., Kaufmann, E., Kaiser, C.: Querying the semantic web with ginseng: A guided input natural language search engine. In: 15th Workshop on Information Technologies and Systems, Las Vegas, NV, pp. 112–126 (2005)

8. Brooke, J.: Sus - a “quick and dirty” usability scale. In: Jordan, P.W., et al. (eds.) *Usability Evaluation in Industry*, Taylor & Francis, London (1996)
9. Cimiano, P., Haase, P., Heizmann, J., Mantel, M.: Orakel: A portable natural language interface to knowledge bases. Technical report, Institute AIFB, University of Karlsruhe (2007)
10. Dittenbach, M., Merkl, D., Berger, H.: A natural language query interface for tourism information. In: 10th Intl. Conf. on Information Technologies in Tourism, Helsinki, Finland, pp. 152–162 (2003)
11. Duke, A., Glover, T., Davies, J.: Squirrel: An advanced semantic search and browse facility. In: 4th ESWC, Innsbruck, A, pp. 341–355 (2007)
12. Frank, A., Krieger, H.-U., Xu, F., Uszkoreit, H., Crysmann, B., Jörg, B., Schäfer, U.: Question answering from structured knowledge sources. *Journal of Applied Logic*, Special Issue on Questions and Answers: Theoretical and Applied Perspectives 5(1), 20–48 (2007)
13. Guarino, N., Masolo, C., Vetere, G.: Ontoseek: Content-based access to the web. *IEEE Intelligent Systems* 14(3), 70–80 (1999)
14. Hallett, C., Power, R., Scott, D.: Intuitive querying of e-health data repositories. In: UK E-Science All-hands Meeting, Nottingham, UK (2005)
15. Katz, B., Lin, J., Quan, D.: Natural language annotations for the semantic web. In: Intl. Conf. on Ontologies, Databases, and Applications of Semantics, Irvine, CA (2002)
16. Kaufmann, E., Bernstein, A., Fischer, L.: Nlp-reduce: A “naïve” but domain-independent natural language interface for querying ontologies. In: 4th ESWC, Innsbruck, A (2007)
17. Kaufmann, E., Bernstein, A., Zumstein, R.: Querix: A natural language interface to query ontologies based on clarification dialogs. In: 5th ISWC, Athens, GA, pp. 980–981 (2006)
18. Lopez, V., Motta, E., Uren, V.: Poweraqua: Fishing the semantic web. In: 3rd ESWC, Budva, Montenegro, pp. 393–410 (2006)
19. Minock, M.: A phrasal approach to natural language interfaces over databases. Technical Report UMINF-05.09, University of Umea (2005)
20. Popescu, A.-M., Etzioni, O., Kautz, H.: Towards a theory of natural language interfaces to databases. In: 8th Intl. Conf. on Intelligent User Interfaces, Miami, FL, pp. 149–157 (2003)
21. Reichert, M., Linckels, S., Meinel, C., Engel, T.: Student’s perception of a semantic search engine. In: IADIS Cognition and Exploratory Learning in Digital Age, Porto, Portugal, pp. 139–147 (2005)
22. Schwitter, R., Tilbrook, M.: Let’s talk in description logic via controlled natural language. In: *Logic and Eng. of Natural Language Semantics*, Tokyo, Japan (2006)
23. Spoerri, A.: Infocrystal: A visual tool for information retrieval management. In: 2nd Intl. Conf. on Information and Knowledge Management, Washington, D.C. (1993)
24. Tang, L.R., Mooney, R.J.: Using multiple clause constructors in inductive logic programming for semantic parsing. In: 12th Europ. Conf. on Machine Learning, Freiburg, Germany, pp. 466–477 (2001)
25. Thompson, C.W., Pazandak, P., Tennant, H.R.: Talk to your semantic web. *IEEE Internet Computing* 9(6), 75–78 (2005)
26. Wang, C., Xiong, M., Zhou, Q., Yu, Y.: Panto - a portable natural language interface to ontologies. In: 4th ESWC, Innsbruck, A, pp. 473–487 (2007)
27. Wang, T.D., Parsia, B.: Cropcircles: Topology sensitive visualization of owl class hierarchies. In: 5th ISWC, Athens, GA, pp. 695–708 (2006)

# The Fundamentals of iSPARQL: A Virtual Triple Approach for Similarity-Based Semantic Web Tasks

Christoph Kiefer, Abraham Bernstein, and Markus Stocker

Department of Informatics, University of Zurich, Switzerland  
{kief,bernstein,stocker}@ifi.uzh.ch

**Abstract.** This research explores three SPARQL-based techniques to solve Semantic Web tasks that often require similarity measures, such as semantic data integration, ontology mapping, and Semantic Web service matchmaking. Our aim is to see how far it is possible to integrate customized similarity functions (CSF) into SPARQL to achieve good results for these tasks. Our first approach exploits virtual triples calling property functions to establish virtual relations among resources under comparison; the second approach uses extension functions to filter out resources that do not meet the requested similarity criteria; finally, our third technique applies new solution modifiers to post-process a SPARQL solution sequence. The semantics of the three approaches are formally elaborated and discussed. We close the paper with a demonstration of the usefulness of our iSPARQL framework in the context of a data integration and an ontology mapping experiment.

## 1 Introduction

Semantic Web tasks such as semantic data integration [18], ontology mapping [9], Semantic Web service matchmaking [14], and similarity-based retrieval [13] depend on some *notion of similarity*. Therefore, researchers still try to find sound *customized similarity functions* (CSF) to achieve good results for these tasks. Finding good similarity functions is, however, data-, context-, and sometimes even user-dependent, and needs to be reconsidered every time new data or a new task is inspected. Nonetheless, good CSFs are crucial for the success of the above-mentioned Semantic Web tasks.

In the past, we made the following two observations: First, it is often not enough to use a single similarity measure to achieve good results. In this case, a (possibly weighted) combination of atomic measures needs to be engineered (or even learned), which turns out to be best for the specific task and data [12]. In this paper, we, therefore, formally define the concept of a *similarity strategy* which can utilize a multitude of individual similarity measures and aggregation schemes to compare Semantic Web resources.

Second, in recent years, the RDF query language SPARQL has gained increasing popularity. It offers well-known constructs from database technology, defines



standardized results formats, and is a protocol for distributed querying as well. The current W3C candidate recommendation of SPARQL [22] does, however, not support CSFs to analyze the data during query processing. This paper focuses on how to overcome this limitation by introducing iSPARQL – an extension of SPARQL that supports CSFs in order to query RDF graphs for similarities. The “i” stands for *imprecise* indicating that two or more resources are compared by using similarity measures. The proposed iSPARQL framework should be easy to use, easily extendable with new user-defined, task-specific similarity functions, as well as have a high degree of flexibility in terms of customization to the actual Semantic Web task.

In the following sections, we propose three approaches for integrating CSFs into SPARQL. Our first approach uses *virtual triples* calling property functions in the subject-predicate-object-style. These triples are not matched against the underlying ontology graph, but instead, employ CSFs to establish virtual relations (of similarity) between the resources under comparison. In that context, we will define the concept of *similarity joins* between RDF graph patterns which apply similarity measures to combine data. The second approach is based on pure SPARQL *extension functions* to filter out resources which are not sufficiently similar to each other. Our third method introduces new *solution modifiers* to the current SPARQL grammar to post-process and transform the solution of the graph pattern matching part into a new one by means of similarity measures. We evaluate our prototype iSPARQL system using two sets of experiments: (1) a *data integration experiment* and (2) an *ontology mapping experiment*.

Our contribution is, hence, twofold: first, *we present and compare three novel approaches for integrating similarity querying and SPARQL* resulting in a proposition for iSPARQL; second, *we show the importance of this proposition using two real-world Semantic Web tasks*.

The paper is structured as follows: next, we briefly introduce the most important related work, before we explain the details of our formal setup and the similarity measures used in the paper (Section 3). Section 4 explains our three approaches to add the notion of similarity into SPARQL, which are evaluated in Section 5. We close the paper with a brief discussion, limitations, some insights into future work, and our conclusions.

## 2 Related Work

**Similarity.** The concept of similarity is a heavily researched subject in Computer Science, Psychology, Artificial Intelligence, and Linguistics literature. Typically, those studies focus on the similarity between vectors [1], strings [7], trees [24], or objects [26]. In our case, we are interested in the similarity between (complex) Semantic Web resources of ontologies (*i.e.*, classes and individuals). Apart from this, several other studies reason generally about the theory of similarity (dissimilarity) [21,20]. Most notably, Orozco and Belanche [20] define the concept of *similarity aggregation operators* – the counterpart to our *similarity aggregation schemes* – and its properties on a formal level.



**SPARQL.** Cyganiak [8] describes how to transform (a subset of) SPARQL into relational algebra, which is, as argued by the author, the language of choice when analyzing queries in terms of query planning and optimization. A translation into SQL is explained and the semantics of the relational algebra operators is defined. This idea is further refined by Pérez *et al.* [21] who conduct an extensive analysis of the semantics and complexity of SPARQL, focusing on the algebraic operators JOIN, UNION, OPTIONAL, and FILTER. The semantics and complexity of these operators are studied in great detail and insights into query optimization possibilities are presented. The foundations of SPARQL extension and property functions are, however, not addressed in their paper.

Siberski *et al.* [25] propose SPARQL extensions to allow the user to query RDF graphs with *user-defined preference criteria*. To achieve this goal, a new solution modifier is added to the official SPARQL grammar [22]. The main difference to our approach is the way the results are ranked: iSPARQL uses a multitude of similarity strategies (focusing on different dimensions of resources) to determine an overall degree of similarity between resources. To that end, it employs aggregation schemes to give more or less preference to one of the dimensions considered. Finally, the ranking is produced by ordering the results according to the overall similarity score. Thus, while preference criteria require an explicit formulation of one's preferences, which is oftentimes very difficult before seeing the result set, iSPARQL allows their implicit determination via similarity measures – the approach preferred in Information Retrieval (IR).

In earlier work [4], we already introduced iRDQL – our extension of RDQL (SPARQL predecessor) with similarity joins. A limitation of iRDQL is that it allows the user to define only one similarity measure per query.

**Similarity Joins (Data Integration).** Our approach is partly inspired by studies from database research. Thus, we succinctly summarize the relevant publications. To perform data integration, Cohen [6] presents WHIRL and the notion of *similarity joins* by which data is joined on *similarity* rather than on *equality*. In WHIRL, the TF-IDF weighting scheme from IR [11] is applied together with the cosine similarity measure to determine the affinity of simple text in relations. Similar approaches are proposed by Gravano *et al.* [11] employing *text joins* to correlate information from different web sources.

In addition, a series of studies focuses on Semantic Web data integration: Noy [18] summarizes the necessity and requirements of ontology integration on the Semantic Web, pointing out the need for (semi-) automatic similarity detection between ontologies. This research resulted in the PROMPT Suite [19] to compare and align ontologies (among others). Furthermore, in two recent studies, Lam *et al.* [15] and Meštrović and Čubrillo [17] propose their very specific approaches to Semantic Web data integration using the Flora-2 system and an RDF-enabled Oracle 10g database respectively.

**Ontology Mapping.** Euzenat *et al.* [10] propose an ontology alignment API and a Java tool call OLA that implements a universal measure for comparing resources in ontologies. Contributing to the same task, Ehrig *et al.* [9] present a

layered approach to ontology mapping that focuses on different (modeling) aspects of ontologies. In their work, they define *similarity amalgamation functions* that are revived and studied in this paper, and denoted by *similarity aggregation schemes*.

### 3 Foundations: SPARQL and Similarity Measures

Before investigating different ways for adding the notion of similarity into SPARQL, we must lay out the foundations for this endeavor. Specifically, we need to discuss SPARQL and notions of similarity measures.

#### 3.1 SPARQL and Similarity

In the heart of SPARQL, an RDF graph matching algorithm exhaustively tries to find mappings between query variables and graph nodes. According to [22], a solution mapping  $\mu(?v \mapsto t)$  is defined as a mapping of a query variable  $?v \in V$  to an RDF term  $t$ , where  $V$  is the infinite set of query variables and  $t$  a member of the set union of IRIs, RDF literals, and blank nodes. A multiset (or bag) of possible solution mappings is denoted by  $\Omega$ . We define a similarity measure to be used in our iSPARQL approach (see Section 4) as follows:

**Definition 1.** *A similarity measure  $sm$  is a function  $sm : \mu_1 \times \mu_2 \mapsto \mathbb{R}$  that associates the similarity of two input solution mappings  $\mu_1$  and  $\mu_2$  to a similarity score  $sc \in \mathbb{R}$  in the range  $[0, 1]$ .*

In this context, a similarity score of 0 stands for complete inequality and 1 for equality of the input solution mappings  $\mu_1$  and  $\mu_2$ .

**Definition 2.** *Solution mappings  $\mu_1(?v_1 \mapsto t_1)$  and  $\mu_2(?v_2 \mapsto t_2)$  are similar if the values  $t_1$  and  $t_2$  bound to their query variables are similar.*

In an iSPARQL query it should be possible to apply many different similarity measures, which would result in a set  $SC$  of individual similarity scores. When the similarity between (complex) Semantic Web resources should be calculated, it is desirable to combine different measures (e.g., by using weights to give more or less importance to an individual similarity computation). For that purpose, we introduce our concept of a similarity aggregation scheme:

**Definition 3.** *A similarity aggregation scheme  $as$  is a function  $as : SC \mapsto \mathbb{R}$  that defines how previously calculated similarity scores  $sc_i \in SC$ , where  $i \in \mathbb{N}$ , are combined. The result is again a similarity score  $sc_o \in \mathbb{R}$ .*

An aggregation scheme may combine the similarities using any type of mathematical function. It is left to the user who executes an iSPARQL query to consider the semantics of such functions. We can now define the concept of an iSPARQL similarity strategy:

**Definition 4.** *A similarity strategy  $st$  is a function  $st : SM \times AS \mapsto \mathbb{R}$  that takes a set of similarity measures  $SM$  and aggregations schemes  $AS$  and returns*

a single similarity value  $sc$  expressing the combined, overall similarity between Semantic Web resources.

Of course, an iSPARQL query can employ any number of different similarity strategies, which then, altogether, define an overall strategy to compare resources (refer to the extended query in Appendix A).

### 3.2 Similarity Measures

In all our experiments, we found it, a priori, hard to say which measure (or strategy) was best to be used in a query. Furthermore, the choice of the best performing similarity measure is often context- and data-dependent [3]. We, therefore, implemented a set of similarity measures that performed well in different application domains in a generic Java library called *SimPack* [1]. For the sake of completeness, we succinctly review the similarity measures we use in this paper: Levenshtein string similarity (*isparql:lev*), Jaccard similarity (*isparql:jac*), and TF-IDF (*isparql:tfidf*). Generally speaking, all measures of *SimPack* can be used in our iSPARQL framework.

The Levenshtein string similarity determines the relatedness of two strings in terms of the number of insert, remove, and replacement operations to transform one string  $str_1$  into another string  $str_2$  [16]. This edit distance is defined as  $xform(str_1, str_2)$ . As a normalization factor, the worst case transformation cost  $xform_{wc}(str_1, str_2)$  is calculated replacing all parts of  $str_1$  with parts of  $str_2$ , then deleting the remaining parts of  $str_1$ , and inserting additional parts of  $str_2$ . The final similarity between  $str_1$  and  $str_2$  is calculated by  $sim_{lev}(str_1, str_2) = 1 - \frac{xform(str_1, str_2)}{xform_{wc}(str_1, str_2)}$  turning the normalized edit distance into a similarity score.

Co-occurrence measures are widely used for calculating similarity scores in text mining and IR [23]. The Jaccard measure calculates the similarity of two sets  $A$  and  $B$  as the ratio of the number of shared elements to the number of unified elements. For Jaccard, this is expressed by  $sim_{jac}(A, B) = \frac{|A \cap B|}{|A \cup B|}$ .

The TF-IDF measure aims at computing the degree of overlap of text documents as the cosine of the angle between the weighted vectors representing the documents [1]. TF-IDF gives each term  $t_i$  in a document  $d$  a weight which can be computed as  $w_{t_i, d} = tf_{t_i, d} \times idf_{t_i} = tf_{t_i} \times \log(\frac{N}{d_{t_i}})$ , where  $tf_{t_i, d}$  is the number of occurrences of  $t_i$  in  $d$ ,  $N$  the total number of text documents, and  $d_{t_i}$  the number of documents where  $t_i$  appears. A high TF-IDF weight is reached by a high term frequency and a low inverse document frequency. Hence, common terms in the document collection are penalized. The similarity between two text documents is the cosine of the angle between their document vectors  $v_{d_1}$  and  $v_{d_2}$ :  $sim_{tfidf}(v_{d_1}, v_{d_2}) = \frac{v_{d_1} \cdot v_{d_2}}{\|v_{d_1}\|_2 \cdot \|v_{d_2}\|_2}$ , where  $\|v\|_2$  is the  $L^2$ -vector norm.

## 4 Our Approach: Imprecise SPARQL

In this section, we present three approaches to the task of extending SPARQL with similarity operators: (1) the *virtual triple approach*: calling customized

<sup>1</sup> <http://www.ifi.uzh.ch/ddis/simpack.html>

---

```

PREFIX isparql: <java:isparql.>
PREFIX swrc: <http://swrc.ontoware.org/ontology#>
PREFIX opus: <http://lsdis.cs.uga.edu/projects/semdis/opus#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

```

---

**Listing 4.1.** Query prefixes used in this paper

similarity functions (CSFs) that take some inputs and return an output to the query engine; (2) the *extension function approach*: using existing SPARQL filtering functionalities in combination with CSFs; and (3), the *solution modifier approach*: adding new solution modifiers to the official W3C SPARQL grammar to perform similarity computations.

The example query used in this section intends to find similar publications in two different datasets. It compares publication titles and book titles (*i.e.*, journal names) utilizing two similarity measures (*isparql:jac* and *isparql:lev*) and the *score*-aggregation scheme. This scheme sums up weighted similarity scores as follows:  $\sum_{i=1}^n w_i sc_i$ , where  $i \in \mathbb{N}$  and  $\sum_{i=1}^n w_i = 1$ . The datasets are denoted by *opus* and *swrc* (see Section 5). The queries in the remainder of the paper use the prefixes shown in Listing 4.1.

## 4.1 Virtual Triple Approach

Our first proposed approach to solve the problem of adding similarity operators to SPARQL makes use of so called *magic properties*<sup>2</sup>. The concept behind this is simple: whenever the predicate of a triple pattern is prefixed with a special name, a call to a customized, external similarity function (CSF) is made and arguments are passed to the function (by the object of the triple pattern). Finally, a value is computed and returned to the subject variable of the triple pattern. We call this approach the *virtual triple approach* as such triple patterns are not matched against the underlying ontology graph, but against the only virtually existing similarity between the resources referred to in the triple. We define a virtual triple *vt* as a triple employing a particular kind of property function as follows:

**Definition 5.** A virtual triple pattern *vt* is a triple of the form  $[?v \text{ } \textit{apf} : \textit{funct} \textit{ } \textit{ArgList}]$  where *funct* is a property function and *ArgList* a list of solution mapping arguments  $\mu(?x_1 \mapsto t_1), \mu(?x_2 \mapsto t_2), \dots, \mu(?x_n \mapsto t_n)$  of *funct*.

A virtual triple establishes a relation between two Semantic Web resources, which is neither modeled in nor inferred using the typical RDFS or OWL semantics. The relation is entirely determined by the property function-defined logic and exists only during query execution (unless materialized in advance, see end of section). Virtual triples can conceptually be thought of as virtual relations such as  $[?pub1 \text{ isSimilarTo } ?pub2]$  that would associate the two publication

---

<sup>2</sup> <http://jena.sourceforge.net/ARQ/extension.html#propertyFunctions>

---

```

1 SELECT ?publication1 ?publication2 ?similarity
2 WHERE
3 { ?publication1 rdfs:label ?title1 .
4   ?publication1 opus:book_title ?booktitle1 .
5   ?publication2 swrc:title ?title2 .
6   ?publication2 swrc:booktitle ?booktitle2 .
7
8   IMPRECISE {
9     ?sim1 isparql:jac (?title1 ?title2) .
10    FILTER (?sim1 >= 0.5) .
11    ?sim2 isparql:lev (?booktitle1 ?booktitle2) .
12    FILTER (?sim2 >= 0.5) .
13    ?similarity isparql:score (?sim1 ?sim2 0.6 0.4) .
14    FILTER (?similarity >= 0.5) }
15 } ORDER BY DESC(?similarity)

```

---

**Listing 4.2.** iSPARQL example query for the virtual triple approach

**Table 4.1.** Extended SPARQL grammar for the virtual triple approach

[22]	$GraphPatternNotTriples ::= OptionalGraphPattern \mid GroupOrUnionGraphPattern \mid GraphGraphPattern \mid SimilarityBlockPattern$
[22.1]	$SimilarityBlockPattern ::= 'IMPRECISE' \{ ' ( ( VARI FunctionCall )+ Filter? )+ ' \}$

resources with a similarity score. Here, the predicate `isSimilarTo` does not have to exist in the RDF dataset  $D$ ; it is, at this point, still considered as *imaginary*.

*Syntax and Grammar.* The query in Listing 4.2 shows the example query using virtual triples. The extended SPARQL grammar is shown in Table 4.1. To implement our virtual triple approach, we added a *SimilarityBlockPattern* symbol to the official SPARQL grammar rule of *GraphPatternNotTriples* [22]. The structure of *SimilarityBlockPattern* resembles the one of *OptionalGraphPattern* but has complete different semantics: instead of matching patterns in the RDF graph, the triples in an *SimilarityBlockPattern* act as virtual triple patterns, which are interpreted by the query processor. A *SimilarityBlockPattern* expands to rule [22.1] that adds the new keyword `IMPRECISE` to the grammar, which is followed by a number of virtual triples and optional `FILTER`-statements.

*Semantics.* The evaluation of the first four triple patterns (lines 3–6) results in four individual sets of solution mappings  $\Omega_1, \dots, \Omega_4$  that are successively tried to be joined. This operation is performed by extending the sets of mappings with compatible mappings from other sets (until all sets are processes and no more compatible mappings are found). The semantics of this join operation of basic graph pattern matching is described in details by Pérez *et al.* in [21].

The semantics of a *SimilarityBlockPattern* is basically that of a *similarity join* and an (optional) *filter operation*: (1) it computes the query solutions for the similarity scores; (2) it eliminates those solutions which do not meet the filter constraints; and (3), it joins the remaining similarity scores to the solution mappings found by normal graph pattern matching.

We define the sets of *virtual solution mappings*  $\mu_v$  as  $\Omega_{VT}$  and the sets of solution mappings found by normal graph matching as  $\Omega_{GPM}$ . Furthermore, on

the basis of the definition of basic graph patterns in [22], we define *virtual graph patterns*  $VP$  as sets of virtual triple patterns  $vt$ . Based on this definitions, we define the similarity join between basic and virtual graph patterns as follows:

**Definition 6.** *A similarity join between basic and virtual graph pattern expressions  $P$  and  $VP$  extends the sets  $\Omega_{GPM}$  returned from normal graph pattern matching with the sets of virtual solution mappings  $\Omega_{VT}$  determined from virtual triple pattern matching.*

$$\Omega_{GPM} \bowtie_s \Omega_{VT} = \{ \mu_1 \cup \mu_2 \mid \mu_1 \in \Omega_{GPM}, \mu_2 \in \Omega_{VT} \text{ AND } \mu_2 \models R^- \}$$

Definition 6 accounts for *build-in* conditions  $R^-$ , which virtual solution mappings optionally must satisfy (denoted by  $\mu_2 \models R^-$ ). In our case, possible conditions are constructed by using constants, virtual solution mappings, and the operators  $<$ ,  $\leq$ ,  $\geq$ ,  $>$ , and  $=$ . If no conditions are defined,  $\mu_2 \models R^-$  always evaluates to true. In accordance to Pérez *et al.* [21], we can now define the semantics of virtual graph pattern expressions using an evaluation function  $[[\cdot]]$  over a dataset  $D$ .

$$[[vt]] = \{ \mu_v(?v \mapsto sc) \mid sc = \text{apf:funct}(\mu(?x_1 \mapsto t_1), \dots, \mu(?x_n \mapsto t_n)) \}$$

$$[[ (P \text{ SIMJOIN } VP) ] ]_D = [[P]]_D \bowtie_s [[VP]] \quad (1)$$

The first part of Equation 1 takes a virtual triple pattern expression and returns a set containing a single virtual solution mapping  $\mu_v$ . In other words, a new solution mapping is generated that is not found by ordinary graph pattern matching and that assigns a similarity score to a query variable. Note that for a similarity measure, we limited `funct` to two input arguments, whereas more than two arguments can be passed to an aggregation scheme.

*Pros and Cons.* The following list summarizes the pros and cons of this approach.

- + Multiple similarity measures can be employed to compose sophisticated user- and data-specific similarity strategies.
- + Similarity scores are assigned to variables, thus, can be reused in the query for aggregation and ranking or can be returned for further processing.
- + Aggregation schemes can be applied to calculate overall similarity scores.
- The SPARQL-grammar needs to be extended to account for the IMPRECISE-statements. This requires an adaptation of the query engines.
- Queries using property functions depend on a query engine extension (currently only implemented in Jena ARQ<sup>3</sup>), hence, have limited interoperability.

Note that while we regard the need for extending SPARQL-engines with the iSPARQL grammar and property functions as the major downside of this approach, we think the benefits – mainly the possibility to establish virtual relations and to reuse similarity scores in the query for aggregation and ranking – are sufficient to justify such extensions.

<sup>3</sup> <http://jena.sourceforge.net/ARQ/>

---

```

1 CONSTRUCT
2 { ?publication1 simont:isSimilarTo ?publication2 . # similarity ontology
3   ?publication1 simont:sc ?sim
4 }
5 WHERE
6 { ?publication1 rdfs:label ?title1 .
7   ?publication2 swrc:title ?title2 .
8   ?sim isparql:lev (?title1 ?title2) .
9   FILTER ( ?sim >= 0.5 )
10 }

```

---

**Listing 4.3.** CONSTRUCT-query to materialize similarity computations

---

```

1 SELECT ?publication1 ?publication2
2 WHERE
3 { ?publication1 rdfs:label ?title1 .
4   ?publication2 swrc:title ?title2 .
5   ?publication1 opus:book_title ?booktitle1 .
6   ?publication2 swrc:booktitle ?booktitle2 .
7
8   FILTER ( isparql:jac(?title1, ?title2) >= 0.5
9           && isparql:lev(?booktitle1, ?booktitle2) >= 0.5
10          && ( 0.6 * isparql:jac(?title1, ?title2)
11              + 0.4 * isparql:lev(?booktitle1, ?booktitle2) >= 0.5 ) )
12 }

```

---

**Listing 4.4.** iSPARQL example query for the extension function approach

Also, virtual triples can be materialized by using, for example, the SPARQL CONSTRUCT-query form in combination with a similarity ontology as sketched in Listing 4.3. This query constructs new triples as defined in the graph template of the query’s construct-clause. It would require a similarity ontology (*simont*) that would specify all the classes and predicates necessary to model similarity calculations. Afterwards, the generated graph can be queried for the similarity scores with a common SELECT-query.

## 4.2 Extension Function Approach

We present a second approach that does not require to extend the SPARQL grammar. It is solely based on pre-defined SPARQL filter functions to carry out the desired similarity computations as part of the filtering process. Listing 4.4 shows the version of the example query that uses only extension functions to calculate and weight individual similarity scores.

*Semantics.* The semantics is that of a SPARQL FILTER-expression with its filter condition  $R$  as defined in [21] but extended to account for extension functions. Thus, the set of operators to build the filter condition  $R$  has to be extended with the similarity measures and the symbols  $<$ ,  $\leq$ ,  $\geq$ , and  $>$ . We denote such a condition with similarity measures as  $R^+$ .

$$[[ (P \text{ FILTER } R^+) ] ]_D = \{ \mu \in [[P]]_D \mid \mu \models R^+ \} \quad (2)$$



---

```

1 SELECT ?publication1 ?publication2
2 WHERE
3   { ?publication1 rdfs:label ?title1 .
4     ?publication1 opus:book_title ?booktitle1 .
5     ?publication2 swrc:title ?title2 .
6     ?publication2 swrc:booktitle ?booktitle2
7   }
8 IMPRECISE ( ?title1 ?title2 ) SIMMEASURE ( isparql:jac ) THOLD 0.5
9 IMPRECISE ( ?booktitle1 ?booktitle2 ) SIMMEASURE ( isparql:lev ) THOLD 0.5
10 AGGREGATOR ( isparql:score(0.6 0.4) ) THOLD 0.5

```

---

**Listing 4.5.** iSPARQL example query for the solution modifier approach

*Pros and Cons.* Assuming that there is a reference to the implementing class of the similarity measure, this approach can be used immediately with the current SPARQL specification. The similarity scores can, however, not be reused in the query. The list below summarizes the pros and cons of this approach.

- + No language extensions are necessary; all required features are already implemented in SPARQL.
- + Queries are interoperable with other SPARQL engines (assuming the engine can interpret the similarity measure specification referenced).
- Individual similarity scores cannot be assigned to variables. They, hence, cannot be reused in the query for aggregation and ranking.
- Aggregation schemes are more complex to compose as they have to be specified within filter expressions.
- The performance is likely to be suboptimal as similarity scores have to be calculated repeatedly (as long as no caching mechanisms are used).

### 4.3 Solution Modifier Approach

We mention a third approach more for completeness than to elaborate it in detail. The approach adds a new (complex) *solution modifier* to the official W3C SPARQL grammar [22]. Solution modifiers (aka *sequence solution modifiers*) (1) take the sequence of solutions (set of solution mappings) which is returned after the SPARQL graph pattern matching is finished, (2) modify it according to their semantics, and (3) return a new sequence of solutions to the user – essentially, they perform a post-processing step. This was the approach we suggested in iRDQL [4], except that iRDQL allows the user to define only one similarity measure per query. The query in Listing 4.5 shows our example using only solution modifiers to define similarity strategies.

We could not identify any benefits of this approach compared to the virtual triple or extension function approach mentioned previously. From a query design point of view, data constraints should be specified in the `WHERE`-clause. Furthermore, solution modifiers can neither introduce new nor assign to existing query variables. Hence, there are no means to return the similarity scores to the user. Lastly, solution modifiers are not intended to access the ontology but only the result variables – an intention the similarity comparison between resources would break. Therefore, we decided to not further investigate this approach in this paper.



**Summary.** In this section, we proposed three approaches for extending SPARQL with similarity joins. Given our elaborations about the pros and cons of all three approaches, we claim that the virtual triple approach is superior to the others as it (1) permits to return computed similarity scores (which neither of the other approaches does) and (2) allows the user to elegantly specify aggregations/combinations of such scores for customized similarity functions (which the extension function approach does not). Its major drawback is the use of virtual triples that some might deem as conceptually problematic. We disagree: in some sense, the specification of a similarity function is akin to the specification of an additional inferencing rule. *Hence, virtual triples can simply be regarded as part of the inferred knowledge base.*

## 5 Experiments: Illustrating the Power of iSPARQL

To show the power of the resulting iSPARQL framework, we performed two sets of experiments: (1) a *data integration experiment* – combining information from different RDF datasets, and (2) an *ontology mapping experiment* – aligning different ontologies along their class descriptions. In earlier works [12,13], we already showed the power of iSPARQL for Semantic Web service matchmaking and similarity-based retrieval in large knowledge bases.

For both experiments, we used the SwetoDblp<sup>4</sup> and viewAIFB\_OWL<sup>5</sup> datasets (with the prefixes `opus` and `swrc`). The former focuses on bibliography information of Computer Science publications and is based on DBLP<sup>6</sup> whereas the latter is a collection of OWL annotations for persons, publications, and projects from SEAL (AIFB SEMantic portAL). For our ontology mapping experiments, we considered only the ontology schema files `opus` and `swrc` of SwetoDblp and viewAIFB\_OWL respectively. Both files are available online at the respective project’s websites.

### 5.1 Experiment 1: Semantic Web Data Integration

With the first set of experiments, we wanted to evaluate the applicability of our iSPARQL framework to the task of Semantic Web data integration. This task is highly relevant to distributed communities that want to integrate their heterogeneous knowledge bases (KB) to add, for example, cross-references, or to perform more sophisticated queries across different KBs to gain additional information. The problem arises when different parties have a different understanding of the same ontology for the same domain. This is a very typical situation in the Semantic Web, databases, and the Web in general: people working in the same or related field, model their domain ontologies (database schemas) similarly but, nevertheless, introduce syntactic, structural, and semantic differences.

In this section, we present our iSPARQL-based approach to Semantic Web data integration. To that end, we ran a slightly extended version of the query

<sup>4</sup> <http://lsdis.cs.uga.edu/projects/semdis/swetodblp/>

<sup>5</sup> [http://www.aifb.uni-karlsruhe.de/viewAIFB\\_OWL.owl](http://www.aifb.uni-karlsruhe.de/viewAIFB_OWL.owl)

<sup>6</sup> <http://dblp.uni-trier.de/>

**Table 5.1.** Results of the data integration experiments

opus:pub	swrc:pub	opus:title	opus:bt	opus:date	swrc:pp	sim
MikaISA04	id170instance	Ontology-based Content Management in a Virtual Organization	Handbook on Ontologies	2003-09-15	455-476	0.99
OberleS04	id206instance	The Knowledge Portal “OntoWeb”	Handbook on Ontologies	2004-03-30	499-517	0.94
OberleVSM04	id207instance	An Extensible Ontology Software Environment	Handbook on Ontologies	2004-03-30	311-333	0.93
SureSS04	id169instance	On-To-Knowledge Methodology (OTKM)	Handbook on Ontologies	2004-03-30	117-132	0.86

shown in Listing 4.2 on the two datasets SwetoDblp and viewAIFB\_OWL. Note that this query is very similar to the one presented by Lam *et al.* [15] on Page 6 that aims at integrating publication information from different drug datasets. Whereas they have to rely solely on Oracle’s SQL `regexp_like`-function<sup>7</sup> iSPARQL can take advantage of a whole library of different similarity measures (*i.e.*, SimPack).

The query’s final result set includes information from both datasets: title, proceedings name (bt), and a publication’s last modification date from SwetoDblp, and the number of pages from viewAIFB\_OWL, together with the similarity of the publications. The topmost results of the query based on similarity are shown in Table 5.1. All of the shown results provide the correct match between the two datasets. While this is not a statistical statement, we have shown the (statistical significant) usefulness of iSPARQL for the conceptually similar matchmaking and retrieval tasks elsewhere [5,12,13].

## 5.2 Experiment 2: Ontology Mapping

With our second set of experiments, we evaluated the applicability of our iSPARQL system to the task of ontology mapping. In other words, the task is to find classes in different ontologies, which *model the same real world concepts*.

To give a very simple example, we have chosen the query shown in Listing 5.1 that should find similar OWL classes in different ontologies. The query retrieves all resources which are of type `owl:Class`, filters out any anonymous nodes (complex classes), calculates the similarity of the class names using the Levenshtein similarity measure (see Section 3.2), and finally removes all solutions which are not sufficiently similar to each other.

The 5 topmost answers to the query are listed in Table 5.2 that shows classes of both ontologies together with their similarity scores. Note that we are aware of the extreme simplicity of this example. We think that it shows well, however, the potential of the iSPARQL framework. Imagine to use different similarity measures focusing on different modeling aspects of ontologies such as *data*, *structure*, and *context* [9], it is possible to compose much more complex similarity strategies than the one used in this experiment.

<sup>7</sup> Enabling regular expressions in queries.

---

```

1 SELECT ?OWLClass1 ?OWLClass2 ?similarity
2 FROM NAMED <http://lsdis.cs.uga.edu/projects/semdis/opus#>
3 FROM NAMED <http://swrc.ontoware.org/ontology#>
4 WHERE
5   { GRAPH opus:
6     { ?OWLClass1 rdf:type owl:Class .
7       FILTER ( !isBlank(?OWLClass1) )
8     }
9   GRAPH swrc:
10    { ?OWLClass2 rdf:type owl:Class .
11      FILTER ( !isBlank(?OWLClass2) )
12    }
13    ?similarity isparql:lev ( ?OWLClass1 ?OWLClass2 ) .
14    FILTER ( ?similarity > 0.65 )
15  }
16 ORDER BY DESC(?similarity)

```

---

**Listing 5.1.** iSPARQL example query for the ontology mapping task

**Table 5.2.** Results of the ontology mapping experiments

OWLClass1	OWLClass2	similarity
opus:Proceedings	swrc:Proceedings	1.0
opus:University	swrc:University	1.0
opus:Masters_Thesis	swrc:MasterThesis	0.857
opus:Proceedings	swrc:InProceedings	0.846
opus:Book	swrc:InBook	0.667

## 6 Limitations, Future Work, and Conclusions

In this paper, we have shown the syntactical and semantical foundations for a similarity join extension of SPARQL. A comparison of three such approaches, namely the virtual triple, the extension function, and the solution modifier approach has shown that the virtual triple approach is superior and provides SPARQL users with most flexibility in terms of defining customized similarity functions (CSF).

The major limitation of this approach lies in the need for extending existing SPARQL engines. As discussed above, we believe that the benefits warrants such an extension. Another limitation of the use of any similarity functions in SPARQL lies in the possibly enormous number of expensive (cross) joins involved during query execution. This problem can be addressed by (1) pre-computing similarity joins using a **CONSTRUCT**-statement as shown in Listing 4.3 or storing them in an index (as shown in 5); or by (2) re-ordering the triples such that similarity joins will be executed only on subsets of the overall ontology as constrained by the query, which we also explored in 5.

Having implemented an iSPARQL query engine as an extension to Jena ARQ, we intend to further investigate the potential for iSPARQL query optimization beyond the first steps shown in 5. Furthermore, we hope to embark on a systematic exploration of the suitability of similarity measures for different standard Semantic Web applications. Lastly, we intend to explore the possibility of extending description logic reasoners with customized similarity functions.

Whatever our further explorations will reveal, we firmly believe that the use of similarity functions is foundational for a large number of Semantic Web tasks and that this paper's discussion of the syntax and semantics of iSPARQL can provide a foundation for their use.

## References

1. Baeza-Yates, R., Ribeiro-Neto, B.: *Modern Information Retrieval* (1999)
2. Batagelj, V., Bren, M.: Comparing Resemblance Measures. *J. of Classification* 12(1), 73–90 (1995)
3. Bernstein, A., Kaufmann, E., Bürki, C., Klein, M.: How Similar Is It? Towards Personalized Similarity Measures in Ontologies. In: 7. Int. Tagung WI, pp. 1347–1366 (2005)
4. Bernstein, A., Kiefer, C.: Imprecise RDQL: Towards Generic Retrieval in Ontologies Using Similarity Joins. In: *Proc. of the ACM Symp. on Applied Computing*, pp. 1684–1689. ACM Press, New York (2006)
5. Bernstein, A., Kiefer, C., Stocker, M.: OptARQ: A SPARQL Optimization Approach based on Triple Pattern Selectivity Estimation. Technical Report IFI-2007.02, Department of Informatics, University of Zurich (2007)
6. Cohen, W.W.: Integration of Heterogeneous Databases Without Common Domains Using Queries Based on Textual Similarity. In: *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, pp. 201–212. ACM Press, New York (1998)
7. Cohen, W.W., Ravikumar, P., Fienberg, S.: A Comparison of String Distance Metrics for Name-Matching Tasks. In: *Proc. of the Ws. on Information Integration on the Web*, Acapulco, Mexico, pp. 73–78 (2003)
8. Cyganiak, R.: A relational algebra for SPARQL. Technical Report HPL-2005-170, HP Labs (2005)
9. Ehrig, M., Haase, P., Stojanovic, N., Hefke, M.: Similarity for Ontologies - A Comprehensive Framework. In: *Proc. of the 13th Europ. Conf. on Information Systems* (2005)
10. Euzenat, J., Loup, D., Touzani, M., Valtchev, P.: Ontology Alignment with OLA. In: *Proc. of the 3rd Int. Ws. on Evaluation of Ontology-based Tools*, pp. 59–68 (2004)
11. Gravano, L., Ipeirotis, P.G., Koudas, N., Srivastava, D.: Text Joins in an RDBMS for Web Data Integration. In: *Proc. of the 12th Int. World Wide Web Conf.*, pp. 90–101 (2003)
12. Kiefer, C., Bernstein, A., Lee, H.J., Klein, M., Stocker, M.: Semantic Process Retrieval with iSPARQL. In: *Proc. of the 4th Europ. Semantic Web Conf.*, pp. 609–623 (2007)
13. Kiefer, C., Bernstein, A., Tappolet, J.: Analyzing Software with iSPARQL. In: *Proc. of the 3rd Int. Ws. on Semantic Web Enabled Software Engineering* (2007)
14. Klusch, M., Fries, B., Sycara, K.: Automated Semantic Web Service Discovery with OWLS-MX. In: *Proc. of the 5th Int. Joint Conf. on Autonomous Agents and Multiagent Systems*, pp. 915–922 (2006)
15. Lam, H.Y.K., Marenco, L., Clark, T., Gao, Y., Kinoshita, J., Shepherd, G., Miller, P., Wu, E., Wong, G., Liu, N., Crasto, C., Morse, T., Stephens, S., Cheung, K.-H.: AlzPharm: integration of neurodegeneration data using RDF. *BMC Bioinformatics* 8(3) (2007)
16. Levenshtein, V.I.: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady* 10, 707–710 (1966)

17. Meštrović, A., Čubrillo, M.: Semantic Web Data Integration Using F-Logic. In: Proc. of the 10th Int. Conf. on Intelligent Engineering Systems (2006)
18. Noy, N.F.: What do we need for ontology integration on the Semantic Web, Position statement. In: Proc. of the 1st Semantic Integration Ws., pp. 175–176 (2003)
19. Noy, N.F., Musen, M.A.: The PROMPT Suite: Interactive Tools For Ontology Merging And Mapping. *Int. J. of Human-Computer Studies* 59(6), 983–1024 (2003)
20. Orozco, J., Belanche, L.: On Aggregation Operators of Transitive Similarity and Dissimilarity Relations. In: Proc. of the IEEE Int. Conf. on Fuzzy Systems, pp. 1373–1377. IEEE Computer Society Press, Los Alamitos (2004)
21. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and Complexity of SPARQL. In: Proc. of the 5th Int. Semantic Web Conf., pp. 30–43 (2006)
22. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. Technical report. W3C Candidate Recommendation 14 June (2007)
23. Rorvig, M.: Images of Similarity: A Visual Exploration of Optimal Similarity Metrics and Scaling Properties of TREC Topic-Document Sets. *J. of the Am. Soc. for Inf. Sci.* 50(8), 639–651 (1999)
24. Shasha, D., Zhang, K.: Approximate Tree Pattern Matching. In: Pattern Matching in Strings, Trees, and Arrays, pp. 341–371 (1997)
25. Siberski, W., Pan, J.Z., Thaden, U.: Querying the Semantic Web with Preferences. In: Proc. of the 5th Int. Semantic Web Conf. (2006)
26. Tversky, A.: Features of Similarity. *Psychological Review* 84(2), 327–353 (1977)

## A Extended iSPARQL Example Query

---

```

1 SELECT ?publication1 ?publication2 ?similarity
2 WHERE
3   { ?publication1 rdfs:label ?title1 .
4     ?publication2 swrc:title ?title2 .
5     ?publication1 opus:book_title ?booktitle1 .
6     ?publication2 swrc:booktitle ?booktitle2 .
7
8     IMPRECISE {
9       ?sim1 isparql:jac (?title1 ?title2) .
10      ?sim2 isparql:lev (?booktitle1 ?booktitle2) .
11      ?sim3 isparql:score (?sim1 ?sim2 0.9 0.1) .
12      FILTER (?sim3 > 0.2) }
13
14     ?publication1 opus:hasAuthor ?author1 .
15     ?publication2 swrc:author ?author2 .
16
17     ?author1 foaf:lastname ?lastname1 .
18     ?author2 foaf:lastname ?lastname2 .
19     ?author1 foaf:firstname ?firstname1 .
20     ?author2 foaf:firstname ?firstname2 .
21
22     IMPRECISE {
23       ?sim4 isparql:lev (?lastname1 ?lastname2) .
24       FILTER (?sim4 >= 0.5) .
25       ?sim5 isparql:lev (?firstname1 ?firstname2) .
26       FILTER (?sim5 >= 0.5) .
27       ?sim6 isparql:score (?sim4 ?sim5 0.7 0.3) }
28
29     ?publication1 opus:abstract ?abstract1 .
30     ?publication2 swrc:abstract ?abstract2 .
31
32     IMPRECISE {
33       ?sim7 isparql:tfidf (?abstract1 ?abstract2) .
34       FILTER (?sim7 >= 0.7) .
35       ?similarity isparql:average (?sim3 ?sim6 ?sim7) }
36   }

```

---

**Listing A.1.** Extended iSPARQL example query for the virtual triple approach

# Conjunctive Queries for a Tractable Fragment of OWL 1.1<sup>\*</sup>

Markus Krötzsch, Sebastian Rudolph, and Pascal Hitzler

Institut AIFB, Universität Karlsruhe, Germany

**Abstract.** Despite the success of the Web Ontology Language OWL, the development of expressive means for querying OWL knowledge bases is still an open issue. In this paper, we investigate how a very natural and desirable form of queries—namely conjunctive ones—can be used in conjunction with OWL such that one of the major design criteria of the latter—namely decidability—can be retained. More precisely, we show that querying the tractable fragment  $\mathcal{EL}^{++}$  of OWL 1.1 is decidable. We also provide a complexity analysis and show that querying unrestricted  $\mathcal{EL}^{++}$  is undecidable.

## 1 Introduction

Conjunctive queries originated from research in relational databases [1], and, more recently, have also been identified as a desirable form of querying expressive description logics (DLs) that are underlying OWL [2][3][4][5][6]. At the same time, tractable fragments of OWL are receiving increasing attention as they promise to provide a favourable balance between expressivity and scalability. Such fragments have, in particular, been identified as part of the OWL 1.1 proposal [7] and this raises the question how conjunctive queries can be combined favourably with the underlying description logics.

In this paper, we thus present the very first algorithm for answering conjunctive queries in the tractable  $\mathcal{EL}^{++}$ -fragment of *SROIQ* [7][8], and thus of OWL 1.1. The algorithm is based on an automata-theoretic formulation of complex role inclusion axioms that was also found useful in reasoning with *SROIQ* [9][10].

Our algorithm in particular allows us to derive a number of complexity results related to conjunctive query answering in  $\mathcal{EL}^{++}$ . We first show that conjunctive queries in  $\mathcal{EL}^{++}$  are undecidable in general, and identify the  $\mathcal{EL}^{++}$ -fragment of *SROIQ* as an appropriate decidable sub-DL. Under some related restrictions of role inclusion axioms, we show that conjunctive query answering in general is PSPACE-complete. Query answering for fixed knowledge bases (query complexity) is shown to be NP-complete, whereas for fixed queries (schema complexity) it is merely P-complete.

## 2 Preliminaries

We assume the reader to be familiar with the basic notions of description logics (DLs). The DLs that we will encounter in this paper are  $\mathcal{EL}^{++}$  [7] and, marginally, *SROIQ*

<sup>\*</sup> This work has been supported by the European Commission under contract IST-2006-027595 NeOn, and by the Deutsche Forschungsgemeinschaft (DFG) under the ReaSem project.

<sup>1</sup> See <http://webont.org/owl/1.1/> for both.

[10]. A DL *signature* consists of a finite set of *role names*  $\mathbf{R}$ , a finite set of *individual names*  $\mathbf{I}$ , and a finite set of *concept names*  $\mathbf{C}$ . We will use this notation throughout the paper.  $\mathcal{EL}^{++}$  supports *nominals*, which we conveniently represent as follows: for any  $a \in \mathbf{I}$ , there is a concept  $\{a\} \in \mathbf{C}$  such that  $\{a\}^I = \{a^I\}$  (for any interpretation  $I$ ). As shown in [7], any  $\mathcal{EL}^{++}$  knowledge base is equivalent to one in *normal form*, only containing the following axioms:

$$\begin{array}{llll} \text{TBox: } & A \sqsubseteq C & A \sqcap B \sqsubseteq C & A \sqsubseteq \exists R.C \quad \exists R.A \sqsubseteq C \\ \text{RBox: } & R \sqsubseteq T & R \circ S \sqsubseteq T & \end{array}$$

where  $A, B \in \mathbf{C} \cup \{\top\}$ ,  $C \in \mathbf{C} \cup \{\perp\}$ , and  $R, S, T \in \mathbf{R}$ . Note that ABox statements of the forms  $C(a)$  and  $R(a, b)$  are internalised into the TBox. The standard model theoretic semantics of  $\mathcal{EL}^{++}$  can be found in [7]. Unless otherwise specified, the letters  $C, D, E$  in the remainder of this work always denote (arbitrary) concept names, and the letters  $R, S$  denote (arbitrary) role names. We do not consider concrete domains in this paper, but are confident that our results can be extended accordingly.

For *conjunctive queries*, we largely adopt the notation of [6] but directly allow for individuals in queries. Let  $\mathbf{V}$  be a countable set of *variable names*. Given elements  $x, y \in \mathbf{V} \cup \mathbf{I}$ , a *concept atom* (*role atom*) is an expression  $C(x)$  with  $C \in \mathbf{C}$  ( $R(x, y)$  with  $R \in \mathbf{R}$ ). A *conjunctive query*  $q$  is a set of concept and role atoms, read as a conjunction of its elements. By  $\text{Var}(q)$  we denote the set of variables occurring in  $q$ . Consider an interpretation  $I$  with domain  $\Delta^I$ , and a function  $\pi : \text{Var}(q) \cup \mathbf{I} \rightarrow \Delta^I$  such that  $\pi(a) = a^I$  for all  $a \in \mathbf{I}$ . We define

$$I, \pi \models C(x) \text{ if } \pi(x) \in C^I, \quad \text{and} \quad I, \pi \models R(x, y) \text{ if } (\pi(x), \pi(y)) \in R^I.$$

If there is some  $\pi$  such that  $I, \pi \models A$  for all atoms  $A \in q$ , we write  $I \models q$  and say that  $I$  *entails*  $q$ . We say that  $q$  is entailed by a knowledge base  $KB$ , denoted  $KB \models q$ , if all models of  $KB$  entail  $q$ .

We conclude this section with an important result on conjunctive queries in  $\mathcal{EL}^{++}$ .

**Theorem 1.** *For an  $\mathcal{EL}^{++}$  knowledge base  $KB$  and a conjunctive query  $q$ , the entailment problem  $KB \models q$  is undecidable. Likewise, checking class subsumptions in  $\mathcal{EL}^{++}$  extended with inverse roles or role conjunctions is undecidable, even if those operators occur only in the concepts whose subsumption is checked.*

Intuitively, the result holds since RBoxes can encode context-free languages, the intersection of which can then be checked with conjunctive queries/inverse roles/role conjunctions. This problem is undecidable (see [11] for a proof). Clearly, arbitrary role compositions are too expressive when aiming for a decidable (or even tractable) logic that admits conjunctive queries. We thus restrict our attention to the fragment of  $\mathcal{EL}^{++}$  that is in the (decidable) description logic  $\mathcal{SROIQ}$  [10], and investigate its complexity with respect to conjunctive query answering.

**Definition 1.** *An  $\mathcal{EL}^{++}$  RBox in normal form is regular if there is a strict partial order  $<$  on  $\mathbf{R}$  such that, for all role inclusion axioms  $R_1 \sqsubseteq S$  and  $R_1 \circ R_2 \sqsubseteq S$ , we find  $R_i < S$  or  $R_i = S$  ( $i = 1, 2$ ). An  $\mathcal{EL}^{++}$  knowledge base is regular if it has a regular RBox.*

The existence of  $<$  ensures that the role hierarchy does not contain cyclic dependencies other than through direct recursion of a single role.



**Table 1.** Closure rules for an interpretation  $\mathcal{I}$  w.r.t. some knowledge base  $KB$ . In general, we assume that  $C, D \in \mathbf{C} \cup \{\top, \perp\}$  and  $R_1, R_2, S \in \mathbf{R}$ .

$$\begin{array}{l}
(1) \quad \frac{\delta \in C^{\mathcal{I}} \quad KB \models C \sqsubseteq D}{D^{\mathcal{I}} := D^{\mathcal{I}} \cup \{\delta\}} \\
(2) \quad \frac{\delta \in C^{\mathcal{I}} \quad KB \models C \sqsubseteq \exists R.D \quad KB \not\models D \sqsubseteq \{a\} \text{ for any } a \in \mathbf{I}}{\Delta_{\mathcal{I}} := \Delta_{\mathcal{I}} \cup \{\epsilon\} \quad R^{\mathcal{I}} := R^{\mathcal{I}} \cup \{(\delta, \epsilon)\} \quad D^{\mathcal{I}} := D^{\mathcal{I}} \cup \{\epsilon\}} \text{ where } \epsilon = \epsilon_{\delta, C \sqsubseteq \exists R.D} \\
(3) \quad \frac{\delta \in C^{\mathcal{I}} \quad KB \models C \sqsubseteq \exists R.D \quad KB \models D \sqsubseteq \{a\} \text{ for some } a \in \mathbf{I}}{R^{\mathcal{I}} := R^{\mathcal{I}} \cup \{(\delta, a)\}} \\
(4) \quad \frac{(\delta, \epsilon) \in R^{\mathcal{I}} \quad R \sqsubseteq S \in KB}{S^{\mathcal{I}} := S^{\mathcal{I}} \cup \{(\delta, \epsilon)\}} \quad (5) \quad \frac{(\delta, \epsilon) \in R_1^{\mathcal{I}} \quad (\epsilon, \gamma) \in R_2^{\mathcal{I}} \quad R_1 \circ R_2 \sqsubseteq S \in KB}{S^{\mathcal{I}} := S^{\mathcal{I}} \cup \{(\delta, \gamma)\}}
\end{array}$$

### 3 Canonical Models and Reasoning Automata for $\mathcal{EL}^{++}$

$\mathcal{EL}^{++}$ , like all Horn-DLs, allows for the construction of *canonical* or *universal* models. By this we mean an interpretation that is in a sense most general among the models of a given  $\mathcal{EL}^{++}$  knowledge base, satisfying exactly those formulae that are logical consequences of the knowledge base. This notion could be formalised further (using the concept of *(bi)simulation* between models), but we merely require canonical models to guide us in the development and verification of a query answering algorithm, and hence we will confine ourselves to directly showing the relevant properties.

Consider a regular consistent  $\mathcal{EL}^{++}$  knowledge base  $KB$ . Here and in the following, we assume w.l.o.g. that  $KB$  does not entail  $a \approx b$  (i.e.  $\{a\} \equiv \{b\}$ ) for any  $a, b \in \mathbf{I}$ . Indeed, one can just replace all occurrences of  $b$  with  $a$  in this case, both within  $KB$  and within any query we wish to consider later on (and this case can be detected in polynomial time). Moreover, we assume that there is at least one individual in the language, i.e.  $\mathbf{I} \neq \emptyset$ . We now provide an iterative construction of a model  $\mathcal{I}$  of  $KB$ . Our goal is to obtain a concise definition of a suitable canonical model, so it is no matter of concern that the given construction does not terminate after finitely many steps.

To simplify our arguments, we adopt a naming scheme for potential elements of the domain of  $\mathcal{I}$ . Let  $\Delta$  be the smallest set such that  $\mathbf{I} \subseteq \Delta$  and, for any  $\delta \in \Delta$ ,  $C, D \in \mathbf{C}$ , and  $R \in \mathbf{R}$ , we find that  $\epsilon_{\delta, C \sqsubseteq \exists R.D} \in \Delta$ . We will define  $\mathcal{I}$  such that  $\Delta_{\mathcal{I}} \subseteq \Delta$ .

For any two interpretations  $\mathcal{J}_1$  and  $\mathcal{J}_2$  of  $KB$ , we say that  $\mathcal{J}_1$  is *smaller* than (or equal to)  $\mathcal{J}_2$  if, for any  $F \in \mathbf{C} \cup \mathbf{R} \cup \{\top\}$ ,  $F^{\mathcal{J}_1} \subseteq F^{\mathcal{J}_2}$ . The interpretation  $\mathcal{I}$  is defined to be the smallest interpretation such that  $\Delta_{\mathcal{I}} \subseteq \Delta$ ,  $\{a\}^{\mathcal{I}} := a$  for all  $a \in \mathbf{I}$ , and  $\mathcal{I}$  is closed under the rules of Table 1. It is easy to see that this smallest interpretation exists.

The rules of Table 1 have the special property that each individual is “initialised” with at most one concept name. Formally, we define for each element  $\delta \in \Delta_{\mathcal{I}}$  a concept name  $\iota(\delta)$  as follows:

- if  $\delta \in \mathbf{I}$ ,  $\iota(\delta) := \{\delta\}$ ,
- if  $\delta = \epsilon_{\delta', C \sqsubseteq \exists R.D}$  for some  $\delta' \in \Delta_{\mathcal{I}}$ ,  $C, D \in \mathbf{C}$ ,  $R \in \mathbf{R}$ , then  $\iota(\delta) := D$ .

Note that the above cases are indeed exhaustive and mutually exclusive.



**Lemma 1.** *The interpretation  $\mathcal{I}$  as constructed above is a model of  $KB$ .*

*Proof.* First note that the domain of  $\mathcal{I}$  is non-empty since we assume the existence of at least one individual. We have to check that all axioms of  $KB$  are indeed satisfied. For axioms of the form  $C \sqsubseteq \exists R.D$  this is obvious by rules (2) and (3) of Table [1](#). Similarly, all role inclusion axioms are directly accounted for by rules (4) and (5).

So it remains to show that axioms  $\Phi$  of the forms  $C \sqsubseteq D$ ,  $\exists R.C \sqsubseteq D$ , and  $C_1 \sqcap C_2 \sqsubseteq D$  are satisfied. Obviously, whenever  $\delta \in C^{\mathcal{I}}$  ( $\delta \in \exists R.C^{\mathcal{I}}$ ) for some  $C \in \mathbf{C}$  (and  $R \in \mathbf{R}$ ), we find  $KB \models \iota(\delta) \sqsubseteq C$  ( $KB \models \iota(\delta) \sqsubseteq \exists R.C$ ). We conclude that, whenever the premise of some axiom  $\Phi$  as above is satisfied for  $\delta$ , then it is entailed by  $\iota(\delta)$ , and so its conclusion  $D$  is a direct consequence of  $\iota(\delta)$  under  $KB$ . Thus  $\Phi$  is satisfied by rule (1).  $\square$

We are most interested in the specific structure of the canonical model. Its construction attempts to preserve a form of tree-likeness, broken only by the potential occurrence of nominals. Formally, this is expressed through the following property.

*Property 1.* For any element  $\delta \in \Delta_{\mathcal{I}}$  that is not an individual ( $\delta \notin \mathbf{I}$ ), there is a unique chain of elements  $\delta_0 \dots \delta_k = \delta$  and role names  $R_0, \dots, R_{k-1} \in \mathbf{R}$ , such that  $\delta_0 \in \mathbf{I}$  and, for all  $i = 1, \dots, k$ ,  $\delta_i \in \Delta_{\mathcal{I}}$  is of the form  $\delta_{\epsilon, C \sqsubseteq R.D}$  with  $\epsilon = \delta_{i-1}$  and  $R = R_{i-1}$ . This is easily verified by observing that any  $\delta$  of the given form must have been entailed by rule (2), and by applying a simple induction on the depth of this entailment. In this case, we say that  $\delta_i$  *generates*  $\delta$  via the roles  $R_i \dots R_k$  ( $i = 0, \dots, k$ ).

The canonicity of the model  $\mathcal{I}$  manifests itself in the fact that structures in the model are necessary logical consequences of given axioms.

*Property 2.* Consider elements  $\delta, \epsilon \in \Delta_{\mathcal{I}}$  such that  $\delta$  generates  $\epsilon$  via the roles  $R_0 \dots R_k$ . Then  $\iota(\delta) \sqsubseteq \exists R_0.(\dots \exists R_k.\iota(\epsilon) \dots)$  is a consequence of  $KB$ . This is obvious by another simple inductive argument that utilises the preconditions of the applications of rule (3).

*Property 3.* For any  $(\delta, \epsilon) \in R^{\mathcal{I}}$ , there is a chain of elements  $\delta = \delta_0 \dots \delta_k = \epsilon$  and role names  $R_i$  ( $i = 0, \dots, k-1$ ), such that

- $(\delta_i, \delta_{i+1}) \in R_i^{\mathcal{I}}$  is directly entailed by one of rules (2) and (3), and
- $R_0 \circ \dots \circ R_{k-1} \sqsubseteq R$  is a consequence of  $KB$ .

We show this by an inductive argument as follows: for the base case, assume that  $(\delta, \epsilon) \in R^{\mathcal{I}}$  follows from rule (2) or (3). Then the above condition clearly holds. For the induction step, assume that  $(\delta, \epsilon) \in R^{\mathcal{I}}$  follows by applying rule (5) to  $R_1 \circ R_2 \sqsubseteq R$ , and that the claim holds for the statements  $(\delta, \delta_j) \in R_1^{\mathcal{I}}$  and  $(\delta_j, \epsilon) \in R_2^{\mathcal{I}}$ . We easily can construct from these assumptions a suitable chain of elements from the chains postulated for  $R_1$  and  $R_2$ . Similarly, the second condition of the claim follows from the assumption that  $R_1 \circ R_2 \sqsubseteq R$  and the induction hypothesis. Rule (4) is treated analogously.

In the remainder of this section, we investigate various means of presenting logical inferences by means of automata. These encodings will play a major role within our subsequent query answering algorithm. We describe nondeterministic finite automata (NFA)  $\mathcal{A}$  as tuples  $(Q_{\mathcal{A}}, \Sigma_{\mathcal{A}}, \delta_{\mathcal{A}}, i_{\mathcal{A}}, F_{\mathcal{A}})$ , where  $Q_{\mathcal{A}}$  is a finite set of states,  $\Sigma_{\mathcal{A}}$  is a finite alphabet,  $\delta_{\mathcal{A}} : Q_{\mathcal{A}} \times Q_{\mathcal{A}} \rightarrow 2^{\Sigma_{\mathcal{A}}}$  is a transition function that maps pairs of states to sets of alphabet symbols,  $i_{\mathcal{A}}$  is the initial state, and  $F_{\mathcal{A}}$  is a set of final states.

<sup>2</sup> A possibly more common definition is to map pairs of states and symbols to sets of states, but the above is more convenient for our purposes.

**Table 2.** Completion rules for constructing an NFA from an  $\mathcal{EL}^{++}$  knowledge base  $KB$ 

- (CR1) If  $C' \in \delta(C, C)$ ,  $C' \sqsubseteq D \in KB$ , and  $D \notin \delta(C, C)$  then  $\delta(C, C) := \delta(C, C) \cup \{D\}$ .  
 (CR2) If  $C_1, C_2 \in \delta(C, C)$ ,  $C_1 \sqcap C_2 \sqsubseteq D \in KB$ , and  $D \notin \delta(C, C)$  then  $\delta(C, C) := \delta(C, C) \cup \{D\}$ .  
 (CR3) If  $C' \in \delta(C, C)$ ,  $C' \sqsubseteq \exists R.D \in KB$ , and  $R \notin \delta(C, D)$  then  $\delta(C, D) := \delta(C, D) \cup \{R\}$ .  
 (CR4) If  $R \in \delta(C, D)$ ,  $D' \in \delta(D, D)$ ,  $\exists R.D' \sqsubseteq E \in KB$ , and  $E \notin \delta(C, C)$  then  $\delta(C, C) := \delta(C, C) \cup \{E\}$ .  
 (CR5) If  $R \in \delta(C, D)$ ,  $\perp \in \delta(D, D)$ , and  $\perp \notin \delta(C, C)$  then  $\delta(C, C) := \delta(C, C) \cup \{\perp\}$ .  
 (CR6) If  $\{a\} \in \delta(C, C) \cap \delta(D, D)$ , and there are states  $C_1, \dots, C_n$  such that
  - $C_1 \in \{C, \top, A\} \cup \{\{b\} \mid b \in \mathbf{I}\}$ ,
  - $\delta(C_j, C_{j+1}) \neq \emptyset$  for all  $j = 1, \dots, n-1$ ,
  - $C_n = D$ ,
 and  $\delta(D, D) \not\subseteq \delta(C, C)$  then  $\delta(C, C) := \delta(C, C) \cup \delta(D, D)$ .  
 (CR7) If  $R \in \delta(C, D)$ ,  $R \sqsubseteq S$ , and  $S \notin \delta(C, D)$  then  $\delta(C, D) := \delta(C, D) \cup \{S\}$ .  
 (CR8) If  $R_1 \in \delta(C, D)$ ,  $R_2 \in \delta(D, E)$ ,  $R_1 \circ R_2 \sqsubseteq S$ , and  $S \notin \delta(C, E)$  then  $\delta(C, E) := \delta(C, E) \cup \{S\}$ .

**Proposition 1.** *Given a regular  $\mathcal{EL}^{++}$  RBox, and some role  $R \in \mathbf{R}$ , there is an NFA  $\mathcal{A}(R)$  over the alphabet  $\mathbf{R}$  which accepts a word  $R_1 \dots R_n$  iff  $R_1 \circ \dots \circ R_n \sqsubseteq R$  is a consequence of every  $\mathcal{EL}^{++}$  knowledge base with the given RBox.*

One possible construction for the required automaton is discussed in [10]. Intuitively, the RBox can be understood as a grammar for a regular language, for which an automaton can be constructed in a canonical way. The required construction of  $\mathcal{A}(R)$  might be exponential for some RBoxes. In [9], restrictions have been discussed that prevent this blow-up, leading to NFA of only polynomial size w.r.t. the RBox. Accordingly, an RBox is *simple* whenever, for all axioms of the form  $R_1 \circ S \sqsubseteq S$ ,  $S \circ R_2 \sqsubseteq S$ , the RBox does not contain a common subrole  $R$  of  $R_1$  and  $R_2$  for which there is an axiom of the form  $R \circ S' \sqsubseteq R'$  or  $S' \circ R \sqsubseteq R'$ . We will usually consider only such simple RBoxes whenever the size of the constructed automata matters.

Next we describe the construction of a novel kind of automaton that encodes certain concept subsumptions entailed by an  $\mathcal{EL}^{++}$  knowledge base. The automaton itself is closely related to the reasoning algorithm given in [7], but the representation of entailments via nondeterministic finite automata (NFA) will be essential for the query answering algorithm in the following section.

Consider an  $\mathcal{EL}^{++}$  knowledge base  $KB$ . Given a concept name  $A \in \mathbf{C}$ , we construct an NFA  $\mathcal{A}_{KB}(A) = (Q, \Sigma, \delta, i, F)$  that computes superconcepts of  $A$ , where we omit the subscript if  $KB$  is clear from the context. Set  $Q = F = \mathbf{C} \cup \{\top\}$ ,  $\Sigma = \mathbf{C} \cup \mathbf{R} \cup \{\top, \perp\}$ , and  $i = A$ . The transition function  $\delta$  is initially defined as  $\delta(C, C) := \{C, \top\}$  (for all  $C \in Q$ ) and  $\delta(C, D) := \emptyset$  (for all  $C, D \in Q$  with  $C \neq D$ ), and extended iteratively by applying the rules in Table 2. The rules correspond to completion rules in [7] Table 2, though the conditions for (CR6) are slightly relaxed, fixing a minor glitch in the original algorithm.

It is easy to see that the rules of Table 2 can be applied at most a polynomial number of times. The words accepted by  $\mathcal{A}(A)$  are strings of concept and role names. For each such word  $w$  we inductively define a concept expression  $C_w$  as follows:

- if  $w$  is empty, then  $C_w = \top$ ,
- if  $w = Rv$  for some  $R \in \mathbf{R}$  and word  $v$ , then  $C_w = \exists R.(C_v)$ ,
- if  $w = Cv$  for some  $C \in \mathbf{C}$  and word  $v$ , then  $C_w = C \sqcap C_v$ .

For instance, the word  $CRDES$  translates into  $C_{CRDES} = C \sqcap \exists R.(D \sqcap E \sqcap \exists S. \top)$ . Based on the close correspondence of the above rules to the derivation rules in [7], we can now establish the main correctness result for the automaton  $\mathcal{A}(A)$ .

**Theorem 2.** *Consider a knowledge base  $KB$ , concept  $A$ , and NFA  $\mathcal{A}(A)$  as above, and let  $w$  be some word over the associated alphabet. Then  $KB \models A \sqsubseteq C_w$  iff one of the following holds:*

- $\mathcal{A}(A)$  accepts the word  $w$ , or
- there is a transition  $\perp \in \delta(C, C)$  where  $C = \top$ ,  $C = A$ , or  $C = \{a\}$  for some individual  $a$ .

In particular,  $\mathcal{A}(A)$  can be used to check all subsumptions between  $A$  and some atomic concept  $B$ .

The second item of the theorem addresses the cases where  $A$  is inferred to be empty (i.e. inconsistent) or where the whole knowledge base is inconsistent, from which the subsumption trivially follows. While the above yields an alternative formulation of the  $\mathcal{EL}^{++}$  reasoning algorithm presented in [7], it has the advantage that it also encodes all *paths* within the inferred models. This will be essential for our results in the next section. The following definition will be most convenient for this purpose.

**Definition 2.** *Consider a knowledge base  $KB$ , concepts  $A, B \in \mathbf{C}$ , and the NFA  $\mathcal{A}(A) = (Q, \Sigma, \delta, i, F)$ . The automaton  $\mathcal{A}_{KB}(A, B)$  (or just  $\mathcal{A}(A, B)$ ) is defined as  $(Q, \mathbf{R}, \delta', i, F')$  where  $F' = \emptyset$  if  $\perp \in \delta(A, A)$  and  $F' = \{B\}$  otherwise, and  $\delta'$  is the restriction of  $\delta$  to  $\mathbf{R}$ .*

The automaton  $\mathcal{A}(A, B)$  normally accepts all words of roles  $R_1, \dots, R_n$  such that  $A \sqsubseteq \exists R_1(\dots \exists R_n.B \dots)$  is a consequence of  $KB$ , with the border case where  $n = 0$  and  $KB \models A \sqsubseteq B$ . Moreover, the language accepted by the NFA is empty whenever  $A \sqsubseteq \perp$  has been inferred.

## 4 Deciding Conjunctive Queries for $\mathcal{EL}$

In this section, we present a nondeterministic algorithm that decides the entailment of a query  $q$  with respect to some regular consistent knowledge base  $KB$ . The algorithm constructs a so-called *proof graph* which establishes, for all interpretations  $\mathcal{I}$  of  $KB$ , the existence of a suitable function  $\pi$  that shows query entailment. Intuitively, a proof graph encodes a fragment of the canonical model  $\mathcal{I}$  of Section 3.

Formally, a proof graph is a tuple  $(N, L, E)$  consisting of a set of nodes  $N$ , a labelling function  $L : N \rightarrow \mathbf{C} \cup \{\top\}$ , and a *partial* transition function  $E : N \times N \rightarrow \mathbf{A}$ , where  $\mathbf{A}$  is the set of all NFA over the alphabet  $\mathbf{C} \cup \{\top, \perp\} \cup \mathbf{R}$ . A node  $m \in N$  is *reachable* if there is some node  $n \in N$  such that  $E(n, m)$  is defined, and *unreachable* otherwise. The nodes of the proof graph are abstract representations of elements in the domain of the

**Table 3.** A nondeterministic algorithm for deciding conjunctive queries in  $\mathcal{EL}^{++}$ 

A. Query factorisation	1	Select a (possibly empty) set $X \subseteq \text{Var}(q)$
	2	For each $x \in X$
	3	Select some $e \in \text{Var}(q) \cup \mathbf{I}$ and replace all occurrences of $x$ in $q$ with $e$
B. Initialise proof graph $(N, L, E)$	4	$N := \mathbf{I} \cup \text{Var}(q)$ , let $E$ be undefined for all arguments
	5	For each $a \in \mathbf{I}$ , $L(a) := \{a\}$
	6	For each $x \in \text{Var}(q)$ , select $L(x) \in \mathbf{C} \cup \{\top\}$
	7	For each $n \in N$ , $a \in \mathbf{I}$ , $E(n, a) := \mathcal{A}(L(n), L(a))$
	8	While there is an unreachable node
	9	Select some unreachable $x \in \text{Var}(q)$ , select some reachable $n \in N$
C. Check proof graph	10	$E(n, x) := \mathcal{A}(L(n), L(x))$
	11	For each $n \in N$ , $m \in \text{Var}(q)$
D. Check concept entailment	12	If $E(n, m)$ is defined and accepts no word, terminate with failure
	13	For each concept atom $C(n) \in q$
E. Split role automata	14	If not $KB \models L(n) \sqsubseteq C$ , terminate with failure
	15	For each role atom $R(n, m) \in q$
	16	Compute shortest path $n = n_0, \dots, n_k = m$ from $n$ to $m$
	17	Split $\mathcal{A}(R)$ into $k$ automata $\mathcal{A}(R(n, m), n_0, n_1), \dots, \mathcal{A}(R(n, m), n_{k-1}, n_k)$
	18	For each $\mathcal{A}(R(n, m), n_{i-1}, n_i)$
F. Check role entailment	19	If $\mathcal{A}(R(n, m), n_{i-1}, n_i)$ accepts no word, terminate with failure
	20	$acc := \text{true}$
	21	For each $n, m \in N$ with $E(n, m)$ defined
	22	If $m \in \mathbf{I}$
	23	For each split automaton $\mathcal{A}(F, n, m)$
	24	If $\mathcal{A}(F, n, m)$ and $E(n, m)$ do not accept a common word
	25	$acc := \text{false}$
	26	Else if $m \in \text{Var}(q)$
	27	If no word is accepted by $E(n, m)$ and all split automata $\mathcal{A}(F, n, m)$
	28	$acc := \text{false}$
	29	If $acc$ is <b>false</b> , then terminate with failure
	30	Else accept the query

canonical model  $I$  of  $KB$ . The labels assign a concept to each node, the intuition being that this is the “main concept”  $\iota(\delta)$  defined in Section 3. Finally, the transition function encodes role paths in the canonical model, which provide the basis for inferencing about relationships between elements. It would be possible to adopt a more concrete representation for role paths (e.g. by guessing a single path), but our formulation reduces nondeterminism and eventually simplifies our investigation of algorithmic complexity.

Our algorithm for deciding conjunctive query entailment is given in Table 3. Any occurrence of the word “select” in the description indicates a nondeterministic choice of the algorithm. Step A is a standard preprocessing step for many query answering algorithms. Step B initiates the proof graph and ensures that all nodes are reachable. Variable nodes eventually are reachable through exactly one predecessor node, so the structure of the proof graph resembles the canonical model (compare Property 1 of Section 3). Steps C and D verify that the selected proof graph indeed establishes the existence of the required anonymous elements in the model (C) and the entailment of

the query’s concept atoms (D). At this stage, the proof graph still represents many possible fragments of the canonical model: the edge NFA that connect to variable nodes encode possible generating role paths (in the sense of Property 1 Section 3), each of which leads to a different element in the canonical model. The edges leading to individual nodes have a slightly different meaning: all of the paths they represent must actually exist in any model. Summing up, the proof graph still represents many possible matches between the query and a model of  $KB$ , though a number of basic decisions on the structure of the considered matches has already been made and it is known that any such match suffices to entail the concept atoms of the query.

Now Step E computes the RBox automata  $\mathcal{A}(R)$  of Section 3 and applies a nondeterministic *splitting* operation, which we define next. We remark that the required “shortest path” exists and is easily found in polynomial time (see [11]).

**Definition 3.** Consider an NFA  $\mathcal{A} = (Q, \Sigma, \delta, i, \{f\})$ . A split of  $\mathcal{A}$  into  $k$  parts is given by NFA  $\mathcal{A}_1, \dots, \mathcal{A}_k$  with  $\mathcal{A}_j$  of the form  $(Q, \Sigma, \delta, q_{j-1}, \{q_j\})$  such that  $q_0 = i, q_k = f$ , and  $q_j \in Q$  for all  $j = 1, \dots, k - 1$ .

It is easy to see that, if each split automaton  $\mathcal{A}_j$  accepts some word  $w_j$ , we find that  $w_1 \dots w_k$  is accepted by  $\mathcal{A}$ . Likewise, any word accepted by  $\mathcal{A}$  is also accepted in this sense by some split of  $\mathcal{A}$ . Since the combination of any split in general accepts less words than  $\mathcal{A}$ , splitting an NFA usually involves some don’t-know nondeterminism.

The intuition underlying this split is that each role NFA  $\mathcal{A}(R)$  encodes possible chains of roles that suffice to establish role  $R$ . Clearly, one such chain must be found for every query atom  $R(n, m)$ . But the proof graph already imposes a basic structure that defines how elements  $n$  and  $m$  can be connected, and any match with  $R$  must be distributed along the paths of the proof graph. This is implemented by the above split.

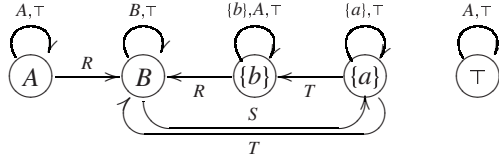
Finally, Step F again verifies the earlier choices of the algorithm by comparing the (logically deducible) role chains given by the edge NFA with the role chains that the split NFA require to exist for establishing a match. The case distinction reflects the different intention of edges leading to individual or variable nodes. For edges leading to a variable node, only a single generating role path exists in the canonical model, and *all* split automata must match one such path (line 27). For edges leading to nominal nodes, all of accepted paths exist in every model. Hence line 24 implements pairwise comparisons of each split NFA with the edge NFA. Concrete implementations for the checks of lines 24 and 27 are discussed in Section 6.

We conclude this section with a small example. Let  $KB$  be the knowledge base consisting of the following axioms:

$$A \sqsubseteq \exists R.B \quad B \sqsubseteq \exists S.\{a\} \quad T \circ R \sqsubseteq T \quad \{a\} \sqsubseteq \exists T.\{b\} \quad \{b\} \sqsubseteq A$$

with concept names  $A$  and  $B$ , role names  $R, S$  and  $T$ , and individuals  $a, b$ . Consider the query  $\{S(x, y), T(y, x)\}$ .

In Step A, the algorithm replaces  $y$  by  $a$  to obtain the query  $\{S(x, a), T(a, x)\}$ . The proof graph built in Step B has nodes  $N = \{a, b, x\}$  with  $L(a) = \{a\}, L(b) = \{b\}$  and  $L(x) = B$ . Edges are constructed between pairs of elements  $(a, b), (b, a), (x, a), (x, b)$ , and  $(b, x)$  (i.e.  $b$  generates  $x$ ). The constructed edge NFA are distinguished only by their start and end states (as rule (CR6) of Table 2 is not used), and have the following structure:



Step C succeeds since every edge automaton accepts some word, and Step D is omitted since no concept atoms appear in the query. The only nontrivial role NFA is  $\mathcal{A}(T)$  which accepts any word that starts with  $T$  followed by an arbitrary number of  $R$ . Due to the presence of the query atom  $T(a, x)$  this NFA must be split along the path from  $a$  over  $b$  to  $x$ , and there is only one split into two NFA that accept nonempty languages. Hence  $\mathcal{A}(T(a, x), a, b)$  accepts the single word  $T$ , and  $\mathcal{A}(T(a, x), b, x)$  accepts any sequence of  $R$ . The only other “split” NFA  $\mathcal{A}(S(x, a), x, a)$  is directly given by  $\mathcal{A}(S)$ , the NFA accepting only the word  $S$ . Finally in Step F the three existing split automata are compared to the corresponding edge NFA.  $\mathcal{A}(T(a, x), a, b)$  and  $E(a, b)$  accept a common word  $T$ ,  $\mathcal{A}(T(a, x), b, x)$  and  $E(b, x)$  accept a common word  $R$ , and  $\mathcal{A}(S(x, a), x, a)$  and  $E(x, a)$  accept a common word  $S$ . Hence the query is accepted.

## 5 Correctness of the Algorithm

**Proposition 2.** *Consider a regular consistent  $\mathcal{EL}^{++}$  knowledge base  $KB$  and a conjunctive query  $q$ . If the algorithm of Section 4 accepts  $q$ , then indeed  $KB \models q$ .*

*Proof.* We use the notation from Section 4 to denote structures computed by the algorithm. When terminating successfully, the algorithm has computed the following:

- A proof graph  $(N, L, E)$ ,
- For each role atom  $R(n, m) \in q$ , a  $k$ -split  $\mathcal{A}(R(n, m), n_0, n_1), \dots, \mathcal{A}(R(n, m), n_{k-1}, n_k)$  of the NFA  $\mathcal{A}(R)$ , where  $k$  is the length of the shortest path from  $n$  to  $m$  in  $(N, L, E)$ .

In the following, let  $\mathcal{I}$  be some model of  $KB$ . To show  $KB \models q$ , we need to provide a mapping  $\pi$  as in Section 2 for  $\mathcal{I}$ . Since  $\mathcal{I}$  is arbitrary, this shows the entailment of  $q$ . We can derive  $\pi$  from the proof graph, and then show its correctness based on the conditions checked by the algorithm.

In Step A, the algorithm replaces variables by individual names or by other variables. This is no problem: whenever a query  $q'$  is obtained from  $q$  by uniformly replacing a variable  $x \in \text{Var}(q)$  by an individual  $a \in \mathbf{I}$  (or variable  $y \in \text{Var}(q)$ ), we have that  $KB \models q'$  implies  $KB \models q$ . Indeed, any mapping  $\pi'$  for  $q'$  can be extended to a suitable mapping  $\pi$  for  $q$  by setting  $\pi(x) := a^{\mathcal{I}}$  ( $\pi(x) := y^{\mathcal{I}}$ ). Thus we can assume w.l.o.g. that all variables  $x \in \text{Var}(q)$  also occur as nodes in the proof graph, i.e.  $x \in N$ .

In Step F, the algorithm checks non-emptiness of the intersection languages of the NFA  $E(n, m)$ , and one/all split NFA  $\mathcal{A}(F, n, m)$ , for each  $n, m \in N$  with  $E(n, m)$  defined. Thus for any pair  $n \in N, m \in \text{Var}(q)$ , there is some word  $w$  accepted by *all* of the given NFA. Choose one such word  $w(n, m)$ . By the definition of the split NFA,  $w(n, m)$  is a word over  $\mathbf{R}$ , and we can assume this to be the case even when no split NFA (but just the single edge automaton) are considered for a given edge.  $E(n, m)$  is of the form  $\mathcal{A}(L(n), L(m))$  (Definition 2) for the selected labels  $L(n)$  and  $L(m)$  of the proof graph.

Now by Theorem 2 the construction of Definition 2 and the fact that  $KB$  is consistent, it is easy to see that  $E(n, m)$  accepts the word  $w(n, m) = R_1 \dots R_l$  iff  $KB \models L(n) \sqsubseteq \exists R_1 \dots \exists R_l L(m)$ . We employ this fact to inductively construct a mapping  $\pi$ .

In Step B the algorithm has defined labels  $L(x)$  for all  $x \in \text{Var}(q)$ , and we will retrace this process to construct  $\pi$ . We claim that the following construction ensures that, whenever a node  $n \in N$  is reachable,  $\pi(n)$  has been assigned a unique value such that  $\pi(n) \in L(n)^I$ . For starting the induction, set  $\pi(a) := a^I$  for each  $a \in \mathbf{I}$  (which is necessarily reachable and clearly satisfies  $\pi(a) \in L(a)^I = \{a\}^I$ ). Now assume that in one step the algorithm selected some  $x \in \text{Var}(q)$  that was not reachable yet, and node  $n \in N$  which is reachable. As noted above,  $KB \models L(n) \sqsubseteq \exists R_1 \dots \exists R_l L(x)$  where  $w(n, x) = R_1 \dots R_l$ , and hence there is an element  $e \in L(x)^I$  such that  $(\pi(n), e) \in R_1^I \circ \dots \circ R_l^I$  (where  $\circ$  denotes forward composition of binary relations). Pick one such  $e$  and set  $\pi(x) := e$ . It is easy to see that the claim of the induction is satisfied.

In Step D it has been verified that  $L(n) \sqsubseteq C$  holds for each  $C(n) \in q$  (using standard polynomial time reasoning for  $\mathcal{EL}^{++}$ ), so we find  $\pi(n) \in C^I$ . It remains to show that a similar claim holds for all binary query atoms. Thus consider some role atom  $R(n, m) \in q$ , and let  $n = n_0, \dots, n_k = m$  denote the shortest path in the proof graph used to split the role automaton. So far, we have defined  $w(n_i, n_{i+1})$  only for cases where  $n_{i+1} \in \text{Var}(q)$ . By a slight overloading of notation, we now let  $w(n_i, n_{i+1})$  for  $n_{i+1} \in \mathbf{I}$  denote some word accepted by the intersection of  $E(n_i, n_{i+1})$  and the specific split automaton  $\mathcal{A}(R(n, m), n_i, n_{i+1})$ , which must exist as the algorithms must have verified non-emptiness of the intersection language. Assuming that  $w(n_i, n_{i+1}) = S_1 \dots S_l$ , we note that this still entails  $KB \models L(n_1) \sqsubseteq \exists S_1 \dots \exists S_l L(n_{i+1})$ . Since  $n_{i+1} \in \mathbf{I}$ , this actually shows that  $(\pi(n_i), \pi(n_{i+1})) \in S_1^I \circ \dots \circ S_l^I$ .

The word  $w = w(n_0, n_1) \dots w(n_{k-1}, n_k)$  is accepted by  $\mathcal{A}(R)$ , which is clear from the construction in Definition 3 as the parts  $w(n_i, n_{i+1})$  are accepted by the respective split automata. Assume that  $w = R_1 \dots R_k$ . We conclude  $(\pi(n), \pi(m)) \in R_1^I \circ \dots \circ R_k^I$  from the construction of  $\pi$  and the above observations for the case of edges connecting to individual elements. Thus by Proposition 4 we have  $(\pi(n), \pi(m)) \in R^I$  as required.  $\square$

It remains to show that the algorithm is also complete. This is done by demonstrating that there are suitable nondeterministic choices that enable the algorithm to accept a query whenever it is entailed. To guide those choices, we use the canonical model  $\mathcal{I}$  introduced in Section 3.

**Proposition 3.** *Consider a regular consistent  $\mathcal{EL}^{++}$  knowledge base  $KB$  and a conjunctive query  $q$ . If  $KB \models q$ , then there is a sequence of nondeterministic choices for the algorithm of Section 4 such that it accepts  $q$ .*

*Proof.* Consider the canonical model  $\mathcal{I}$  as constructed above. Since  $KB \models q$  and  $\mathcal{I} \models KB$ , there is some mapping  $\pi$  such that  $\mathcal{I}, \pi \models q$ . We will use  $\pi$  to guide the algorithm.

In Step A, a variable  $x \in \text{Var}(q)$  is replaced by  $n \in \text{Var}(q) \cup \mathbf{I}$  whenever  $\pi(x) = \pi(n)$ . For Step B, we choose the labelling  $L$  of the proof graph by setting  $L(e) := \iota(\pi(e))$ . As argued in the proof of Lemma 4,  $\delta \in C^I$  iff  $KB \models \iota(\delta) \sqsubseteq C$ , and hence we conclude that  $\pi(e) \in C^I$  implies that  $KB \models L(e) \sqsubseteq C$  for all  $e \in \mathbf{I} \cup \text{Var}(q)$ . Thus all unary atoms of  $q$  are accepted by the algorithm.



Now in each step of the generation of the edges  $E$  of the proof graph, the algorithm needs to pick some (unreachable)  $x \in \text{Var}(q)$  and some reachable node  $n$ . We will utilise the properties established in Section 3. By Property 1, there is a unique generating chain for each  $\pi(x)$  where  $x$  is not reachable within the proof graph yet. Moreover, since the chain of Property 1 is unique and shortest, it is also acyclic. Hence there is some unreachable  $x$  such that  $\pi(x)$  is not generated by any element of the form  $\pi(y)$  with  $y$  unreachable. Pick one such element  $x$ . Finally select one element  $n \in \mathbf{I} \cup \text{Var}(q)$  such that  $\pi(n)$  generates  $\pi(x)$ , and such that there is no element  $m$  for which  $\pi(m)$  generates  $\pi(x)$  and  $\pi(n)$  generates  $\pi(m)$ . Construct an edge  $E(m, x)$ .

Now for any elements  $n$  and  $m$  of the query, with  $m \in \text{Var}(q)$  and  $E(n, m)$  defined, the automaton  $E(n, m)$  accepts a non-empty language. This is seen by combining Property 2 with Theorem 2, where the second case of the theorem is excluded since  $KB$  is consistent. The algorithm's checks in Step C thus succeed.

The algorithm now has completed the proof graph construction, and the selection of split automata is required next. For all query atoms  $R(n, m)$ , we find that  $(\pi(n), \pi(m)) \in R^I$ , and thus we can apply Property 3 to obtain a respective chain of elements and role names, which we denote as  $\delta_0 \dots \delta_k$  and  $R_0 \dots R_{k-1}$  in the remainder of this proof.

Let  $j > 0$  denote the largest index of  $\delta_0 \dots \delta_k$ , such that  $\delta_j$  is of the form  $\pi(e_1)$  for some  $e_1 \in \mathbf{I}$ , if any such element exists. Otherwise, let  $j > 0$  denote the smallest index such that  $\delta_i$  is of the form  $\pi(e_1)$  for any  $e_1 \in \text{Var}(q)$ . We claim that there is a connection between  $n$  and  $e_1$  in the proof graph. Clearly, this is true if  $e_1 \in \mathbf{I}$  since these edges were constructed explicitly. Otherwise, Property 1 and our choice of  $e_1$  imply that an edge from  $n$  to  $e_1$  was constructed by the algorithm. Starting by  $\delta_{j+1}$ , find all elements  $\delta_i$  of the form  $\pi(e)$ ,  $e \in \text{Var}(q)$ , and label them consecutively as  $e_2, \dots, e_l$ . Note that this sequence can be empty, in which case we define  $l := 1$ . Obviously,  $e_l = m$ . We claim that  $n = e_0 \dots e_l = m$  is the shortest path from  $n$  to  $m$  within the proof graph. We already showed the connection between  $n = e_0$  and  $e_1$ . The connections between  $e_i$  and  $e_{i+1}$  are also obvious, since each  $e_i$  generates  $e_{i+1}$  by definition. Since the latter path is also the only path from  $e_i$  to  $e_l$ , the overall path is clearly the shortest connection.

The algorithm now splits  $\mathcal{A}(R)$  along the path  $n = e_0 \dots e_l = m$ . For each  $e_i$ , there is an index  $j(i)$  such that  $\delta_{j(i)} = \pi(e_i)$ . Hence, for each pair  $(e_i, e_{i+1})$ , there is a corresponding sequence of roles  $R_{j(i)+1} \dots R_{j(i+1)}$  which we denote by  $r_i$  ( $i = 0, \dots, l-1$ ), and the concatenation of those sequences yields the original  $R_0 \dots R_{k-1}$ . By Proposition 1 and Property 3, the automaton  $\mathcal{A}(R)$  accepts the word  $R_0 \dots R_{k-1}$ . To split the automaton, we consider one accepting run and define  $q_i$  to be the state of the automaton after reading the partial sequence  $r_i$ , for each  $i = 0, \dots, l-1$ . The states  $q_i$  are now used to construct the split automata  $\mathcal{A}_i$ , and it is easy to see that those automata accept the sequences  $r_i$ .

Now assume that all required split automata have been constructed in this way. Consider any pair of query elements  $e, e' \in \mathbf{I} \cup \text{Var}(q)$  for which a split automaton  $\mathcal{A}(F, e, e')$  was constructed using a partial sequence of roles  $r$ . We claim that the edge automaton  $E(e, e')$  accepts  $r$ . Indeed, this follows from Property 2 and Theorem 2. This shows non-emptiness of intersections between any single split automaton and the corresponding edge automaton in the proof graph, and thus suffices for the case where  $e' \in \mathbf{I}$ .

Finally, consider the case that  $e' \in \text{Var}(q)$ , and assume that two split automata  $\mathcal{A}(F, e, e')$  and  $\mathcal{A}(F', e, e')$  have been constructed for the given pair, based on two



partial role sequences  $r$  and  $r'$ . We claim that  $r = r'$ . Indeed, this is obvious from the fact that  $r$  and  $r'$  both correspond to the unique generating sequence of roles for the elements  $e$  and  $e'$ , which is part of the sequence constructed for Property 1. This shows that  $r$  is accepted both by  $\mathcal{A}(F, e, e')$  and by  $\mathcal{A}(F', e, e')$ . We conclude that the intersection of all split automata and the edge automaton  $E(e, e')$  is again non-empty.

The algorithm thus has completed all checks successfully and accepts the query.  $\square$

## 6 Complexity of Query Answering for $\mathcal{EL}^{++}$

Finally, we harvest a number of complexity results from the algorithm of Section 4.

**Lemma 2.** *Given a regular  $\mathcal{EL}^{++}$  knowledge base  $KB$  and a conjunctive query  $q$ , the entailment problem  $KB \models q$  is hard for NP w.r.t. the size of  $q$ , hard for P w.r.t. the size of the ABox of  $KB$ , and hard for PSPACE w.r.t. to the combined problem size, even when restricting to simple RBoxes.*

The hardness proofs in [11] apply known hardness results for the data-complexity of instance checking in fragments of  $\mathcal{EL}$  [12], evaluation of single Datalog clauses (NP-complete, [13]), and emptiness of NFA intersection languages (PSPACE-complete, [14]).

We remark that the above results are quite generic, and can be established for many other DLs. Especially, NP-hardness w.r.t. knowledge base size can be shown for any logic that admits an ABox, whereas PSPACE hardness of the combined problem follows whenever the DL additionally admits role composition and existential role restrictions.

**Lemma 3.** *Given a regular  $\mathcal{EL}^{++}$  knowledge base  $KB$  and a conjunctive query  $q$ , the entailment problem  $KB \models q$  can be decided in P w.r.t. the size of the knowledge base, in NP w.r.t. the size of the query, and in PSPACE w.r.t. the combined problem size, given that RBoxes are simple whenever  $KB$  is not fixed.*

*Proof.* First consider Step A of Table 3. It clearly can be performed nondeterministically in polynomial time. If the query is fixed, the number of choices is polynomially bounded, and so the whole step is executable in polynomial time.

Similar observations hold for Step B. Concept names and automata for edges can be assigned in polynomial time by a nondeterministic algorithm (and thus in polynomial space). If the query has fixed size, available choices again are polynomial in the size of  $KB$ : the assignment of labels  $L$  admits at most  $|C|^{|\text{Var}(q)|}$  different choices, and for each such choice, there are at most  $n^2$  possible proof graphs, where  $n$  is the number of nodes in the graph. Since  $n$  and  $|\text{Var}(q)|$  are considered fixed, this yields a polynomial bound.

Further nondeterminism occurs in Step E. But if the query is fixed, each of the polynomially many proof graphs dictates a number of splits that is bounded by the size of the query  $m$ . Since splitting a role NFA into  $k$  parts corresponds to selecting  $k$  (not necessarily distinct) states from this NFA, there are  $|Q_{\mathcal{A}}|^k$  different ways of splitting  $\mathcal{A}$ . Since  $k$  is bounded by the size of the query  $m$ , we obtain an upper bound  $|Q|^{m^m}$  that is still polynomial in the size of  $KB$  (which, by our assumptions on simplicity of the RBox, determines the maximum number of states  $|Q|$  of some role NFA). If the query is not fixed, splitting can be done nondeterministically in polynomial time.

Now for Step F, the algorithm essentially has to check the emptiness of intersection languages of various automata. Given NFA  $\mathcal{A}_1, \dots, \mathcal{A}_l$ , this check can be done in two ways, each being worst-case optimal for different side conditions of the algorithm:

- (1) Initialise state variables  $q_1, \dots, q_l$  as being the initial states of the involved NFA. Then nondeterministically select one input symbol and one transition for this symbol in each of the considered NFA, and update the states  $q_j$  accordingly. The algorithm is successful if at some stage each  $q_j$  is a final state of the automaton  $\mathcal{A}_j$ . The algorithm runs in  $\text{NPSpace}$  w.r.t. the accumulated size of the input automata.
- (2) Iteratively compute the intersection NFA for  $\mathcal{A}_j = (Q_j, \Sigma, \delta_j, i_j, F_j)$  and  $\mathcal{A}_{j+1} = (Q_{j+1}, \Sigma, \delta_{j+1}, i_{j+1}, F_{j+1})$ . This intersection is the NFA  $(Q_j \times Q_{j+1}, \Sigma, \delta, (i_j, i_{j+1}), F_j \times F_{j+1})$ , with  $\delta((a_1, b_1), (a_2, b_2)) = \delta(a_1, a_2) \cap \delta(b_1, b_2)$ . The algorithm is successful if the intersection is non-empty. This construction is polynomial if the number of the input NFA is known to be bounded.

Method (1) establishes a general (nondeterministic) polynomial space procedure, which by Savitch's Theorem is also in  $\text{PSPACE}$ . Method (2) can be used to establish tighter bounds in special cases: each intersection might cause a quadratic increase of the size of the NFA, but the number of required intersections is bounded if  $KB$  or  $q$  are fixed. Indeed, if the query is fixed, the number of required intersections is bounded by the overall number of role atoms in the query. If the knowledge base is fixed, the number of interesting intersections is bounded by the number of split NFA that can be produced from role NFA constructed from the RBox, which is bounded by a fixed value. In both cases, checking intersections can be done deterministically in polynomial time.  $\square$

The below table summarises some common complexity measures for the case of conjunctive query answering in regular  $\mathcal{EL}^{++}$  knowledge bases. Whenever the RBox is variable, we assume that it is simple. It should be remarked that TBox and ABox could always be considered variable without increasing any of the given complexities.

	Variable parts:				Complexity
	Query	RBox	TBox	ABox	
Combined complexity	×	×	×	×	$\text{PSPACE-complete}$
Query complexity	×				$\text{NP-complete}$
Schema complexity		×	×	×	$\text{P-complete}$
Data complexity				×	$\text{P-complete}$

## 7 Conclusion

We have proposed a novel algorithm for answering conjunctive queries in  $\mathcal{EL}^{++}$  knowledge bases, which is worst-case optimal under various assumptions. To the best of our knowledge, this also constitutes the first inference procedure for conjunctive queries in a DL that supports complex role inclusions (including composition of roles) in the sense of OWL 1.1. Showing undecidability of conjunctive queries for unrestricted  $\mathcal{EL}^{++}$ , we illustrated that the combination of role atoms in queries and complex role inclusion axioms can indeed make reasoning significantly more difficult.

A compact automata-based representation of role chains *and* (parts of) models allowed us to establish polynomial bounds for inferencing in various cases, thus identifying querying scenarios that are still tractable for  $\mathcal{EL}^{++}$ . Conjunctive queries inherently introduce some nondeterminism, but automata can conveniently represent sets of possible solutions instead of considering each of them separately. We therefore believe that the presented algorithm can be a basis for actual implementations that introduce additional heuristics to ameliorate nondeterminism.

## References

1. Chandra, A.K., Merlin, P.M.: Optimal implementation of conjunctive queries in relational data bases. In: Hopcroft, J.E., Friedman, E.P., Harrison, M.A. (eds.) STOC 1977. Proc. 9th annual ACM Symposium on Theory of Computing, pp. 77–90. ACM Press, New York (1977)
2. Horrocks, I., Sattler, U., Tessaris, S., Tobies, S.: How to decide query containment under constraints using a description logic. In: Parigot, M., Voronkov, A. (eds.) LPAR 2000. LNCS (LNAI), vol. 1955, pp. 326–343. Springer, Heidelberg (2000)
3. Hustadt, U., Motik, B., Sattler, U.: A decomposition rule for decision procedures by resolution-based calculi. In: Baader, F., Voronkov, A. (eds.) LPAR 2004. LNCS (LNAI), vol. 3452, pp. 21–35. Springer, Heidelberg (2005)
4. Ortiz, M.M., Calvanese, D., Eiter, T.: Data complexity of answering unions of conjunctive queries in SHIQ. In: Proc. 2006 Description Logic Workshop (DL 2006), CEUR Electronic Workshop Proceedings (2006), <http://ceur-ws.org/>
5. Ortiz, M.M., Calvanese, D., Eiter, T.: Characterizing data complexity for conjunctive query answering in expressive description logics. In: AAAI 2006. Proc. 21st Nat. Conf. on Artificial Intelligence (2006)
6. Glimm, B., Horrocks, I., Lutz, C., Sattler, U.: Conjunctive query answering for the description logic SHIQ. In: IJCAI 2007. Proc. 21st Int. Joint Conf. on Artificial Intelligence, Hyderabad, India (2007), available at <http://www.ijcai.org/papers07/contents.php>
7. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: IJCAI 2005. Proc. 19th Int. Joint Conf. on Artificial Intelligence, Morgan-Kaufmann Publishers, Edinburgh, UK (2005)
8. Krisnadhi, A., Lutz, C.: Data complexity in the EL family of DLs. In: Proc. DL 2007. CEUR Electronic Workshop Proceedings (2007)
9. Horrocks, I., Sattler, U.: Decidability of SHIQ with complex role inclusion axioms. In: Gottlob, G., Walsh, T. (eds.) IJCAI 2003. Proc. 18th Int. Joint Conf. on Artificial Intelligence, Acapulco, Mexico, pp. 343–348. Morgan-Kaufmann Publishers, San Francisco (2003)
10. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible SROIQ. In: KR 2006. Proc. 10th Int. Conf. on Principles of Knowledge Representation and Reasoning, pp. 57–67. AAAI Press, Stanford, California, USA (2006)
11. Krötzsch, M., Rudolph, S.: Conjunctive queries for EL with role composition. Technical report, Universität Karlsruhe (TH), Germany (2007), available at [http://www.aifb.uni-karlsruhe.de/Publikationen/showPublikation?publ\\_id=1463](http://www.aifb.uni-karlsruhe.de/Publikationen/showPublikation?publ_id=1463)
12. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Data complexity of query answering in description logics. In: KR 2006. Proc. 10th Int. Conf. on Principles of Knowledge Representation and Reasoning, pp. 260–270 (2006)
13. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity and expressive power of logic programming. ACM Computing Surveys 33, 374–425 (2001)
14. Kozen, D.: Lower bounds for natural proof systems. In: Proc. 18th Symp. on the Foundations of Computer Science, pp. 254–266 (1977)

# Continuous RDF Query Processing over DHTs<sup>\*</sup>

Erietta Liarou<sup>1</sup>, Stratos Idreos<sup>1</sup>, and Manolis Koubarakis<sup>2</sup>

<sup>1</sup> CWI, Amsterdam, The Netherlands

<sup>2</sup> National and Kapodistrian University of Athens, Athens, Greece

**Abstract.** We study the continuous evaluation of conjunctive triple pattern queries over RDF data stored in distributed hash tables. In a continuous query scenario network nodes subscribe with long-standing queries and receive answers whenever RDF triples satisfying their queries are published. We present two novel query processing algorithms for this scenario and analyze their properties formally. Our performance goal is to have algorithms that scale to large amounts of RDF data, distribute the storage and query processing load evenly and incur as little network traffic as possible. We discuss the various performance tradeoffs that occur through a detailed experimental evaluation of the proposed algorithms.

## 1 Introduction

Continuous querying of RDF data has been studied only by a few Semantic Web researchers, although it is an important component of many Semantic Web applications [6,5,15,14,4,13]. In a continuous query scenario, users post long-standing queries expressing their interests and are notified whenever new data matching their queries become available (e.g., “Notify me when a new article by John Smith becomes available”). Prominent examples of applications based on continuous querying of RDF data are the following: alert systems for Web resource management systems with resources annotated by RDF metadata (e.g., e-learning systems like ELENA [19], semantic blogging systems [10], RSS feeds etc.), notification mechanisms for Semantic Grid software [7], data coordination in heterogeneous P2P databases [2] based on RDF and so on.

In this work, we study the evaluation of *continuous conjunctive queries* composed of *triple patterns* over RDF data stored in *distributed hash tables (DHTs)*. Conjunctions of triple patterns are core constructs of some RDF query languages (e.g., RDQL [18] and SPARQL [16]) and used implicitly in all others (e.g., in the generalized path expressions of RQL [11]). DHTs are an important class of P2P networks that offer distributed hash table functionality, and allow one to develop scalable, robust and fault-tolerant distributed applications [1]. DHTs have recently been used for instantaneous querying of RDF data by [4,12] and continuous querying of RDF data by [4,13]. Unfortunately, both papers [4,13] use a rather limited query language allowing only *atomic* triple patterns or conjunctions of triple patterns with the *same* variable or constant subject and possibly different *constant* predicates (the so-called conjunctive multi-predicate queries). Thus, the problem of evaluating arbitrary continuous conjunctive queries composed of triple patterns is left open by [4,13].

---

<sup>\*</sup> E. Liarou and M. Koubarakis were supported in part by the European project Ontogrid.

In this paper, we solve this problem by presenting two novel algorithms (the *continuous query chain* algorithm - CQC and the *continuous spread-by-value* algorithm - CSBV) for the continuous evaluation of conjunctive triple pattern queries on top of DHTs. The core ideas of CQC and CSBV come from the algorithms QC and SBV of [12] that solve the corresponding query processing problem for one-time queries. The first contribution of the present paper is to show that the main ideas of QC and SBV are powerful enough to be applicable in a continuous query setting, and to develop the required machinery of the new algorithms CQC and CSBV. The second contribution is a detailed experimental evaluation of CQC and CSBV. We focus on two critical parameters in a distributed setting; load distribution and network traffic. Both algorithms are designed in such a way that they try to involve as many nodes as possible in the query processing procedure, while taking into account the amount of traffic they create. This involves a tradeoff and we clearly demonstrate that one algorithm can sacrifice some nice load distribution properties to keep network traffic at a lower level or vice versa.

For a continuous conjunctive query of  $k$  triple patterns, we may need  $k$  different triples and each triple may participate in more than one answer sets for a given query. These triples may arrive asynchronously. This means that when a triple  $t$  arrives and we detect that it can be used to generate an answer for a query  $q$ , we should “remember” this information to use it in the future, when the rest of the triples that are necessary to create an answer for  $q$  (together with  $t$ ) arrive. To avoid recomputing queries each time new matching data arrive, we keep a distributed state of already received triples. We achieve this by careful assignment of intermediate results to the proper nodes (where future matching data might arrive) and by rewriting queries into ones with fewer conjuncts that reflect the fact that certain triples have arrived. Another key point is that our algorithms split the responsibility of handling events at the triple pattern level. When a query  $q$  is inserted in the network, it is not assigned to a single node. Instead, different nodes are responsible for different triple patterns of  $q$  which allows for better load distribution.

We present a large number of experimental results. For example, we study the effect of varying the number of indexed queries. The larger the number of continuous queries indexed in the network waiting for data, the harder it is to find answers each time new data arrives. We show that our algorithms manage to spread the extra load created to a large part of the network by maintaining their nice load distribution and limited network traffic creation properties. Another important parameter we study is the rate of triple publication in the network. If RDF triples arrive more frequently, we also have to perform query processing operations more often. We demonstrate that our algorithms manage to keep distributing the extra load without creating heavy network traffic and without overloading a restricted set of nodes. We also show that if more resources are available they can nicely be used by our algorithms, e.g., when more nodes connect in the network, they will be assigned parts of the current query processing operations to remove load from existing nodes.

The rest of the paper is organized as follows. Section 2 presents our assumption regarding the system architecture, the data and query model. Sections 3 and 4 present the alternative indexing and query processing algorithms. In Section 5, we present a detailed experimental evaluation and comparison under various parameters. Finally, Section 6 presents related work and Section 7 concludes the paper.

## 2 System Model and Data Model

In this section, we introduce the system and data model. Essentially, we extend the models of [12] to deal with the continuous query scenario of this paper.

**System model.** We assume an overlay network where all nodes are equal, they run the same software and they have the same rights and responsibilities. Nodes are organized according to the Chord DHT protocol [20] and are assumed to have synchronized clocks. In practice, nodes will run a protocol such as NTP and achieve accuracies within few milliseconds [3]. Each node can insert data and pose continuous queries. Each node  $n$  has a unique key, denoted by  $key(n)$  and each data item (RDF triple or query) has a key denoted by  $key(i)$ . Chord uses consistent hashing to map keys to identifiers. Each node and item is assigned an  $m$ -bit identifier using function  $Hash(k)$  (e.g., SHA-1, MD5) that returns the  $m$ -bit identifier of key  $k$ . Identifiers are ordered in an *identifier circle (ring)* modulo  $2^m$  i.e., from 0 to  $2^m - 1$ . Key  $k$  is assigned to the first node which is equal or follows  $Hash(k)$  clockwise in the identifier space. This node is called the *successor* node of identifier  $Hash(k)$  and is denoted by  $Successor(Hash(k))$ . A query for locating the node responsible for a key  $k$  can be done in  $O(\log N)$  steps with high probability [20], where  $N$  is the number of nodes in the network.

Our algorithms use the API defined in [9,8] that provides two functionalities not given by the standard DHT protocols: (i) send the same message to multiple nodes and (ii) send  $d$  messages to  $b$  nodes where each node receives one or more messages. Function  $send(msg, id)$ , where  $msg$  is a message and  $id$  is an identifier, delivers  $msg$  from any node to node  $Successor(id)$  in  $O(\log N)$  hops. Function  $multiSend(msg, I)$ , where  $I$  is a set of  $d > 1$  identifiers  $I_1, \dots, I_d$ , delivers  $msg$  to nodes  $n_1, n_2, \dots, n_d$  such that  $n_j = Successor(I_j)$ , where  $1 < j \leq d$ . This happens in  $O(d \log N)$  hops. This function is also used as  $multiSend(M, I)$ , where  $M$  is a set of  $d$  messages and  $I$  is a set of  $d$  identifiers ( $b$  distinct ones). If more than one messages, say  $j$ , have the same receiver node  $n$ , then the identifier of  $n$  will appear  $j$  times in the set  $I$ . For each  $I_j$ , message  $M_j$  is delivered to  $Successor(I_j)$  in  $O(b \log N)$  hops.

**Data model.** In the application scenarios we target, each network node is able to describe in RDF the resources that it wants to make available to the rest of the network, by *publishing* metadata in the form of *RDF triples*. In addition, each node can *subscribe with continuous queries* that describe resources that this node wants to receive *answers* about. Different schemas can co-exist but we do not support schema mappings. Each node uses some of the available schemas for its descriptions and queries.

We will use the standard RDF concept of a triple. Let  $D$  be a countably infinite set of URIs and RDF literals. A triple is used to represent a statement about the application domain and is a formula of the form  $(subject, predicate, object)$ . The *subject* of a triple identifies the resource that the statement is about, the *predicate* identifies a property or a characteristic of the subject, while the *object* identifies the value of the property. The subject and predicate parts of a triple are URIs from  $D$ , while the object is a URI or a literal from  $D$ . For a triple  $t$ , we will use  $subj(t)$ ,  $pred(t)$  and  $obj(t)$  to denote the string value of the subject, the predicate and the object of  $t$  respectively.

As in RDQL [18], a *triple pattern* is an expression of the form  $(s, p, o)$  where  $s$  and  $p$  are URIs or variables, and  $o$  is a URI, a literal or a variable. A *conjunctive query*  $q$  is a



formula of the following form:  $?x_1, \dots, ?x_n : (s_1, p_1, o_1) \wedge (s_2, p_2, o_2) \wedge \dots \wedge (s_n, p_n, o_n)$  where  $?x_1, \dots, ?x_n$  are variables, each  $(s_i, p_i, o_i)$  is a triple pattern, and each variable  $?x_i$  appears in at least one triple pattern  $(s_i, p_i, o_i)$ . Variables will always start with the '?' character. Variables  $?x_1, \dots, ?x_n$  are called *answer variables* distinguishing them from other variables of the query. A query with a single conjunct is called *atomic*.

Let us define the concept of valuation (to talk about values satisfying a query). Let  $V$  be a finite set of variables. A *valuation*  $v$  over  $V$  is a total function  $v$  from  $V$  to the set  $D$ . In the natural way, we extend a valuation  $v$  to be identity on  $D$  and to map triple patterns  $(s_i, p_i, o_i)$  to triples and conjunctions of triple patterns to conjunctions of triples.

Each triple  $t$  has a time parameter called *publication time*, denoted by  $pubT(t)$ , that represents the time that the triple is inserted into the network. Each query  $q$  has a time parameter too, called *subscription time*, denoted by  $subscrT(q)$ . Each triple pattern  $q_i$  of a query  $q$  inherits the subscription time, i.e.,  $subscrT(q_i) = subscrT(q)$ . A triple  $t$  can satisfy/trigger a triple pattern of query  $q$  only if  $subscrT(q) \leq pubT(t)$ , i.e., only triples that are inserted after a continuous query was subscribed can participating in its satisfaction. Finally, each query  $q$  has a unique key, denoted as  $key(q)$ , that is created by concatenating an increasing number to the key of the node that posed  $q$ .

Let us now give the semantics of query answering in our continuous query processing setting. We first deal with instantaneous queries [12], and then use their semantics to define the concept of answer to a continuous query.

An *RDF database* is a set of triples. Let  $DB$  be an RDF database and  $q$  an *instantaneous* conjunctive query  $q_1 \wedge \dots \wedge q_n$  where each  $q_i$  is a triple pattern. The *answer* to  $q$  over database  $DB$  consists of all  $n$ -tuples  $(v(?x_1), \dots, v(?x_n))$  where  $v$  is a valuation over the set of variables of  $q$  and  $v(q_i) \in DB$  for each  $i = 1, \dots, n$ .

Let  $q$  be a *continuous* query submitted to the network at time  $T_0$  to be evaluated continuously for the interval  $[T_0, \infty]$ . Let  $t$  be a time instant in  $[T_0, \infty]$ , and  $DB_t$  the set of triples that have been published in the network during the interval  $[T_0, t]$ . The *answer* to query  $q$  at time  $t$ , denoted by  $ans(q, t)$ , is the bag union of the results of evaluating the instantaneous query  $q$  over  $DB_{t'}$  at every time instant  $T_0 \leq t' < t$ .

The above definition assumes bag semantics for query evaluation. This semantics is supported by the algorithms CQC and CSBV. Simple modifications to the algorithms are possible so that set semantics (i.e., duplicate elimination) is also supported.

Note also that the above definition defines the answer to a query at each time  $t$  after this query was submitted. In practice, continuous query processing algorithms such as CQC and CSBV will evaluate submitted queries incrementally i.e., triples in the answer will be made available to the querying node as soon as possible after they are generated.

### 3 The CQC Algorithm

Let us now describe our first algorithm, the *continuous query chain* algorithm (CQC). In the presentation of our algorithms, it will be useful to represent a conjunctive query  $q$  of the form  $q_1 \wedge \dots \wedge q_n$  in list notation i.e., as  $[q_1, \dots, q_n]$ .

**Indexing a query.** Assume a node  $n$  that wants to subscribe with a conjunctive query  $q = [q_1, \dots, q_k]$  with set of answer variables  $V$ . Node  $n$  indexes each triple pattern  $q_j$  to a different node  $n_j$ . Each node  $n_j$  is responsible for query processing regarding  $q_j$ , and

all nodes  $n_1, \dots, n_k$  will form the *query chain* of  $q$ . To determine the satisfaction of  $q$  for a given set of incoming triples, the nodes of a query chain have to collaborate by exchanging intermediate results.

Now let us see how a node indexes each triple pattern. For each triple pattern  $q_j$  of  $q$ ,  $n$  computes an identifier  $I_j$  using the parts of  $q_j$  that are constant. For example, assume a triple pattern  $q_j = (?s_j, p_j, ?o_j)$ . Then, the identifier for  $q_j$  is  $I_j = Hash(pred(q_j))$  since the predicate part is the only constant part of  $q_j$ . This identifier is used to locate the node  $n_j$  that will be responsible for  $q_j$ . In Chord terminology, this node will be the successor of the identifier  $I_j$ , namely  $n_j = Successor(I_j)$ . If a triple pattern has just one constant, this constant is used to compute the identifier of the node that will store the triple pattern. Otherwise, if the triple pattern has multiple constants,  $n$  will heuristically prefer to use first the subject, then the object and finally the predicate to determine the node that will evaluate  $q_j$ . Intuitively, there will be more *distinct* subject or object values than *distinct* predicates values in an instance of a given schema. Thus, our decision helps us to achieve a better distribution of the query processing load.

So, for the query  $q$ , we have  $k$  identifiers whose successors are the nodes that will participate in the query chain of  $q$ . Node  $n$  has to send to each one of these nodes a message with the appropriate information notifying them that from there on, each one of them will be responsible for one of the triple patterns of  $q$ . The exact procedure is as follows. For simplicity, assume that triple patterns are indexed in the order they appear in the query. Thus, the first node in the query chain is responsible for the first triple pattern in the query, the second node is responsible for the second triple pattern and so on. In Section 4 we revisit this issue. For each triple pattern  $q_1, \dots, q_k$ ,  $n$  creates a message  $IndexTriplePattern(q_j, V, key(q), I_{j+1}, First)$  to be delivered to nodes  $n_1, \dots, n_k$  respectively. Identifier  $I_{j+1}$  allows node  $n_j$  to be able to contact the next node in the query chain  $n_{j+1}$ . When the message is sent to the last node  $n_k$  in a query chain, this argument takes the value  $key(n)$  so that  $n_k$  will be able to deliver results back to the node  $n$  that submitted  $q$ . Parameter *First* is a Boolean that indicates whether  $n_j$  will be the first node in the query chain of  $q$  or not. After having created a collection of  $k$  messages (one for each triple pattern),  $n$  uses function  $multiSend()$  to deliver the messages. Thus,  $q$  is indexed in  $k * O(\log N)$  overlay hops, where  $N$  is the size of the network.

When a node  $n_j$  receives a message  $IndexTriplePattern()$ , it stores all its parameters in the local *query table* ( $QT$ ) and waits for triples to trigger the triple pattern.

**Indexing a new triple.** Let us now proceed with the next logical step in the sequence of events in a continuous query system. We have explained so far how a query is indexed. We will now see how an incoming triple is indexed. We have to make sure that a triple will meet all relevant triple patterns so that our algorithm will not miss any answers. Looking back to how a triple pattern is indexed, we see that we always use the constant parts of a triple pattern. Thus, we have to index a new triple in the same way. Therefore, a new triple  $t = (s, p, o)$  has to reach the successor nodes of identifiers  $I_1 = Hash(s)$ ,  $I_2 = Hash(p)$  and  $I_3 = Hash(o)$ . The node that inserts  $t$  will use the  $multiSend()$  function to index  $t$  to these 3 nodes in  $O(\log N)$  overlay hops. In the next paragraph we discuss how a node reacts upon receiving a new triple.



**Receiving a new triple.** Assume a node  $n_j$  that receives a new triple  $t$ .  $n_j$  has to determine if  $t$  is relevant to any already indexed queries so  $n_j$  searches its local  $QT$  for triple patterns that match  $t$ . Assume that a matching triple pattern  $q_j$  belonging to query  $q$  is found, i.e., there is a valuation  $v$  over the variables of  $q_j$  such that  $v(q_j) = t$ . According to the position/order of  $n_j$  in the query chain of  $q$ ,  $n_j$  now acts differently. We will distinguish between two cases: (a) when  $n_j$  is the first node in the query chain of  $q$  and (b) when  $n_j$  is any other node but the first one. For ease of presentation we point out now that in the second case, a node always stores the new triple in its *triple table* ( $TT$ ). Later on, we will come back to this case to explain the rest of the steps.

If  $n_j$  is the first node in the query chain of a query  $q$ ,  $n_j$  forwards valuation  $v$  to the next node  $n_{j+1}$  in the chain. Valuation  $v$  holds a partial answer to  $q$ . Thus, from here on, we will call such valuations *intermediate results*. In the implemented system,  $n_j$  creates the following message  $FwdValuation(v, key(q))$  that has to be delivered to  $n_{j+1} = Successor(Hash(I_{j+1}))$ . So, for all  $l$  queries in  $QT$  whose triple patterns have been triggered in  $n_j$  by  $t$ ,  $n_j$  will perform the operations we just described and use the  $multiSend()$  function to forward the various intermediate results to the appropriate nodes in query chains. This will cost  $l * O(\log N)$  overlay hops.

**Receiving intermediate results.** Let us now see how a node  $n_j$  reacts upon receiving an intermediate result i.e., a valuation  $w$ . First,  $n_j$  applies  $w$  to  $q_j$ , the triple pattern it is responsible for, to compute  $q'_j = w(q_j)$ . Then  $n_j$  tries to find if triples matching  $q'_j$  have already arrived. So,  $n_j$  searches its  $TT$  and for *each* triple  $t \in TT$  that matches  $q'_j$  (i.e., there is a valuation  $v$  over the variables of  $q'_j$  such that  $v(q'_j) = t$ ),  $n_j$  produces a new intermediate result, the valuation  $w' = w \cup v$ . Then,  $n_j$  forwards the new intermediate results to the next node  $n_{j+1}$  in the query chain of  $q$  in a single message using the  $Send()$  function with a cost of  $O(\log N)$  hops. In addition,  $n_j$  will store the intermediate result  $w$  locally in its *intermediate results table* ( $IRT$ ) to use it whenever new triples arrive. Node  $n_{j+1}$  that receives the set of new intermediate results will react in exactly the same way for each member of the set and so on. When the last node in the chain for a query  $q$  (i.e.,  $n_k$ ) receives a set of intermediate results, stored triples in  $TT$  are checked for satisfaction against each  $q'_k$ , and for each successful triple, an *answer* to the query  $q$  is generated using each valuation  $w'$  and is returned to the node that originally posed  $q$ .

Now we come back to finish the discussion on what happens when a node  $n_j$  receives a new triple  $t$  that triggers a triple pattern  $q_j$  and  $n_j$  is not the first node in the query chain of  $q$ . So far, we have only said that  $n_j$  will store  $t$  in its  $TT$ . In addition,  $n_j$  has to search its  $IRT$  table to see whether the evaluation of a query that has been suspended can now continue due to  $t$  that has just arrived. For each intermediate result  $w$  found in  $IRT$  that can be used to compute  $q'_j = w(q_j)$  that matches  $t$ ,  $n_j$  produces a new intermediate result (i.e., a valuation  $w' = w \cup v$  where  $t = v(q'_j)$ ) and forwards it to the next node  $n_{j+1}$  in the query chain of  $q$  with a  $FwdValuation()$  message.

**Example.** CQC is shown in operation in Figure [□](#). Each *event* represents an event in the network, i.e., the arrival of a new triple or query. Events are drawn from left to right which represents the chronological order in which these events have happened. In each event, the figure shows the steps of the algorithm that take place due to this event.

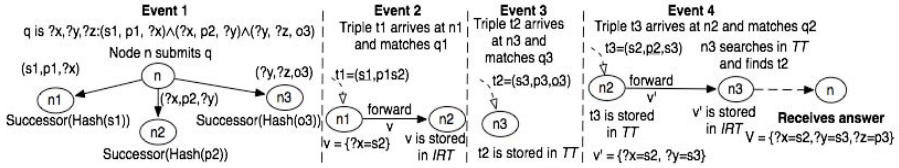


Fig. 1. The algorithm CQC in operation

For readability and ease of presentation in each event we draw only the nodes that do something due to this event, i.e., rewrite a query, search or store queries or triples etc.

### 4 The CSBV Algorithm

Let us now proceed with the description of our second algorithm, the *continuous spread-by-value* algorithm (CSBV). CSBV extends the ideas of CQC to achieve a better distribution of the query processing load. In CQC, a query chain for a query  $q$  is created at the time that  $q$  is submitted and leads to a query plan with a *fixed* number of participating nodes (one node per triple pattern in the absence of collisions in the DHT). Notice that whenever the first node  $n_1$  in the chain for query  $q = [q_1, \dots, q_k]$  creates a new intermediate result (i.e., a valuation  $v$ ), the triple pattern  $q_1$  corresponding to  $n_1$  is satisfied by  $v$  and  $q$  has been partially evaluated. The motivation for CSBV is that node  $n$  can now use valuation  $v$  to *rewrite*  $q$  into a new query with *fewer* conjuncts  $q' = [v(q_2), \dots, v(q_n)]$  and decide *on the fly* the next node of the network to undertake the query processing of  $q'$ . Because  $q'$  is conjunctive like  $q$ , its processing can proceed in a similar manner. The answer bag of  $q$  can then be computed by the union of the answer bags of queries  $q'$  combined with valuations  $v$ . In this way, a node  $n_i$  in the chain of CQC for a query  $q$  can be said to have multiple next nodes in CSBV depending on the triples that trigger  $q_i$ . Thus, the responsibility of evaluating the next triple pattern of  $q$  is distributed to multiple nodes (depending on values used) compared to just one in CQC, leading to a much better query processing load distribution. If we generalize this to all nodes participating in a query plan, it is easy to see that a query plan in CSBV does not have the shape of a chain (as in CQC) but rather that of a tree. We will describe CSBV by pointing to the different actions that are taken comparing with CQC in each step.

**Indexing a query.** In CQC, when a query  $q = [q_1, \dots, q_k]$  is inserted, we immediately create a query chain of  $k$  nodes. In CSBV, no query chain is created. Instead,  $q$  is indexed *only to one node* that will be responsible for one of the triple patterns of  $q$ . Thus, a query is indexed with only  $O(\log N)$  hops. For now assume that, as in the description of CQC, triple patterns are handled in the order they appear in the query i.e.,  $q_1$  is used to index  $q$  to node  $n_1$ .

In CSBV, we follow the same indexing heuristics as in CQC, when there is just one constant part in a triple pattern. But in case that there are multiple constants, we use the *combination* of *all* constant parts to index the query. For example, if  $q_j = (?s_j, p_j, o_j)$ , we have  $I_j = Hash(pred(q_j) + obj(q_j))$ . We use the operator  $+$  to denote the *concatenation* of string values. Multiple constants typically occur in triple patterns where

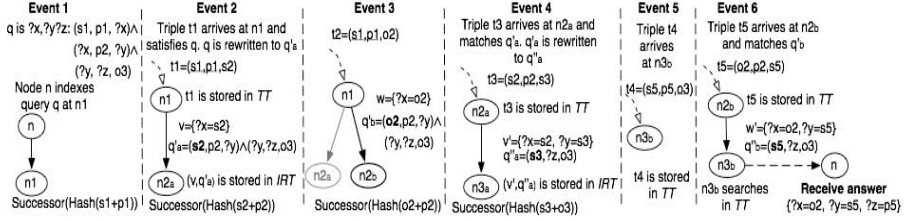


Fig. 2. The algorithm CSBV in operation

variables have been substituted by values of incoming triples (see discussion below in Paragraph “Receiving a new triple”). Using these combinations, a node in CSBV can direct intermediate results towards different branches of the distributed query plan tree (or dynamically create a new branch) depending on values used in incoming triples.

**Indexing a new triple.** As we discussed, CSBV uses the combination of constant parts in a triple pattern to index a query. Thus, in order not to miss possible answers, a new triple  $t = (s, p, o)$  has to reach the successor nodes of identifiers  $I_1 = Hash(s)$ ,  $I_2 = Hash(p)$ ,  $I_3 = Hash(o)$ ,  $I_4 = Hash(s + p)$ ,  $I_5 = Hash(s + o)$ ,  $I_6 = Hash(p + o)$  and  $I_7 = Hash(s + p + o)$ . Thus, a node  $n_1$  that inserts  $t$  will use the *multiSend()* function to index  $t$  to these 7 nodes in  $7 * O(\log N)$  overlay hops.

**Receiving a new triple.** As in CQC, when a node  $n_j$  receives a new triple  $t$ , first it has to find if  $t$  triggers any local query  $q$  (possibly in combination with some valuation  $v$ ). If it does,  $n_j$  rewrites  $q$  using  $t$  and  $v$ , and new intermediate results will be forwarded to the next node in the query chain. The critical difference with CQC, is how  $n_j$  decides who will be the next node  $n_{j+1}$ . In CQC this information is given to each node in the chain upon insertion of the original query where the whole chain is created at once. Thus, in CQC,  $n_j$  knows that  $n_{j+1}$  is always the same node no matter what the triple that arrived is. On the contrary, in CSBV, this is a dynamic procedure and node  $n_{j+1}$  can be a different node for different triples that arrive in  $n_j$ . Nodes in CSBV use the rewritten queries that they create to decide who the next node is.

Let us see an example. Consider the query  $q = [(s_1, p_1, ?x), (?x, p_2, ?y), (?y, p_3, o_3)]$  indexed at node  $n_1$ . If  $t_1 = (s_1, p_1, s_2)$  arrives, then the new rewritten query is  $q' = [(s_2, p_2, ?y), (?y, p_3, o_3)]$  and the valuation is  $v = \{?x = s_2\}$ . Now the intermediate result is the pair  $(v, q')$ . In CQC,  $n_2$  would be *Successor(Hash(p2))* since this has been decided upfront (using the second triple pattern of  $q$ ). However, CSBV uses  $q'$  to decide what the next node will be. It exploits the new value  $s_2$  in the first triple pattern of  $q'$  to decide that the next node is the *Successor(Hash(s2 + p2))*. Assume now that another triple  $t_2 = (s_1, p_1, s_3)$  arrives at  $n_1$ .  $n_1$  rewrites  $q$  again and the new rewritten query now is  $q'' = [(s_3, p_2, ?y), (?y, p_3, o_3)]$ , while the new valuation is  $w = \{?x = s_3\}$ . In CSBV, node  $n_1$  will forward the pair  $(w, q'')$  to a different node than before, namely to node *Successor(Hash(s3 + p2))*, whereas in CQC it would go again to *Successor(Hash(p2))*.

In CQC, nodes participating in a query plan for a query  $q$  have the knowledge that they are members of this plan since they receive the appropriate triple pattern to be

responsible for at the time that  $q$  is submitted. Thus, when a node receives a new triple  $t$ , it does not need to store it if no locally stored triple pattern matches  $t$ . In CSBV, nodes do not have such knowledge since they are becoming part of a query plan dynamically, i.e., a node is not able to know if there is a query indexed in the network that can be triggered by  $t$  in the future when other triples with appropriate values arrive. Thus, a node in CSBV *always* stores locally a new triple to guarantee completeness.

**Receiving intermediate results.** Let us now see how a node  $n_j$  reacts upon receiving an intermediate result  $(w, p)$  where  $w$  is a valuation and  $p = [p_1, \dots, p_m]$  is a conjunctive query. First,  $n_j$  tries to find if relevant triples have already arrived that can contribute to the satisfaction of  $p_1$ , and thus to the satisfaction of the original query  $q$  from which  $p$  has been produced after possibly multiple rewriting steps. For this reason,  $n_j$  searches its local table  $TT$  and for *each* triple  $t \in TT$  that matches  $p_1$  (i.e., there is a valuation  $v$  such that  $v(p_1) = t$ ),  $n_j$  produces a new intermediate result  $(v', p')$ . In this case,  $v'$  is the union of  $w$  with  $v$  and  $p'$  is  $[v'(p_2), \dots, v'(p_m)]$ . To decide which node  $x$  will receive the new intermediate result,  $n_j$  uses the first triple pattern in  $p'$  using combinations of constant parts whenever possible to compute the identifier that will lead to  $x$ . When all matching triples have been processed, a *set* of new intermediate results has been created each one to be delivered to a possibly different node. Then, function *multiSend()* is used to deliver each intermediate result to the appropriate node with a cost of  $z * O(\log N)$  hops, where  $z$  is the number of unique identifiers created while searching  $TT$ . In addition,  $n_j$  will store the intermediate result  $(w, p)$  locally in its *intermediate results table (IRT)* to use it whenever new triples arrive.

Each node  $n_{j+1}$  that receives one of the new intermediate results will react in exactly the same way and so on. When a node  $n_k$  is responsible for the last triple pattern of a query and receives a set of intermediate results of the form  $(w, [q_k])$  then no intermediate results are generated. Instead, stored triples in  $TT$  are checked for satisfaction against  $q_k$ , and for each successful triple, an *answer* to the query  $q$  is generated using  $w$  and is returned to the node that originally posed  $q$ . In Figure 2 we show an example of CSBV.

**Optimizing network traffic.** To further optimize network traffic we use the *IP cache (IPC)* routing table we proposed in [12]. In both algorithms, each time a node  $n_j$  forwards intermediate results to the next node  $n_{j+1}$  in a query plan, we pay  $O(\log N)$  overlay hops. With the *IPC* after the first time that  $n_j$  has sent a message to  $n_{j+1}$ ,  $n_j$  keeps track of the IP address of  $n_{j+1}$  and uses it in the future when forwarding intermediate results through this query chain. Then,  $n_j$  can send a message to  $n_{j+1}$  in just 1 hop. Similarly, if a new triple initiates a new rewritten query  $q$  in the root of a query chain of  $k$  nodes, then  $q$  will need  $k * O(\log N)$  hops to reach the end of the query chain. With *IPC*, it will need just  $k$  hops. The cost for the maintenance of the *IPC* is only local.

**Optimizing a query chain.** It is important to find a *good order* of nodes in the query chain, so as to achieve the least possible network traffic and the least possible total load. A simple but powerful idea is to take into account the rate of published triples that trigger the triple patterns of the query (e.g., the rate in the last time window). We place *early* in a query chain nodes that are responsible for triple patterns that are triggered *very rarely*, while nodes that are responsible for triple patterns that are triggered *more frequently* are placed towards the end. An easy way to do this at the expense of

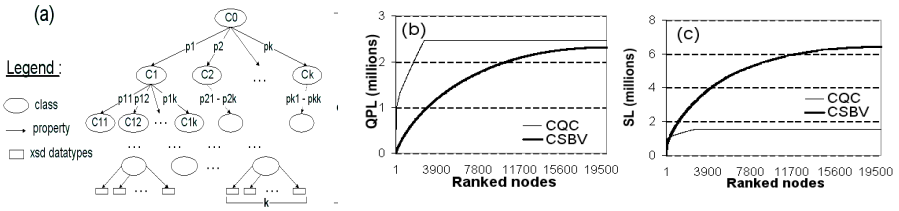


Fig. 3. Query processing and storage load

book-keeping by individual nodes and some extra messages upon query indexing is to ask all nodes that will participate in the query chain for the rate of incoming triples related to the triple pattern that they are going to be assigned. Thus, a node needs  $3k * O(\log N)$  messages to index a query instead of  $k * O(\log N)$ . For example, in CQC, when a node  $n$  wants to submit a query  $q$  of  $k$  triple patterns, it splits  $q$  to the triple pattern it consists of and assigns each  $q_i$  at a different node  $n_i$ . Before sending the triple patterns,  $n$  sends a message  $getRates(q_i)$  to each node  $n_i$ . When all answers return,  $n$  decides the order of the nodes having the most frequently accessed triple pattern towards the end of the query chain. Similarly in CSBV, when a node wants to submit a query  $q$ , it asks all possible candidate nodes based on the triple patterns of  $q$ . Only one node  $n_1$  is chosen to receive the query, the one responsible for the triple pattern with the lowest rate of incoming triples. From there on, when  $n_1$ , or any other node in the query chain of  $q$ , wants to forward intermediate results to a new node (i.e., create a new node in the query chain), it will follow the same procedure as to determine who will be this next node. These simple rules are sufficient to significantly improve network traffic in our setting at a minimal cost. The order of nodes can be periodically reevaluated and change (by migrating intermediate results through the nodes) in case the rates of incoming triples change. Due to space limitations we omit further analysis of these techniques.

### 5 Experiments

In this section, we experimentally evaluate our algorithms based on a Java implementation where we can run multiple nodes in one machine. We synthetically create a uniform workload as we did in [12]. We assume an RDFS schema of the form shown in Figure 3(a), i.e., a balanced tree of depth  $d$  and branching factor  $k$ . Each class has a set of  $k$  properties. Each property of a class  $C$  at depth  $l < d - 1$  ranges over another class at depth  $l + 1$ . Each class of depth  $d - 1$  has also  $k$  properties which have values that range over XSD datatypes. To create a triple, we first randomly choose a depth. Then, we randomly choose a class  $C_i$  among the classes of this depth. We randomly choose an instance of  $C_i$  to be  $subj(t)$ , a property  $p$  of  $C_i$  to be  $pred(t)$  and a value from the range of  $p$  to be  $obj(t)$ . If the range of  $p$  are instances of a class  $C_j$  that belongs to the next level, then  $obj(t)$  is a resource, otherwise it is a literal. For our experiments, we use an important type of conjunctive queries, namely *path queries* of the form:  $?x : (?x, p_1, ?o_1) \wedge (?o_1, p_2, ?o_2) \wedge \dots \wedge (?o_{n-1}, p_n, o_n)$ . To create a query, we randomly choose a property  $p_1$  of class  $C_0$ .  $p_1$  leads to a class  $C_1$  at the next level. Then we

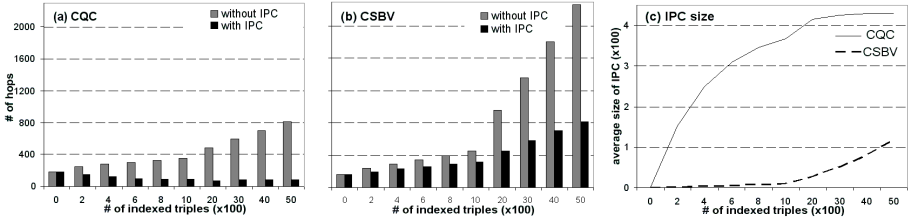


Fig. 4. Network traffic and *IPC* cost

randomly choose a property  $p_2$  of  $C_1$ . This is repeated until we create  $n$  triple patterns. For the last one, we randomly choose a value (literal) from the range of  $p_n$  as  $o_n$ .

Our experiments use a schema with  $d = 4$ . The number of instances of each class is 1000, the number of properties that each one has is  $k = 3$  while a literal can take up to 1000 different values. Finally, each query has 5 triple patterns.

**E1: Load distribution.** We define two types of load; the *query processing load* (QPL) and the *storage load* (SL). The QPL of node  $n$  is the sum of the number of triples that  $n$  receives to check against locally stored queries plus the number of intermediate results that arrive to  $n$  to be compared against its locally stored triples. The SL of a node is the sum of the number of triple patterns for which it is responsible, plus the number of triples and intermediate results that it stores.

We create a network of  $2 * 10^4$  nodes and insert  $10^5$  queries. Then, we insert  $6 * 10^5$  triples and measure the QPL and the SL of each node. In Figure 3(b), we show the QPL distribution. On the  $x$ -axis of this graph, nodes are ranked starting from the node with the highest load. The  $y$ -axis represents the *cumulative load*, i.e., each point  $(a, b)$  in the graph shows that  $b$  is the sum of the  $a$  most loaded nodes. CSBV achieves to distribute the QPL to a significantly higher portion of nodes, i.e., in CQC, there are only 2685 nodes (out of  $2 * 10^4$ ) participating in query processing, while in CSBV there are 19779 nodes. CSBV has a slightly lower total load than CQC since nodes in CSBV have more opportunities to group similar queries. Figure 3(c) shows the SL distribution. In CQC, the total SL is significantly less because in CSBV a new triple is indexed/stored four more times than in CQC, by using the combinations of the triple values. However, because of dynamic creation of query plans, this SL is nicely distributed. A higher total SL in the network is the price we pay for the better distribution of the QPL in CSBV.

**E2: Network traffic and *IPC* effect.** For this experiment, we create a network of  $2 * 10^4$  nodes and install  $10^5$  queries. Then, we train *IPCs* with a varying number of incoming triples, starting from 200 triples up to 5000. In each training phase, we insert 1000 triples and measure (a) the average number of overlay hops that are needed to *index* one triple and to *evaluate all* existing queries when using *IPCs*, (b) the size of *IPCs* at each node and (c) the same as (a) but this time we do not use *IPCs*. Finally, after each training phase, we measure how much it costs to insert a new triple.

Let us first see algorithm CQC, shown in Figure 4(a). The point 0 on the  $x$ -axis has the minimum cost, since it represents the cost to insert the first triple. There are



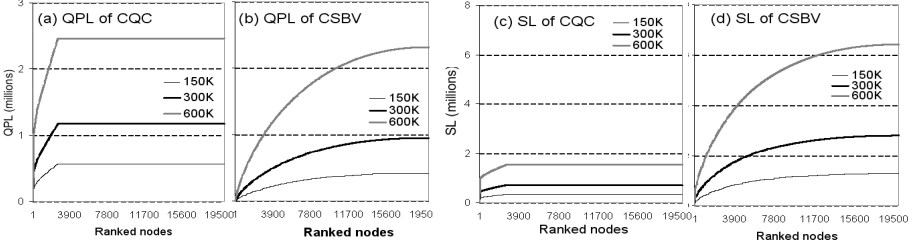


Fig. 5. QPL and SL when increasing the rate of incoming triples

no previous inserted triples so there are no partial results waiting for triples; therefore network traffic at this point is produced only because of the indexing of this triple to the network. *IPCs* are empty at this point so their use has no effect. However, in the next phases we observe a different behavior. Without *IPCs*, the network traffic required to insert a triple is increased, after each time we inserted a number of triples. This happens because each group of inserted triples results in the creation of new intermediate results. Thus, a next triple insertion has a higher probability to meet and trigger queries (and thus create more network traffic). This is why we see the gray bars in Figure 4(a) going higher after each phase. However, for the same reason, the black bars that represent the cost when using *IPCs* are going down. Triple insertions that trigger queries result in the forwarding of intermediate results. But when we use *IPCs*, these actions also fill the *IPCs* with IP addresses that can reduce subsequent forwarding actions. Thus, a next triple insertion will have a higher chance to cause forwarding of intermediate results with cost 1 instead of  $O(\log N)$  hops. For example, after 5000 triples, a triple insertion costs CQC 800 hops but with *IPCs* it costs only 60. Of course, this huge gain comes with a cost; in Figure 4(c) we show the average size (number of entries) of the *IPC* at each node. Naturally, this size is increased as more triples are inserted, but also observe that this is only a local storage cost at each node (there is no maintenance cost). Since even a small *IPC* size can significantly reduce network traffic (e.g., after 200 or 400 triples), we can allow each node to fill its *IPC* as long as it can handle its size.

In Figure 4(b), we show the network traffic cost for the CSBV algorithm. Results are explained with the same arguments as in CQC. The difference this time is that we see a much higher cost for CSBV both when using and when not using *IPCs*. This is due to the fact that nodes in CSBV cannot always group new intermediate results and send them with a single message to the next node in the query chain as it happens in CQC since usually there are more than one next nodes in CSBV. For the same reason, in Figure 4(c), we see that the *IPC* cost for nodes in CSBV is much smaller.

**E3: Effect of increasing the rate of incoming triples.** The base setting is a network of  $2 \times 10^4$  nodes with  $10^5$  queries and  $1.5 \times 10^5$  incoming triples. We present how the two algorithms are affected when the incoming triples become  $T = 3 \times 10^5$  and  $T = 6 \times 10^5$ .

In Figures 5(a) and (b), we show the cumulative query processing load distribution. In both algorithms the total load becomes higher, while the number of incoming triples is increasing since the already indexed queries are triggered by more triples. In CQC,

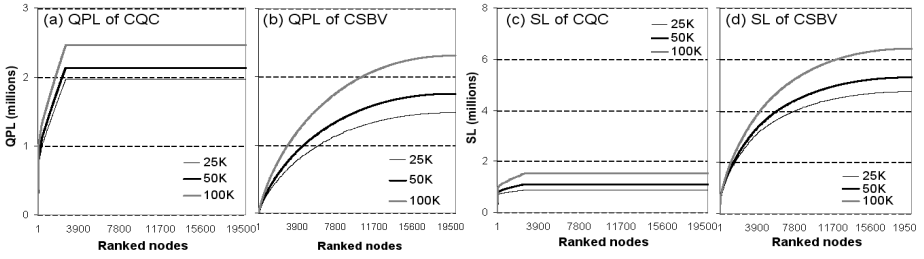


Fig. 6. QPL and SL when increasing the number of indexed queries

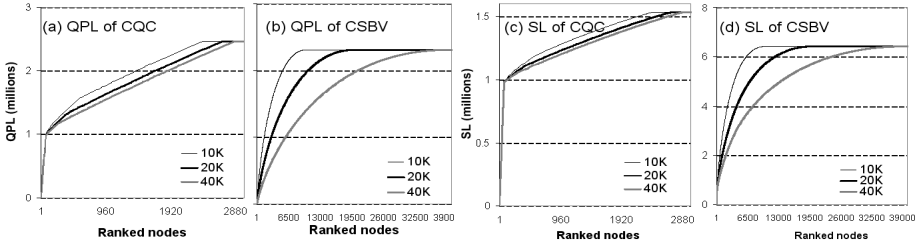


Fig. 7. QPL and SL when increasing the network size

the load distribution remains the same independently of the number of incoming triples; since a query chain is fixed at query submission, the responsible nodes remain always the same. Instead, in CSBV, query plans are formed while triples are arriving, thus, as the number of triples is increased, new responsible nodes are defined and incur part of the QPL. Thus, the load distribution in CSBV becomes more fair as the number of triples increases. Notice, also that CQC reaches a higher total load. This is due to the creation time of the query chains. In CQC, the nodes that are responsible for the submitted queries are determined initially, so when triples are inserted they have to check for satisfied triple patterns or forwarded intermediate valuations. On the other hand, in CSBV, the nodes that incur the QPL, start when the appropriate triples that trigger the corresponding triple patterns are inserted, so they do not work in vain.

In Figures 5(c) and (d), we present the SL distribution. Naturally, in both algorithms, the total load increases with the number of inserted triples since each incoming triple is stored and creates intermediate results. As expected, the load in CSBV is higher since it indexes a triple to four more nodes than in CQC. Also, in CQC, a node stores a triple  $t$  only if it is responsible for a triple pattern that is triggered by  $t$  while in CSBV, a node always stores a triple it receives. Finally, CSBV uses a combination of the constant parts of a triple to index it, and thus it achieves a better load distribution than CQC.

**E4: Effect of increasing the number of queries.** The base setting for this experiment is a network of  $2 \times 10^4$  nodes with  $2.5 \times 10^4$  queries and  $10^6$  triples. We increase the number of indexed queries to  $5 \times 10^4$  and  $10^5$ . In Figures 6 we see that the performance patterns remain the same while increasing queries, i.e., CSBV outperforms CQC in



terms of load distribution (both for QPL and SL) due to the dynamic query plans. This comes at the expense of a higher SL per node.

**E5: Effect of increasing the network size.** In a network of  $N = [10^4, 2 * 10^4, 4 * 10^4]$  nodes, we index  $10^5$  queries and then we insert  $10^6$  triples. Figure 7 shows that CQC is not able to exploit the new nodes while CSBV distributes the QPL and SL to almost as many nodes are available by dynamically creating query plans.

## 6 Related Work

This paper extends the one-time query processing algorithms QC and SBV of [12] to a continuous query processing environment. Historically, the study of continuous querying of RDF data in P2P networks was initiated in [6]. [6,5] deal with conjunctive multi-predicate queries (a subclass of the class of conjunctive triple pattern queries studied in this paper) and adopt HyperCup [17] as the underlying P2P infrastructure. Thus, their algorithms are not directly comparable with the ones of this paper.

[4] introduced publish/subscribe in the system RDF-Peers. The query language of RDF-Peers supports conjunctive multi-predicate queries, disjunctions of such conjunctions and range queries. RDF-Peers is built on top of an extension to Chord that supports order-preserving hashing so that range queries can be implemented easily. [4] concentrates mainly on one-time queries and the publish/subscribe subsystem of RDF-Peers is only briefly presented. Recently, [13] implemented and evaluated the algorithms QC and MQC for continuous conjunctive multi-predicate queries on top of Chord. QC is essentially the algorithm sketched (but not implemented or evaluated) in [4] while MQC is the algorithm that has motivated us to develop CSBV. It is not difficult to extend our algorithms to deal with disjunctions or range queries. For the former type of queries the extension is straightforward; for the latter, we could rely on an order-preserving hashing extension of Chord such as the one of [4].

Finally, [2][15][14] are some other recent papers on continuous querying of RDF data. In these papers, graph-based RDF queries are supported using centralized indices.

From the area of relational databases, [9] is the paper most closely related to our work. In [9], we have discussed algorithms for continuous two-way equi-join queries.

## 7 Conclusions and Future Work

We introduced and compared two novel algorithms for the evaluation of continuous conjunctive triple pattern queries over RDF data stored in a DHT. The algorithms manage to distribute the query processing load to a large part of the network and keep network traffic low. Our future work plans are to design techniques for handling skewed workload efficiently and to take into account physical network proximity. We also plan to support RDFS reasoning. Since RDFS triples can be handled similarly with RDF triples, the main challenge is how to support the inference of *new* RDFS triples using the RDFS inference rules in a compatible way with our query processing framework. This can be done in a forward chaining manner by extending CSBV. The algorithms of this paper have recently been implemented in our system Atlas available at <http://atlas.di.uoa.gr>. We are currently evaluating Atlas on PlanetLab.

## References

- [1] Aberer, K., et al.: The essence of P2P: A reference architecture for overlay networks. In: IEEE P2P 2005, IEEE Computer Society Press, Los Alamitos (2005)
- [2] Arenas, M., Kantere, V., Kementsietsidis, A., Kiringa, I., Miller, R.J., Mylopoulos, J.: The Hyperion Project: From Data Integration to Data Coordination. SIGMOD Record 32(3), 53–58 (2003)
- [3] Bawa, M., et al.: The Price of Validity in Dynamic Networks. In: SIGMOD 2004 (2004)
- [4] Cai, M., Frank, M.R., Yan, B., MacGregor, R.M.: A Subscribable Peer-to-Peer RDF Repository for Distributed Metadata Management. Journal of Web Semantics 2(2), 109–130 (2004)
- [5] Chirita, P., Idreos, S., Koubarakis, M., Nejdl, W.: Designing semantic publish/subscribe networks using super-peers. In: Semantic Web and Peer-to-Peer, Springer, Heidelberg (2006)
- [6] Chirita, P., Idreos, S., Koubarakis, M., Nejdl, W.: Publish/Subscribe for RDF-based P2P Networks. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, Springer, Heidelberg (2004)
- [7] Corcho, O., Alper, P., Kotsiopoulos, I., Missier, P., Bechhofer, S., Goble, C.: An overview of S-OGSA: a Reference Semantic Grid Architecture. Journal of Web Semantics 4(2), 102–115 (2006)
- [8] Idreos, S.: Distributed Evaluation of Continuous Equi-join Queries over Large Structured Overlay Networks. Master Thesis. Technical University of Crete (September 2005)
- [9] Idreos, S., Tryfonopoulos, C., Koubarakis, M.: Distributed Evaluation of Continuous Equi-join Queries over Large Structured Overlay Networks. In: ICDE 2006 (2006)
- [10] Karger, D.R., Quan, D.: What would it mean to blog on the semantic web? Journal of Web Semantics 3(2-3), 147–157 (2005)
- [11] Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D., Scholl, M.: RQL: A Declarative Query Language for RDF. In: WWW 2002 (2002)
- [12] Liarou, E., Idreos, S., Koubarakis, M.: Evaluating Conjunctive Triple Pattern Queries over Large Structured Overlay Networks. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
- [13] Liarou, E., Idreos, S., Koubarakis, M.: Publish-Subscribe with RDF Data over Large Structured Overlay Networks. In: DBISP2P 2005,
- [14] Liu, H., Petrovic, M., Jacobsen, H.-A.: Efficient and scalable filtering of graph-based metadata. Journal of Web Semantics 3(4), 294–310 (2005)
- [15] Petrovic, M., Liu, H., Jacobsen, H.-A.: G-ToPSS - fast filtering of graph-based metadata. In: WWW 2005 (2005)
- [16] Prud'hommeaux, E., Seaborn, A.: SPARQL Query Language for RDF, <http://www.w3.org/TR/rdf-sparql-query/>
- [17] Schlosser, M.T., et al.: HyperCuP - Hypercubes, Ontologies, and Efficient Search on P2P Networks. In: Moro, G., Koubarakis, M. (eds.) AP2PC 2002. LNCS (LNAI), vol. 2530, Springer, Heidelberg (2003)

- [18] Seaborne, A.: RDQL - A Query Language for RDF. W3C Member Submission (2004)
- [19] Simon, B., et al.: Smart Space for Learning: A Mediation Infrastructure for Learning Services. In: WWW 2003 (2003)
- [20] Stoica, I., et al.: Chord: A scalable peer-to-peer lookup service for internet applications. In: SIGCOMM 2001 (2001)
- [21] Wang, J., Jin, B., Li, J.: An Ontology-Based Publish/Subscribe System. In: Middleware 2004 (2004)

# Bringing Semantic Annotations to Web Services: OWL-S from the SAWSDL Perspective

David Martin<sup>1</sup>, Massimo Paolucci<sup>2</sup>, and Matthias Wagner<sup>2</sup>

<sup>1</sup> Artificial Intelligence Center, SRI International  
martin@ai.sri.com

<sup>2</sup> DoCoMo Communications Laboratories Europe GmbH  
{paolucci, wagner}@docomolab-euro.com

**Abstract.** Recently the World Wide Web Consortium (W3C) produced a standard set of “Semantic Annotations for WSDL and XML Schema” (SAWSDL). SAWSDL provides a standard means by which WSDL documents can be related to semantic descriptions, such as those provided by OWL-S (OWL for Services) and other Semantic Web services frameworks. We argue that the value of SAWSDL cannot be realized until its use is specified, and its benefits explained, in connection with a particular framework. This paper is an important first step toward meeting that need, with respect to OWL-S. We explain what OWL-S constructs are appropriate for use with the various SAWSDL annotations, and provide a rationale and guidelines for their use. In addition, we discuss some weaknesses of SAWSDL, and identify some ways in which OWL-S could evolve so as to integrate more smoothly with SAWSDL.

## 1 Introduction

The driving objective behind Web services technologies, such as the Web Services Description Language (WSDL) [2], is to provide reliable, ubiquitous software interoperability across platforms, across networks, and across organizations. Accordingly, the primary technical focus has been on standardizing and validating the syntax and mechanisms of message exchange, so as to support reliable, vendor-neutral communications between Web services and their users.

*Semantic Web services* technology aims to provide for richer semantic specifications of Web services, so as to enable fuller, more flexible automation of service provision and use, to support the construction of more powerful tools and methodologies, and to promote the use of semantically well-founded reasoning about services. The field, which got under way around 2001 [14], includes substantial bodies of work, such as the efforts around OWL for Services (OWL-S) [11], the Web Services Modeling Ontology (WSMO) [10], and METEOR-S [18].

Each of these efforts has sought to build out from, or integrate with, WSDL, rather than reinventing that part of the Web services picture. This has resulted in several distinct, ad hoc, styles of integration with WSDL. Recently, however, the World Wide Web Consortium (W3C) produced a standard set of “Semantic Annotations for WSDL and XML Schema” (SAWSDL) [6]. SAWSDL, based primarily on the earlier

work on WSDL-S [1], provides a standard means by which WSDL documents can be related to semantic descriptions, such as those provided by OWL-S and WSMO.

SAWSDL represents a conservative, incremental approach to introducing semantic characterization of Web services into mainstream Web service practices. Its objectives are modest. For example, it aims to provide semantic characterization of a service's input and output types, which can be useful in disambiguating those types in the context of simple forms of service discovery. But it does not attempt to provide a comprehensive framework to support more sophisticated approaches to discovery, composition, or any of the other service-related tasks that Semantic Web services research aims to automate.

SAWSDL does not specify a particular semantic framework within which to characterize the semantics of Web services. Rather, it defines a small set of WSDL extension attributes, which may be used to refer to constructs within *any* external semantic framework. SAWSDL is completely noncommittal regarding the choice of semantic framework. It is important to understand, however, that SAWSDL is of very little use unless there is an additional specification of conventions and guidelines for what can be referred to in a particular semantic framework, and what it means to do so. Consequently, such a specification is an essential and timely next step in bringing Semantic Web services research to fruition.

In addition to discovery, the SAWSDL specification mentions that SAWSDL annotations can be used during composition and invocation (Sections 1 and 2 of [6]). However, the specification says essentially nothing about how these tasks are to be supported or what degree of automation may be achieved. Indeed, there is very little that can be said, without reference to a particular semantic framework. (As we will see, the intended use of the schema mapping attributes *is* pretty clear, because they are more specialized than the `modelReference` attribute. However, even in this case, no guidance can be given regarding the details of how to specify a mapping, without reference to a particular semantic framework.)

In this paper, we provide guidelines regarding the use of OWL-S in conjunction with SAWSDL. These guidelines are provided *from the SAWSDL perspective*. That is, we do not try to explain everything that can be done with OWL-S in conjunction with WSDL. Rather, we simply explain what OWL-S constructs are appropriate for use with the various SAWSDL annotations. These explanations are provided with a view to supporting WSDL users and WSDL tool vendors in achieving the kinds of objectives that are associated with SAWSDL.

An analysis with similar objectives has previously been given for using WSMO with WSDL-S [8], and the use of SAWSDL is beginning to appear in WSMO tools such as [5].

Because of space limitations, it is not possible to give an adequate overview of WSDL or OWL-S. For introductory material on WSDL, the reader is referred to [2]. In Section 2, we give a brief characterization of OWL-S. Section 3 discusses the use of SAWSDL's `modelReference` attribute with OWL-S, and Section 4 discusses the use of SAWSDL's schema mapping attributes. In Section 5, we discuss some overarching issues and summarize our recommendations. Section 6 concludes.

## 2 OWL-S

As noted in [12], the principal high-level objectives of OWL-S are (i) to provide a general-purpose representational framework in which to describe Web Services; (ii) to support automation of service management and use by software agents; (iii) to build, in an integral fashion, on existing Web Service standards and existing Semantic Web standards; and (iv) to be comprehensive enough to support the entire life cycle of service tasks.

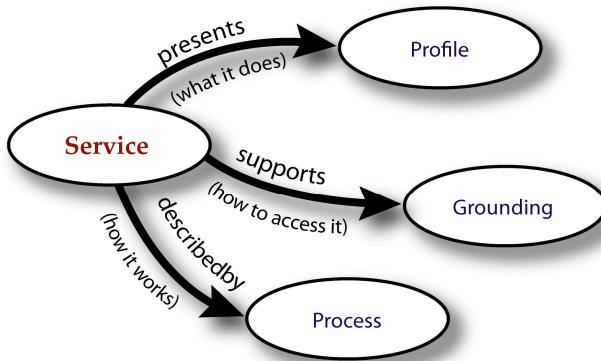


Fig. 1. Top level of the OWL-S service ontology

OWL-S (formerly known as DAML-S) is an OWL ontology [13] that includes three primary subontologies: the service profile, process model, and grounding. The service profile is used to describe *what the service does*; the process model is used to describe *how the service is used*; and the grounding is used to describe *how to interact with the service*. The service profile and process model are thought of as *abstract* characterizations of a service, whereas the grounding makes it possible to interact with a service by providing the necessary *concrete* details related to message format, transport protocol, and so on. Figure 1 shows the relationships between the top-level classes of the ontology. In this figure, an oval represents an OWL class, and an arc represents an OWL property. For example, the *presents* property represents a relationship that can hold between a *Service* and a *Profile*.<sup>1</sup>

Each service described using OWL-S is represented by an instance of the OWL class *Service*, which has properties that associate it with a process model (an instance of the class *Process*), one or more groundings (each an instance of the class *Grounding*), and optionally one or more profiles (each an instance of the class *Profile*). A process model provides the complete, canonical description of how to interact with the service at an abstract level, and the grounding supplies the details of how to embody those interactions in real messages to and from the service. Each

<sup>1</sup> For ease of exposition, Figure 1 presents a slight simplification. In particular, it omits an organizational layer of classes named *ServiceProfile*, *ServiceGrounding*, and *ServiceModel*.

service profile may be thought of as a summary of salient aspects of the process model plus additional information that is suitable for the purposes of advertising and selection. Several different types of grounding have been devised for OWL-S. The default and most widely used grounding, which is included in the OWL-S releases, employs WSDL. [11] discusses the grounding to WSDL 1.1, and [16] presents a proposal for a grounding that employs WSDL 2.0 and SAWSDL.

In this paper, we are concerned with the use of constructs of the profile and process model as referents of SAWSDL annotations. Because this paper adopts a perspective centered around WSDL and SAWSDL, there is no need to employ the OWL-S grounding as a source of referents. OWL-S's grounding reflects an OWL-S perspective; that is, it is motivated by use cases in which service processing, tools, and reasoning of various kinds are organized around OWL-S. For example, the OWL-S Virtual Machine [15] executes OWL-S process models. When an invocation of an external Web service is indicated in a process model, the Virtual Machine uses the grounding to arrange for the invocation of that Web service.

Here, by contrast, we do not assume that processing will be organized around OWL-S. While the guidelines given here are consistent with the OWL-S grounding, they are meant to support the use of semantics in a manner that builds incrementally on WSDL usage, tools, and environments, in keeping with the philosophy underlying SAWSDL. At the same time, we strive to be as general as possible, and to support a variety of service-related tasks in a variety of architectures.

### 3 Using the *modelReference* Annotation

SAWSDL introduces three new extension attributes for use in WSDL and XML Schema documents, and discusses some of their possible uses [6]. *modelReference* can be used in both WSDL and XML Schema documents. The schema mapping attributes, *liftingSchemaMapping* and *loweringSchemaMapping*, are intended for use only in XML Schema documents. The addition of these attributes requires no other changes to existing WSDL or XML Schema documents, or the manner in which they had been used previously. In this section we discuss how *modelReference* can be used with OWL-S.

The SAWSDL specification states that “A model reference may be used with every element within WSDL. However, SAWSDL defines its meaning only for *wsdl:interface*, *wsdl:operation*, *wsdl:fault*, *xs:element*, *xs:complexType*, *xs:simpleType* and *xs:attribute*”<sup>2</sup> [6]. Here, we discuss the OWL-S constructs that are appropriate as referents of *modelReference* in each of these settings. We begin with operations, interfaces, and faults. Then we turn to the XML Schema elements (those with the “*xs*” prefix). Following that we discuss possible uses of *modelReference* with input and output (message) elements (even though those uses are not defined by SAWSDL).

---

<sup>2</sup> It should be noted that the guidance given regarding the uses of *modelReference* for each of these elements has much more the flavor of *suggestions* than *definitions*. For example, the material on usage with interfaces mentions that *modelReference* can be used “to categorize them according to some model, specify behavioral aspects *or other semantic definitions* [emphasis added]”, and similarly for operations.

### 3.1 Operations

In WSDL, an operation represents “a simple interaction between the client and the service. Each operation specifies the types of messages that the service can send or receive as part of that operation. Each operation also specifies a message exchange pattern [MEP] that indicates the sequence in which the associated messages are to be transmitted between the parties” [2].

Conceptually, the atomic process of OWL-S corresponds very closely to WSDL’s operation, and this correspondence was one of the cornerstones of OWL-S’s grounding to WSDL 1.1 [11]. For example, an operation that takes a single input message, and outputs a single output message, exhibits the same behavior as an OWL-S atomic process with a single input and a single output. In many cases such as this it is straightforward to establish a mapping between the constituents of the operation and those of the atomic process. In these straightforward cases, the value of `modelReference` should be the URI of an atomic process. Then, as described in Sections 3.4 and 3.5, the payload of the messages can be mapped to the types of the atomic process’s inputs and outputs (or to the inputs and outputs themselves), using `modelReference` annotations of the relevant XML Schema (or WSDL) declarations.

However, there is a very important caveat regarding the mapping of an operation to an atomic process: it can work only for simple message exchange patterns. The atomic process is defined in terms of a (possibly empty) set of inputs (arriving simultaneously) followed by a (possibly empty) set of outputs (leaving simultaneously). If an MEP cannot be mapped into that simple sequence of events, then that MEP cannot be mapped onto the I/O of an atomic process. (One could imagine a partial mapping, where some messages were ignored, but we will not consider that possibility here.)

WSDL 2.0 provides eight predefined MEPs: In-Only, Robust In-Only, In-Out, In-Optional-Out, Out-Only, Robust Out-Only, Out-In, and Out-Optional-In [3]. Four of these MEPs – In-Only, In-Out, Out-Only, Robust Out-Only – can be mapped onto the I/O of an atomic process.

What about Out-In, Out-Optional-In, Robust-In-Only, In-Optional-Out, and other, more complex MEPs that cannot be mapped onto the I/O of an atomic process? In principle, they can be mapped onto composite processes. Indeed, for any MEP, it is possible to construct an OWL-S composite process that supports (and requires) the same pattern of inputs and outputs. Similarly, for any OWL-S composite process, it is possible to construct a message pattern that corresponds to the I/O behavior of that composite process.

However, there are some issues needing further attention, having to do with the mapping of the inputs and outputs, which we take up in Section 3.5.

Finally, let us also note that it may be useful in some situations to annotate an operation by referring to a *profile*. If one is primarily concerned with *categorizing* operations as to the functionality they provide, an OWL-S profile is more appropriate for that purpose than a process. Nevertheless, the process should be regarded as the most natural referent for an operation, for all the reasons given above. In the following subsection, we discuss another possible use of the OWL-S profile as a referent.



### 3.2 Interfaces

In WSDL, an interface is, in essence, a group of related operations. The WSDL specification is not specific about how these operations are related, except to say that they make up the abstract interface of a Web service: “A WSDL 2.0 interface defines the abstract interface of a Web service as a set of abstract operations, each operation representing a simple interaction between the client and the service.” A service, in turn, “specifies a single interface that the service will support, and a list of endpoint locations where that service can be accessed.”

It should be noted that there is a mismatch between WSDL’s notion of “service” and that of OWL-S. In OWL-S, the *Service* class is an organizational unit that packages up the information that describes a single process; that process is, in effect, the essence of the service. As noted above, OWL-S’s *Process* corresponds to WSDL’s operation. Hence, an OWL-S *Service* also corresponds best to WSDL’s operation, rather than WSDL’s service.

Indeed, OWL-S does not have a construct for grouping processes. Therefore, it does not at present have a construct that corresponds *directly* (structurally) to WSDL’s interface or WSDL’s service. This is an area under consideration for a future release of OWL-S.

Nevertheless, there are three possible ways in which an interface’s *modelReference* can meaningfully refer to an OWL-S construct (or constructs). The first of these is to be preferred, given the intent that is expressed in the SAWSDL specification for these annotations.

(1) The SAWSDL specification indicates a possible use of the interface *modelReference* for *categorization* purposes. It mentions, as an example, an interface annotation that refers to an “electronics” concept in some semantic model. (This example provides an extremely limited bit of information – that is, that the interface has something to do with electronics. No doubt one could do better, for example, by referencing a concept for “ElectronicsRepairService” or “ElectronicsForSale”.)

In fact, the OWL-S Profile is meant to be used for categorization. To do this, one takes advantage, in a very natural way, of OWL’s mechanisms for building a class hierarchy; that is, a hierarchy of subclasses of Profile. For example, one might have *RailTicketSales* as a subclass of *TravelTicketSales*, as a subclass of *TravelAgency*, which in turn is a subclass of Profile. To represent a specific rail ticketing service, one would create an instance (i.e., OWL individual) of the class *RailTicketSales*. A larger, more comprehensive class hierarchy of this kind can be used as the basis for a “yellow pages” registry of services. An instance of a profile class from such a hierarchy can serve as the referent of the *modelReference* annotation of an interface. If a particular instance is not available, the class itself can serve as the referent.

It should be noted that an instance of OWL-S profile normally is bundled with a process model and a grounding, but that is not required by OWL-S.

(2) Since *modelReference* always allows for a *list* of URIs, one can simply list all the URIs of the processes that correspond to the interface’s operations. This information, however, would be redundant with the *modelReference* annotations of the operations themselves, so that limits the value of this approach.

(3) In some cases it is reasonable to map an interface to a composite process. A composite process can be viewed as a grouping mechanism, because it specifies and

coordinates calls (in the form of Perform statements) to a number of atomic processes. In certain cases, it could make sense to regard this set of atomic processes that are called by a composite process as the correlate of a WSDL interface. However, this cannot be regarded as a general rule, because in general the relationship between the atomic processes called from a composite process is quite different from the relationship between the operations grouped into an interface.

### 3.3 Faults

OWL-S does not yet have a concept of *fault* (or *exception*) *per se*. However, OWL-S has the *conditional effect*, which can be used to capture the same intent. A conditional effect (of a process) simply states what effects will occur under a given condition. That condition can directly correspond to a fault, such as the “ItemUnavailable” fault example given in Section 3.3 of [6]. Thus, it makes sense for the `modelReference` of a fault to refer to a conditional effect.<sup>3</sup>

### 3.4 XML Schema Elements

By default, and in what is by far the most common usage, the content of a WSDL message is described using XML Schema. That is, XML Schema is used to define an *element*, which in turn is associated with a message of a WSDL operation. The element defines the syntax that is allowed for the content of the associated message. The XML Schema definitions can appear inline, in the *types* section of a WSDL document, or in a separate XML Schema document that gets imported by the WSDL document.

The SAWSDL charter [9] gives a motivational example in which an operation, having input and output messages “*amount* and *tax*, both of type *xs:double*, could have different meanings: calculation of tax on a product, calculation of income tax, etc.” The problem illustrated by this example, of course, is that a very general, ubiquitous I/O type like *xs:double* tells you very little about the functionality or usage associated with an operation using that type. Here, an annotation referring to a type (e.g., a *SalesTax* concept) defined in some semantic framework, such as OWL, can provide value by helping to discover operations that can meet a given set of requirements.

To support this kind of use case, SAWSDL allows for the annotation of any *xs:element*, *xs:complexType*, *xs:simpleType*, or *xs:attribute* definition. For our purposes, there is little difference between these four kinds of definitions<sup>4</sup>; in each case, a `modelReference` annotation will associate a semantically defined concept with the corresponding unit of structure in XML Schema. In general, it is straightforward to map from a unit of structure in XML Schema to an OWL concept – and there can be a good deal of flexibility in doing so. In many cases, an element (or complex or simple type) such as, for example, `PurchaseOrder`, will map naturally onto an OWL class with similar structure. In other cases, depending on the choices that have been

<sup>3</sup> Precisely speaking, a conditional effect is an instance of OWL-S’s *Result* class.

<sup>4</sup> It should be noted that, “in WSDL 2.0, all normal and fault message types must be defined as single elements at the topmost level (though of course each element may have any amount of substructure inside it)” **Error! Reference source not found.**

made in structuring the ontology, it could also be reasonable to map an element (or complex or simple type) onto an OWL individual. In the case of a complex type, the SAWSDL specification notes that it can be annotated in a top-down style, a bottom-up style, or a combination of the two. Thus, in many cases, a complex type could map very naturally onto an OWL class, and its nested types could map onto the types (ranges) of that class's properties, that is, assuming that the XML Schema type and the OWL class have a parallel structure. But SAWSDL does not assume a parallel structure; indeed, SAWSDL is explicitly noncommittal regarding the relationship between the high-level and the lower-level annotations within a complex type: "A complex type can be annotated at both the top and member level. These annotations are independent of each other" (Section 4.1.2 of [6]).

### 3.5 Input and Output Elements

As described above, SAWSDL defines the use of `modelReference` with several kinds of XML Schema declarations. This gives an effective means of mapping from XML Schema to OWL. That is, given an arbitrary unit of structure defined in XML Schema, SAWSDL allows you to associate it with any OWL entity (or with a list of OWL entities) that can be referenced by URI. (SAWSDL has nothing to say about mapping in the other direction, and that is also out of scope for this paper.)

However, it is important to recognize the limitations of this approach in the context of *services*. The inputs and outputs of services are carried in messages. In WSDL, messages are described using MEPs, and *input* and *output* elements associated with operations. But SAWSDL's XML Schema annotations deal only with *content*, and say nothing about inputs, outputs, or message exchange patterns (MEPs).

Why does this matter? After all, it is certainly true that, with SAWSDL's defined XML Schema annotations, the content of any input or output message (and any element of structure within that content) can be mapped to a semantic referent. The problem is simply that the XML Schema annotations are not adequate to provide full disambiguation of the semantics associated with inputs and outputs – at least not without forcing a cumbersome duplication of XML Schema declarations.

Consider, for example, company X's use of a WSDL document (developed before SAWSDL was available) that defines an XML element *PurchaseOrder*, and reuses that element as the input type of three different operations. Suppose one of those operations uses the *PurchaseOrder* element to carry information for a *new* purchase, whereas another operation uses that same *PurchaseOrder* element to carry information for a *modified* purchase, and yet another operation uses it for a purchase to be *cancelled*. Suppose, further, that company X develops an OWL ontology that has distinct classes for *NewPurchaseOrder*, *ModifiedPurchaseOrder*, and *CancelledPurchaseOrder*, and wants to use its ontology to annotate its existing services. In a scenario such as this, the service could not be properly annotated without defining *the same three distinctions* in XML Schema, as distinct elements. Having done this, `modelReference` could be used, in the element declarations, to refer to the three different OWL classes appropriately. To properly correlate the three new elements, and their annotations, with the operations, the input constructs within the operation definitions would also have to be modified to indicate which of the

elements is used with which operation. This is a workable solution, but at the cost of considerable effort in maintaining legacy services. Given all this required effort at capturing these distinctions in XML Schema, one might well wonder if the semantic annotations are adding any value.

Moreover, when complex MEPs are used, difficulties such as these can arise within the annotation of a *single* operation. If an MEP has more than one input message, a similar situation could arise, in which multiple input messages could carry content of the same XML Schema type, but it would be important to annotate them with different semantic referents.<sup>5</sup> SAWSDL's defined uses of `modelReference` (with operation, interface, fault, and XML Schema constructs) do not readily allow for this. For many purposes, this can be a serious limitation. Even discovery is not well supported anymore. Consider an operation A with an MEP that takes an input message with a semantic referent of X, followed by an input message, using the same XML Schema type, with a semantic referent of Y. Operation B also takes two input messages using the same XML Schema type, but with semantic referents of Y followed by X. If these two operations cannot be distinguished, discovery becomes much less effective.

This situation can be remedied by setting out some guidelines for the use of `modelReference` with WSDL's *input* and *output* (message) constructs. With the use of these constructs, the purchase order example above can be easily accommodated by adding `modelReferences` (pointing to *NewPurchaseOrder*, *ModifiedPurchaseOrder*, and *CancelledPurchaseOrder*) directly onto the WSDL input declarations of the three operations, as appropriate. Examples with complex MEPs can similarly be disambiguated.

Instead of using an OWL *class* as the referent of an input (or output) element's `modelReference`, it is also possible to use an OWL-S input (or output) construct, or a set of OWL-S input (or output) constructs. These OWL-S constructs should, of course, belong to the process that corresponds to the operation of the message. Indeed, this is a more natural mapping for WSDL input and output elements. Compared to the use of OWL classes as referents, there is no lost information, because each OWL-S input and output already includes a mention of the class that serves as the type of the input or output.

## 4 Using Schema Mapping Annotations

SAWSDL's schema mapping annotations, *liftingSchemaMapping* and *loweringSchemaMapping*, "are used to associate a schema type or element with a mapping to an ontology .... The value of the *liftingSchemaMapping* attribute is a set of zero or more URIs that reference mapping definitions. A mapping referenced by this attribute defines how an XML instance document conforming to the element or type defined in a schema is transformed to data that conforms to some semantic model" (Section 4.2 of [6]). Similarly, *loweringSchemaMapping* is used to reference a mapping from data expressed in a semantic model to data expressed in an XML document.

---

<sup>5</sup> Similar difficulties can arise with outputs, of course.

There is very little to say about the schema mapping annotations that is specific to OWL or OWL-S. These annotations are likely to be used in conjunction with XSLT primarily, but the SAWSDL specification does not require XSLT or any other particular mapping language. The schema mapping annotations are the only aspects of SAWSDL that are clearly intended for use at runtime (and only at runtime). It should be noted that the OWL-S 1.1 (and previous release) groundings have also made use of XSLT scripts in the same general manner. As explained in [16], the use of an XSLT (or similar) syntax-based transformation approach from OWL to XML is problematic, because there are generally a number of different ways that the same content can be serialized in OWL. It can be quite complicated to write an XSLT script that handles all the different variants.

The OWL-S 1.1 grounding adopted some measures to alleviate this problem. It allows for the use of precondition expressions to bind variables to values in the semantic model (typically values passed in as inputs), and it specifies that a runtime environment should pass these bindings into corresponding variables declared in XSLT. The extent to which this alleviates the problem will depend upon the OWL-S developer—specifically, on the manner in which he or she writes precondition expressions. Preconditions can be written in a variety of languages, including SWRL [7] and SPARQL [17]. In principle, it is possible to use preconditions to break down complex OWL individuals into primitive elements, thus avoiding the issue of handling multiple possible serializations.

## 5 Discussion

The SAWSDL specification leaves a great deal to the imagination, and it remains to be seen whether, and in what ways, it will come to be widely used. There is very little that can be done with SAWSDL that does not require additional conventions. This paper is meant to be a start toward a set of conventions for using SAWSDL with OWL-S as the source of annotation referents.

This need for additional conventions is perhaps most evident with respect to SAWSDL's schema mappings (lifting and lowering). More than any of SAWSDL's other annotations, it is quite clear that the schema mappings cannot stand alone. That is, additional specifications and machinery are needed for them to be useful. At a minimum, a tool will need to know where to get the semantic data that needs to be lowered, or where to deliver the result of a lifting operation.<sup>6</sup> Of course, a great many other details may need to be specified as well, depending on purpose and context. For example, as noted earlier, OWL-S relies on variable bindings to be propagated from preconditions (normally expressed in SPARQL) to XSLT. Propagating these bindings cannot be accommodated, much less specified, using a schema mapping annotation – because these annotations allow for nothing other than a reference to a mapping script, such as an XSLT script. The conventions for variable bindings from another framework cannot themselves be captured in XSLT; they require additional specification.

---

<sup>6</sup> With model references, at least, one can imagine getting some mileage simply by comparing their URIs.

This is an illustration of the inherent weakness of SAWSDL with respect to more ambitious use cases. These more ambitious scenarios lead one to the conclusion that additional steps beyond SAWSDL will be needed before long; that is, conventions for use with larger frameworks. This conclusion is the motivation for the full OWL-S grounding, discussed in [16], which builds on SAWSDL annotations.

It is important to note that implicit constraints will often be associated with the use of SAWSDL annotations, if they are to be used in a coherent fashion with a single semantic framework such as OWL-S. For example, SAWSDL annotations of XML Schema elements can be used independently of services; they can be used merely to establish correspondences between elements of XML Schema definitions and elements of OWL ontologies. However, in the context of a larger, semantically annotated, WSDL document, an implicit set of constraints is associated with these annotations of XML Schema. This is because of the way in which the XML Schema types are used with operations, on the WSDL side, and the corresponding OWL types are used with the inputs and outputs of atomic processes, on the OWL-S side. Once you have mapped an operation to an atomic process, you have also implicitly established a correspondence between the set of XML Schema types used as the I/O types of the operation, and the set of OWL classes used as the I/O types of the atomic processes. To maintain coherence, then, these types need to be used consistently on both sides, across all operations and processes. Constraints such as these could and should be checked by tools.

OWL-S needs to evolve to support faults in a more straightforward manner. It would also be helpful if OWL-S had an organizational construct that directly correlated to WSDL's notion of an interface as a collection of operations.

## 6 Conclusion

We have given a rationale and guidelines for the use of OWL-S constructs as the referents of SAWSDL annotations. As explained in the Introduction, this kind of coupling of SAWSDL with a particular semantic framework is an essential and timely next step in bringing Semantic Web services research to fruition. Here is a summary of our recommendations:

- The `modelReference` of a WSDL operation can refer to an OWL-S atomic or composite process. With simple MEPs, either an atomic process or a composite process can be used (assuming that the process supports a pattern of I/O that is equivalent to the MEP). With complex MEPs (as characterized in Section 3.1), only a composite process can be used.
- The `modelReference` of a WSDL interface should refer to an instance of an OWL-S profile class (i.e., Profile or a subclass of Profile). If a particular instance is not available, a profile class can serve as the referent.
- The `modelReference` of a WSDL fault should refer to a conditional effect of an OWL-S process – the process that corresponds to the operation for which the fault is declared.
- Model references in XML Schema should refer to OWL constructs, and can do so independently of OWL-S.

- In addition, model references on WSDL input and output elements should be used to relate those elements to inputs and outputs of an OWL-S process – the process that corresponds to the operation for which the input or output element is declared.
- Schema mapping (lifting and lowering) annotations can refer to XSLT scripts. However, the usefulness of these scripts in translating from OWL is limited, as discussed in Section 4.

This paper and these recommendations assume that the most complete possible mapping is desired from WSDL onto OWL-S. A complete annotation of a WSDL document implies a number of constraints on the relationships between the referents of the annotations. Some of these constraints, which are not made explicit in SAWSDL, have been discussed here. They can and should be checked by tools.

These recommendations are, intentionally, of maximum generality so as to be relevant to a wide variety of use cases, purposes, and environments, and are provided here with a view to supporting WSDL users and tool vendors who want to use SAWSDL as the basis for an incremental introduction of semantics into Web service usage. More detailed specifications can evolve from these recommendations to support different situations, tool designs, etc.

## Acknowledgements

Many thanks to Jacek Kopecky for his knowledgeable and insightful comments.

## References

- [1] Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.T., Sheth, A., Verma, K.: Web Service Semantics - WSDL-S. In: W3C Member Submission (November 7, 2005), <http://www.w3.org/Submission/WSDL-S/>
- [2] Booth, D., Canyang, K.: Web Services Description Language (WSDL) Version 2.0 Part 0: Primer, W3C Candidate Recommendation (March 27, 2006), <http://www.w3.org/TR/2006/CR-wsdl20-primer-20060327/>
- [3] Chinnici, R., Haas, H., Lewis, A., Moreau, J., Orchard, D., Weerawarana, S. (eds.): Web Services Description Language (WSDL) Version 2.0 Part 2: Adjuncts, W3C Candidate Recommendation (March 27, 2006), <http://www.w3.org/TR/2006/CR-wsdl20-adjuncts-20060327/>
- [4] Chinnici, R., Moreau, J., Ryman, A., Weerawarana, S.(eds.): Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, W3C Candidate Recommendation (March 27, 2006), <http://www.w3.org/TR/2006/CR-wsdl20-20060327/>
- [5] Dimitrov, M., Simov, A., Konstantinov, M., Cekov, L., Momtchev, V.: WSMO Studio Users Guide (May 11, 2007), available at <http://www.wsmostudio.org/doc/wsmo-studio-ug.pdf>
- [6] Farrell, J., Lausen, H. (eds.): Semantic Annotations for WSDL and XML Schema, W3C Candidate Recommendation (January 26, 2007), <http://www.w3.org/TR/2007/CR-sawsdl-20070126/>

- [7] Horrocks, I., Patel-Schneider, P.F., Boley, H., et al.: SWRL: A Semantic Web Rule Language Combining OWL and RuleML, W3C Member (submission, May 21, 2004), <http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/>
- [8] Kopecký, J.: Aligning WSMO and WSDL-S, WSMO Working Draft (August 5, 2005), <http://www.wsmo.org/TR/d30/v0.1/>
- [9] Kopecký, J., Chair, C., Bournez, E., Prud'hommeaux.: Team Contacts, Semantic Annotations for WSDL Working Group Charter (October 2005), <http://www.w3.org/2005/10/sa-ws-charter>
- [10] Lausen, H., Polleres, A., Roman, D.: Web Service Modeling Ontology (WSMO), W3C Member (submission 2005), <http://www.w3.org/Submission/WSMO/>
- [11] Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., Sycara, K., Semantic, O.W.L.-S.: Markup for Web Services, W3C Member (submission, November 2004), <http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/>
- [12] Martin, D., Burstein, M., McDermott, D., McIlraith, S., Paolucci, M., Sycara, K., McGuinness, D., Sirin, E., Srinivasan, N.: Bringing Semantics to Web Services with OWL-S. *World Wide Web* 10(3) (September 2007)
- [13] McGuinness, D.L., van Harmelen, F., Web, O.W.L.: Ontology Language Overview, W3C Recommendation (February 10, 2004), <http://www.w3.org/TR/2004/REC-owl-features-20040210/>
- [14] McIlraith, S., Son, T.C., Zeng, H.: Semantic Web Services. *IEEE Intelligent Systems* 16(2), 46–53 (2001)
- [15] Paolucci, M., Ankolekar, A., Srinivasan, N., et al.: The DAML-S Virtual Machine. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) *ISWC 2003*. LNCS, vol. 2870, pp. 335–350. Springer, Heidelberg (2003)
- [16] Paolucci, M., Wagner, M., Martin, D.: Grounding OWL-S in SAWSDL. In: *ICSOC 2007. Proceedings of the International Conference on Service Oriented Computing*, Vienna, Austria (2007)
- [17] Prud'hommeaux, E., Seaborne, A.: SPARQL: Query Language for RDF, W3C Working Draft (October 4, 2006), <http://www.w3.org/TR/2006/WD-rdf-sparql-query-20061004/>
- [18] Patil, A., Oundhakar, S., Sheth, A., Verma, K.: METEOR-S Web service Annotation Framework. In: *WWW 2004. Proceedings of the World Wide Web Conference* (2004), available at <http://lsdis.cs.uga.edu/projects/meteor-s/>



# Ontology-Based Controlled Natural Language Editor Using CFG with Lexical Dependency

Hyun Namgoong and Hong-Gee Kim

Biological Knowledge Engineering Lab, Seoul National University,  
28-22 Yeongeon Dong, Jongno Gu, Seoul 110-749, Korea  
{ngh, hgkim}@snu.ac.kr

**Abstract.** In recent years, CNL (Controlled Natural Language) has received much attention with regard to ontology-based knowledge acquisition systems. CNLs, as subsets of natural languages, can be useful for both humans and computers by eliminating ambiguity of natural languages. Our previous work, OntoPath [10], proposed to edit natural language-like narratives that are structured in RDF (Resource Description Framework) triples, using a domain-specific ontology as their language constituents. However, our previous work and other systems employing CFG for grammar definition have difficulties in enlarging the expression capacity. A newly developed editor, which we propose in this paper, permits grammar definitions through CFG-LD (Context-Free Grammar with Lexical Dependency) that includes sequential and semantic structures of the grammars. With CFG describing the sequential structure of grammar, lexical dependencies between sentence elements can be designated in the definition system. Through the defined grammars, the implemented editor guides users' narratives in more familiar expressions with a domain-specific ontology and translates the content into RDF triples.

**Keywords:** Controlled Natural Language, Context-Free Grammar, Lexical Dependency, Ontology, OntoPath, Look-Ahead Editor.

## 1 Introduction

CNLs, as subsets of natural languages, have recently received much attention with regard to ontology-based knowledge acquisition systems, for its ability to eliminate ambiguity of expressions in natural languages. Several studies were devoted to the use of CNL in ontology-related data processing such as ontology construction, query generation, and data annotation [3][4]. A CNL-based guided look-ahead editor might help users select proper words that meet his intended but vague notions without proper knowledge on the sentence structures. The statements controlled by predefined grammars, usually defined in CFG which is a computational notation of natural language structures, can be translated into ontology-referenced data and queries with precision [2][5].

Our previous work, OntoPath, assists editing in such an intelligent way that it recognizes the resource type of a description and offers users context-sensitive actions to perform on that description. A domain-specific ontology plays a role in collecting

language constituents, such as nouns and verbs, to be translated into RDF triples [9][10]. A lightweight look-ahead editor helps users, specifically medical experts, with guidance on choosing next words, using the approved grammars and semantic relations of entities from the ontology. Because most medical sentences have general recommended structures to ensure precise knowledge expression, CNL and a look-ahead guiding system can assume an important role in such an application.

However, our previous work and other systems have difficulties in enlarging the expression capacity, expanding the grammar, specifying patternized sentences, and adapting informal expressions such as Korean sentences with English words. These difficulties are attributed to the fact that the grammar definition system like CFG does not include semantic structures, but sequential structures of a sentence. These limitations need to be solved to deal with various sentences so that users can exploit more familiar expressions, and are enforced into using the patternized sentences.

A newly developed editor, which we propose in this paper, permits grammar definitions through CFG-LD that includes both sequential and semantic views on sentence structures. Using this grammar definition system, we can define grammars and the semantic structures of sentences to be used in our editor. The Grammar definitions include the structural descriptions of grammatical states to mention sequences of POS (Part-Of-Speech) with CFG. Designations of lexical dependency between sentence elements are also included. Using defined grammars, the implemented CNL editor enables us to get structure data from writer's narratives with 1) more sophisticated expressions, 2) patternized expressions, and 3) informal expressions consisting of multi language constituents.

We begin this paper with the description of related works on a CNL. An explanation of the representation of narratives using RDF triples is provided in section 2. The CFG-LD and its definition rules are discussed in section 3. In sections 4 and 5, we explain the architecture and implementation of the developed editor. Finally, we provide conclusions of this work in section 6.

## 2 Controlled Natural Language to Semantic Web Data

### 2.1 Controlled Natural Language

CNL was restricted subsets of natural languages on grammars and dictionaries to eliminate ambiguity and complexity of pure natural languages. Originally, the main purpose of controlled languages was to improve readability for human readers, particularly non-native speakers. An example is AECMA Simplified English that was created as a manual description language for aircraft maintenance guideline. Another advantage of CNL is to improve text processing capability of computers with removed complexity.

Many studies have been done to develop systems that transform written sentences into formal logical expressions. Some well-known examples are as follows: ACE (Attempto Controlled English) [6], CLCE (Common Logic Controlled English) [7], and PENG-D [2]. For an automatic translation from a discourse representation structure into a variant of first-order logic, ACE is defined as a controlled natural language in Attempto project. The ACE based sentences are translated into the

Semantic Web querying language PQL [8]. Other example of a controlled natural language is CLCE that has been developed by Sowa [9]. CLCE, as a formal language with an English-like syntax, is supplied with more expression than ACE in the sense that it supports ontology for sets, sequences, and integers, and also allows in-line declarations of words linked to relational databases. It supports the automated translation of written narratives to conceptual graph or other logical expressions [8]. PENG-D also proposed a computer-processable CNL to be translated into formal logical sentences decidable with an OWL (Web Ontology Language) language.

## 2.2 Representation of Narratives Using RDF Triples in OnthPath

In this subsection, we overview a translation from a sentence to RDF triples with a gross description narratives example of pathologic examination. The description language supported by OnthPath is compatible with a restricted form of RDF and RDF Schema. The system is designed to annotate the semantic metadata in RDF with the vocabularies that are already constrained by a given ontology in RDF Schema. The ontology then plays a role in guiding the generation of medical narratives as RDF documents. The narratives are validated with the syntactic and semantic rules of RDF Schema and are transformed into RDF documents.

An RDF triple statement consists of a specific resource, which is an individual primitive semantic element with a named property and value for that resource. The basic RDF model represents the named properties and property values. A property is a rule that provides the meaning of the expressions, which is specifying the way the thing should be constituted. A built ontology such as a schema, which is a vocabulary description language, provides mechanisms for describing groups of related resources and the relationships between these resources. Instead of defining a class in terms of the properties its instances may have, the ontology describes properties in terms of the resource classes to which they apply. This is the role of the *domain* and *range* mechanisms.

A specimen received contains a cyst which measures  $2 \times 1\text{cm}$ .

This example sentence can be translated as shown in the figure 1 when it is typed in the form of the predefined grammar through the guidance of the editor [9][10]. The instances about a real patient, ‘a specimen received’ and ‘a cyst’ are conceptualized as the instances of classes **Tissue** and **Cyst**, respectively. The properties, *contains* and *measures*, are also specified with their object values in the sentence.

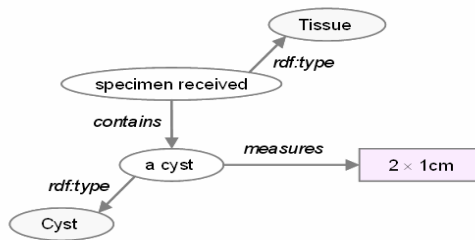


Fig. 1. RDF triples generated from an example sentence

This translation can be definitely performed on the example sentence written in a predefined grammar. However, if the user describes the sentence with another manner, it can not be successfully translated, because the translation system will assume restricted grammars and translation processes. To expand the grammars for enlarging expression capacity, we can add more grammars using CFG, but it is still not enough for this translation work, since the translation can be different from the composed structure and semantic dependency among the sentence elements.

### 3 Grammar Expression Through CFG-LD

In this chapter, we introduce CFG-LD, which is a grammar definition system for describing grammars with lexical dependences. As we have shown in the previous chapter, the translation between a simple English sentence and a RDF triple is possible through quite simple translation rule on the grammar. However, it is hard to deal with those sentences with different structures, and an annexed expression such as idioms (e.g., “there is something”) or patternized phrases appearing in the sentences. Other grammatical expressions following a different sequence of POS such as ‘subject-object-verb’ are also hardly handled through the original approach. Sequential and semantic structures of those sentences should be declared to enlarge the translation capacities.

Resolving the various structures of sentences can be possible through the previously developed CNL systems listed in the previous chapter. Their built-in sentence resolutions mainly relied on English are restricted in the informal expressions consisting of multi-language constituents, and it is also hard to gather well-defined CNL grammars written in every desired language. Therefore, in our CNL based editor, we employ slightly modified grammar expressions named CFG-LD. It notifies a lexical parser for both grammars and lexical dependencies, to let the parser or system know sequential and semantic structures of the grammars where the ontology provides language constituents and domain and range relations of them.

#### 3.1 Grammar Expression with Context-Free Grammar

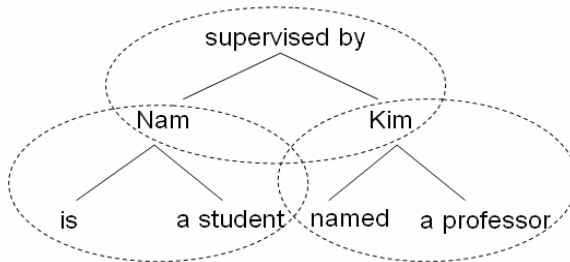
CFG is a famous computational notation used to express natural language structure, and to make development of applications that parse natural language sentences easily. Chomsky proposed the notion of CFG as a model for describing natural languages with following four quantities: Terminals, Non-terminals, Productions, and Start symbol [11].

The grammars described below in CFG express simple grammars to parse an example sentence, ‘Nam is a student supervised by a professor named Kim’ with a set of lexicons enabling aware of terminals’ tokens.

$$\begin{aligned}
 S &\rightarrow NP VP OP \\
 NP &\rightarrow L Be Det N \\
 VP &\rightarrow Verb \\
 OP &\rightarrow Det Verb L
 \end{aligned}
 \tag{1}$$

The states marked with the italic characters means terminals whereas others characters denotes non-terminal states. *L* also denotes unfound literals in the dictionaries, or lexicon set. Other words, such as ‘is’, ‘student’ and ‘supervised by’ were known as each terminal. Now, we can parse the example sentence to make a parse tree. These CFG based definitions, however, only contains the sequential structure, but hardly provide dependency structures that show whether a literal is semantically attached to the verb ‘supervised by’ as a subject or an object. Then, the derived parse tree from the example sentence using the CFG grammar will only contains the structural sequence.

Grasping the dependency structures from a sentence is possible with both structural and the semantic point of view. The semantic view on the sentence can be shown as lexical dependencies which can be generally derived from stochastic analysis to show word-to-word dependency relationships [13][14]. The figure 2 describes the lexical dependency found in the sentence.



**Fig. 2.** Dependency set of the example sentence, ‘Nam is a student supervised by a professor named Kim’

From the figure 2 we can easily capture the semantic dependences between words. As the dotted circles show, there are three meaningful structures that exist in the tree. The domain ontology for representing a triple expression make us generate RDF triples from those dependencies like Nam is *instanceOf* a class **student**, Kim is *instanceOf* a class **professor**, and, Nam *supervisedBy* Kim. Here, it is easy to grasp the difference between the dependency and CFG parse tree even through we do not show the parse tree at here.

Then we can add CFG states to the dependency graph as the figure 3 below. From the integrated graph, the dependencies between CFG states are also easily informed [12]. As the graph shows the VP dominates literals from NP and OP, and the *L* in LP node also depends on *Be*, and *Det N*.

The aim of the CFG-LD is to announce possible dependency structures of a sentence written in the defined grammars. With settled dependencies, the dependency existing between states that appear from an incoming sentence can be caught and can be translated into RDF triples precisely.

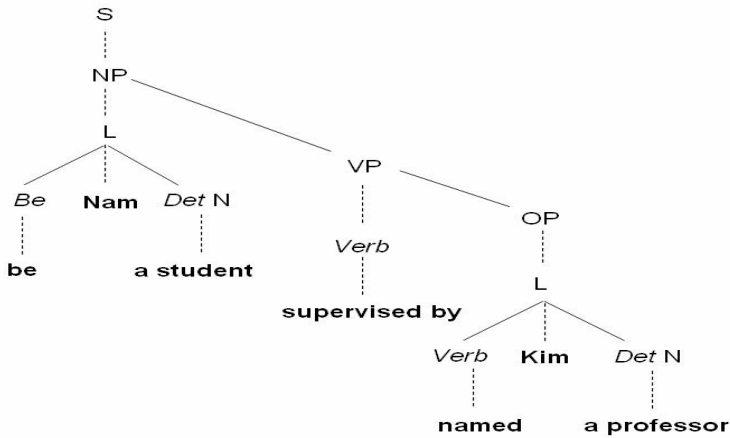


Fig. 3. Integrated parse tree with dependency structures

### 3.2 CFG with Lexical Dependency

Here, we will introduce CFG-LD with an example of a basic grammar definition. As we mentioned, the entities of the domain ontology are a source of lexicons of terminal states; a class in the ontology is presented as a *noun* state whereas a predicate is presented as a *verb*. Supplying additional well-known functional words as lexicons for other terminal states, the words appearing in a sentence can be tokenized and located in its category using the set of the lexicons.

$$\begin{aligned}
 S &\rightarrow NP(S) VP (NP<-V) OP (VP<-O) \\
 NP &\rightarrow L (NP<-S) (S) Be(L<- V) Det Noun (L<-O) \\
 &\quad | L (NP<-S) \\
 VP &\rightarrow Verb (NP<-V) \\
 OP &\rightarrow Det Noun (O) Verb (Noun<-V) \\
 &\quad L (Verb<-S) (OP<-O)
 \end{aligned}
 \tag{2}$$

The above notational rule is an example grammar definition of CFG-LD to handle the sentence ‘Nam is a student supervised by a professor named Kim.’ It shows the rule for a simple sentence composed of non-terminals; ‘NP’, ‘VP’, and ‘OP’. This basic grammar definition is similar to conventional CFG, save the brackets next to each state. The characters in the brackets designate the dependency among the states. As in the first line, we can present the main dependency conducted by NP, VP, and OP. The single character ‘S’ within the brackets, ‘(’ and ‘)’, tells that the state NP has a dependency set, and it has a dependency element ‘Subject.’ The real value of the element will be filled from its nested state because NP is a non-terminal. The symbols with arrow, ‘<-’, within the brackets mean that the state has depending elements such as ‘Verb’ or ‘Object’ on the left side of the arrow, where ‘S’ denotes ‘Subject’, ‘V’ denotes ‘Verb’, and ‘O’ means ‘Object’. In here, the left state should be selected from

the states shown see in the same line. In the second line, a terminal *L* has an actual value for Subject element of the dependency set owed by NP and has its own dependency set starting from Subject with its own value. The new creation of the dependency set constructed with *L* and following *Be* and *noun* is determined by the structure of a sentence varied by an OR operator, ‘l.’

These notations with a set of brackets allow us to describe the sentence structure with lexical dependencies existing in the sentence. Relatively less meaningful words like ‘a’, which is denoted as *Def* state, also can be presented in the sentence structure to elevate the familiarity of expressions, even when we are not intended to deal with the meanings of such words. In the next section, we explain how the rules can be used to parse a sentence using a CFG-LD parser.

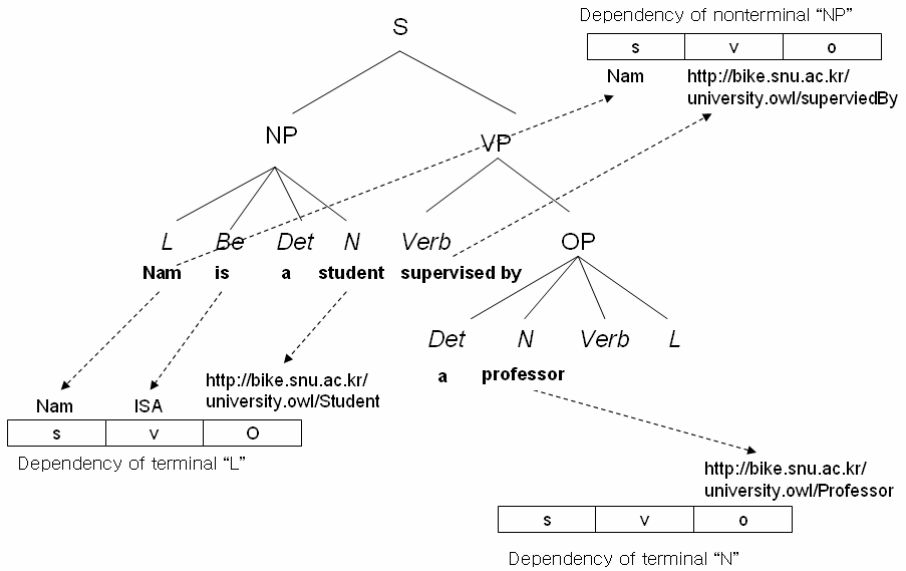
### 3.3 Resolving Sentences Using CFG-LD

Here, we describe how a sentence is resolved in the CFG-LD parser using CFG-LD applied to our previous example. First, as we use CFG rules to deal with sentences, the CFG-LD parser dynamically makes a parse tree from an incoming sentence with a supplied lexicon set. The parse tree can be derived from the example sentence - ‘Nam is a student supervised by a professor named Kim’- or from its front part. From a fundamental description, the lexicon set has ‘is’ as a *Be* state, ‘a’ as a *Def*. The domain ontology provides lexicon sets for nouns and verbs such as ‘student’ and ‘professor’ as *Nouns*, and ‘supervised by’ as a *Verb*. Then, *L* is an unfound token, just a literal.

As figure 4 shows, the parser constructs dependency sets with designated lexical dependencies from the sentence. A set of rectangle boxes means a dependency set created during parsing process. If the parser meets dependency creation instructions such as ‘(S)’, it creates a new dependency set. The dotted lines show which state puts the dependency element values. If a nested terminal state is employed by the incoming sentence, the parser puts the type of state with its value as a dependency element. Especially, *Noun* and *Verb* are states come from the ontology resources, and their actual values become the URIs of the resources.

The produced dependencies sets are used for two functionalities in the CNL editor; triple generation and context-based word recommendation. From the dependency sets in the figure 4, triples can be easily generated from them. Word recommendation also relies on dependency sets. Basically, the editor can know available next states from the parse tree, but it also can narrow down the scope of the proper words using semantic relations among the entities from the domain ontology. In the above example, the editor will recommend next verb-type words by looking ahead the parse tree, and narrow down those verbs by using those *domain* and *range* relations in the ontology. Therefore, the editor will show users those verbs that can take ‘professor’ or the parent classes in the ontology as their range value

To avoid ambiguities, when the dependency sets are captured from the parsed sentences and triples are generated from the sets, the CFG-LD should express following agreements.



**Fig. 4.** Dependency sets from the parse tree from the example sentence, ‘Nam is a student supervised by a professor’

- 1) Every state in CFG is divided into two kinds; ‘dependency involved’, and ‘dependency not involved.’ The ‘dependency involved’ state should provide at least one value for an element to a dependency set when it is fully expanded. The ‘dependency not involved’ state is never involved with any dependency sets. Therefore, the ‘dependency not involved’ state is declared for the definition of additional terms and patternized phrases
- 2) Every state can create a dependency set with elements, e.g., subject, verb and object.
- 3) A state and its nested states can set numerous values for a single element. If an element has multiple values, they are respectively used during triple generations. For example, if a dependency set has two subject values (‘I’ and ‘You’), a verb value (‘like’), and an object value (‘cake’), then two triples, ‘I-like-cake’ and ‘You-like -cake’, are generated.
- 4) A state can be related to multiple dependencies. A state can make a new dependency set and also give values to other dependencies.

With such grammar definitions following the agreements, we can also define grammars specific to languages of interest that exhibit different sequential appearances from sentence constituents, e.g., Korean. The next rule shows the grammar definitions for Korean sentences. We assume that the lexicons of the terminals are provided through pre-declared lexicons and Korean terms for entities of the ontology. For example, ‘초등학생’ is a Korean word for a class ‘ElementarySchoolStudent’, ‘중학생’ is a word for a class ‘MiddleSchoolStudent’, ‘인’ is a postpositional word expressing a ‘is-a’ relation, and ‘와’ is a postpositional



word expressing ‘and’. ‘영희’, ‘철수’, and ‘영수’ are literals, and ‘는 좋아한다’ is a verb corresponding to a property ‘likes.’

$$\begin{aligned}
 S &\rightarrow OP(O) SP(OP \leftarrow S, V) \\
 OP &\rightarrow Class(OP \leftarrow O)(O) is(Class \leftarrow V) literal \\
 &\quad (Class \leftarrow S) ObjectPost \quad (3) \\
 SP &\rightarrow Class(SP \leftarrow S)(O) is(class \leftarrow S) literal \\
 &\quad (Class \leftarrow V) AndPost literal(Class \leftarrow V) Verb(SP \leftarrow V)
 \end{aligned}$$

Through the rule 3, we can notify the structures and dependency of a sentence to the CFG-LD parser- for example, the literal in OP has a relationship with the literals and the verb in SP, so that the editor can make triples from Korean sentences such as ‘초등학생 인 영희 를 중학생 인 철수 와 영수 는 좋아한다’. The triples from the sentence consists of as follows: ‘영희’ is *instanceOf* ‘**ElementarySchoolStudent**’; ‘철수’ is *instanceOf* ‘**MiddleSchoolStudent**’; ‘영수’ is *instanceOf* ‘**MiddleSchool Student**’; ‘철수’ *likes* ‘영희’; and ‘영수’ *likes* ‘영희’.

### 3.4 Internal APIs

The CFG-LD definitions can be used to express grammars with their lexical dependency; however, it is sometimes hard to deal with narratives when a user use anaphoric terms like pronouns and quantifiers. Such anaphoric terms are an inevitable feature of languages because human writers much rely on those terms. They can be replaced with those words that they reference in the document he currently edits.

Therefore, CFG-LD provides a space for specifying dependency values using internal APIs which usable in grammar definitions. By adding the braces, ‘{’, ‘}’, we can also specify the values to be replaced when the dependency set is terminated. In that space, the way for specialized handling of particular anaphoric terms can be presented with several internal APIs or some predefined terms like ISARELATION. The table 1 lists some internal APIs for handling anaphoric terms.

As the functionality descriptions show, the internal functions are mainly related to the referenced instances created in the current document. The first two functions, *nearInstanceOf* and *nearInstancesOf*, return an instance or instance of the designated class already made in the document. The *RecentlyCreatedInstance* and *RecentlyCreatedInstances* functions return any instance recently created. The other functions return a single or multiple instances, or a class suitable to the state’s connected dependency sets.

The rule below shows CFG-LD grammar descriptions using these internal functions. The expressions deal with a pronoun, ‘it’, using the function *RecentlyCreatedInstances()*. The *Noun* with the definite article ‘the’ are handled with the *nearInstanceOf(CLASS)* function where a *Noun* means a class from the domain ontology.

**Table 1.** Internal API set for anaphoric terms

Function Name	Functionality description
<i>nearInstanceOf(Class URI)</i>	Returns the nearest instance of the Class
<i>nearInstancesOf(Class URI)</i>	Returns the near instances of the Class
<i>recentlyCreatedInstance()</i>	Returns instance recently created
<i>recentlyCreatedInstances()</i>	Returns instances recently created
<i>domainInstanceFitDependency()</i>	Returns an instance of domain class of depended predicate
<i>domainInstancesFitDependency()</i>	Returns a set of instances of domain class of depended predicate
<i>rangeInstanceFitDependency()</i>	Returns an instance of range class of depended predicate
<i>rangeInstancesFitDependency()</i>	Returns a set of instances of range class of depended predicate
<i>rangeClassFitDependency()</i>	Returns a class of depended predicate
<i>rangeClassesFitDependency()</i>	Returns a set of classes of depended predicate

$S \rightarrow it (S) \text{ verb } (it<-V) \text{ object } (it<-O)$   
 $it \rightarrow IT (it<-S) \{recentlyCreatedInstance ()\}$   
 $\text{verb} \rightarrow Verb (\text{verb}<-V)$  (5)  
 $\text{object} \rightarrow \text{Def Noun } (\text{object}<-O) \{nearInstanceOf(\text{Noun})\}$

## 4 Architecture of CNL Based Editor

In this chapter, we introduce the architecture of the CNL-based editor. It consists of five main components: Look-Ahead Interface, CFG-LD Parser, Lexicon Pool, Triple Generator, and Predictor, as we can see in figure 5.

1) Look-Ahead Interface: It provides a narrative description interface for a writer. Observing user’s input, it recommends proper next words to let the writer describe sentences with precise structures and semantics.

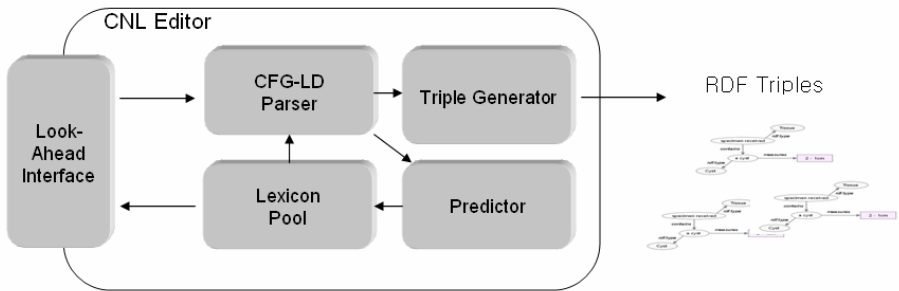
2) CFG-LD Parser: It parses an incoming sentence and makes dependency sets using CFG-LD definitions. First, it tokenizes an incoming sentence with the support

of Lexicon Pool, and then dynamically generates a parse tree and dependency sets from a set of the tokenized words.

3) Predictor: It predicts the next terminal states from the parse tree. It also examines the semantic relations in the domain ontology, e.g., domain and range relations, to provide semantically suitable recommendation

4) Lexicon Pool: It holds lexicon sets from the domain ontology and functional words with their terminal types in the grammar definitions. It provides interfaces for searching for words that have a terminal state and contain a certain substring.

5) Triple Generator: It generates RDF triples from the dependency sets when the sentence is completely terminated. A literal which became an identifier and recursively referenced by other triples in the same sentence are replaced with its URIs in this component.

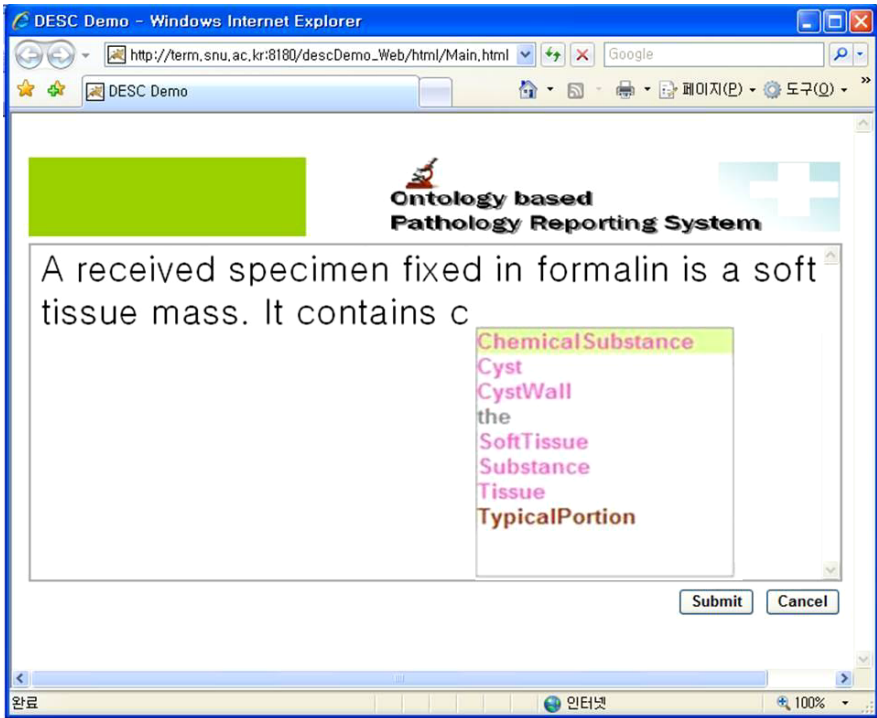


**Fig. 5.** Architecture of CNL based Editor with CFG-LD

When a writer starts to type a sentence with Look-Ahead Interface, the CFG-LD Parser builds a parse tree using the terminals identified by the Lexicon Pool at every modification. By looking ahead the defined grammars and the produced dependency sets, the Predictor foretells the suitable terminals and words. Then, the Lexicon Pool delivers candidate next words to Look-Ahead Interface for guiding a user. At the end of a sentence, when dependency sets are completely filled up by the CFG-LD Parser, the Triple Generator generates RDF triples from the dependency sets.

## 5 Implementation

We implemented a prototype of the Ontology based CNL editor with the explained architecture. In the prototype, AJAX (Asynchronous JavaScript and XML) was employed for implementation of the Look-Ahead Interface. The other components are hidden in the screenshot but, they were supplied a pathology ontology written in OWL and some functional words for medical pathology examination descriptions.



**Fig. 6.** Screenshot of Ontology based Pathology Reporting System exploiting CFG-LD in English narratives editing

User-displayable labels written in the multi language words of the ontology entities are provided, for example, a property, *'fixed\_in'*, is offered as the multiple labels, such as, 'fixed in' and '고정된' which a Korean word. Several grammars are defined and informed to the CFG-LD parser.

The figure 6 shows the English narratives editing. If a writer starts to enter characters for the description, the editor displays proper terms to be selected by his/her for precise sentence expressions. The first sentence is realized through a patternized grammar rule which starts with a phrase, 'A received specimen.' The pink-colored terms in the tool-tip box mean words from the ontology. If a user sends the document by clicking the 'Submit' button, the editor stores the narratives as RDF triples.

Figure 7 also shows Korean narratives editing. The first sentence presented in the editor is attained through a patternized grammar rule which starts with a Korean phrase, '받은 조직' which means 'a received specimen.' The sentence has the same meaning with 'Received Specimen fixed in formalin is a dark-black soft tissue mass,' but with different sequential structures. Because the editor is equipped with the Korean grammars, the tool-tip box shows some Korean words.

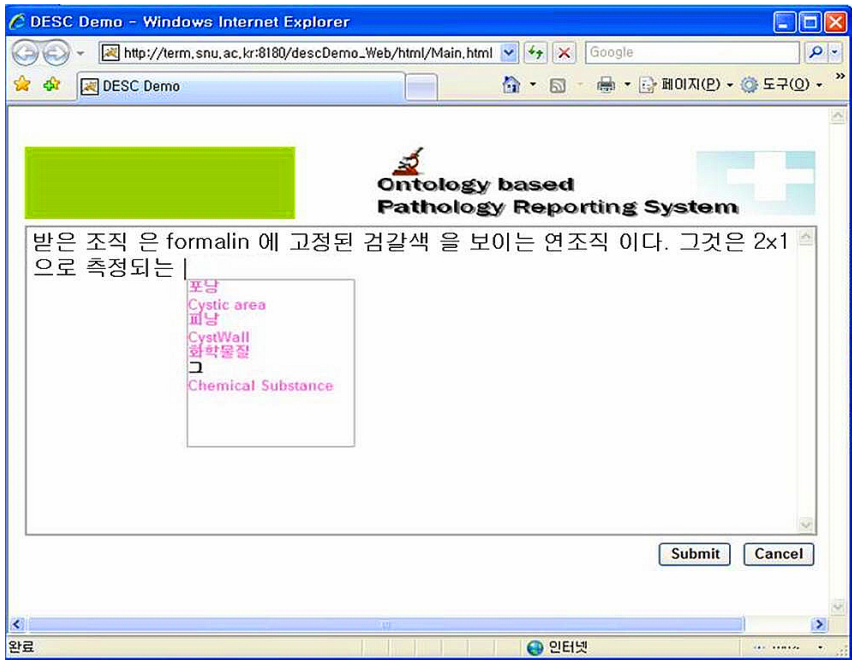


Fig. 7. Screenshot of Ontology based Pathology Reporting System exploiting CFG-LD in Korean narratives editing

## 6 Conclusion

We proposed a slightly revised grammar definition system named CFG-LD (CFG with Lexical Dependency) in this paper. It permits grammar expressions with lexical dependency designation among sentence constituents. To meet our research purpose for the enablement of domain-specific descriptions, we arranged grammars usable in the ontology-based CNL editor through the definition system. The editor, we implemented with a CFG-LD parser, provides guidance on proper choice of words and translates the end results into RDF triples.

We expect that such grammar definitions can facilitate the definition of the approved and recommended sentence structures for domain-specific narratives. With the implemented CFC-LD Parser, it enables the development of a lightweight CNL editor with domain-specific ontology. The CNL editors will give us a chance to get structured data from users' narratives, along with 1) more sophisticated expressions, 2) patternized expressions, and 3) informal expressions consisting of multi language constituents. However, the dependency structure appearing in a sentence can vary due to types and semantics of language constituents. Though it might be possible to describe states in a more detailed way according to their characteristics, it is by no means an easy task to complete. We are planning to improve our system and CFG-LD to unburden those efforts for the definitions of CFG-LD grammars. In this paper, validation and resolution of conflicts between the grammar rules were not considered

when a sentence is parsed by the CFG-LD parser. We will conduct further improvements of the parser considering occurrence of conflicts in our future works.

## References

1. Bernstein, A., Kaufmann, E., Fuchs, N., von Bonin, J.: Talking to the Semantic Web – A Controlled English Query Interface for Ontologies. In: proceeding of 14th Workshop on Information Technology and Systems (2004)
2. Schwitter, R., Tilbrook, M.: Controlled natural language meets the Semantic Web. In: Proceedings of the Australasian Language Technology Workshop 2004, pp. 55–62 (2004)
3. Schwitter, R.: Controlled Natural Language as Interface Language to the Semantic Web. In: proceeding of 2nd Indian International Conference on Artificial Intelligence, pp. 1699–1718 (2005)
4. Fuchs, N.E., Schwertel, U., Schwitter, R.: Attempto Controlled English – Not just another logic specification language. In: Flener, P. (ed.) LOPSTR 1998. LNCS, vol. 1559, pp. 1–20. Springer, Heidelberg (1999)
5. Bernstein, A., Kaufmann, A.: GINO- A Guided Input Natural Language Ontology Editor. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 144–157. Springer, Heidelberg (2006)
6. Fuchs, N.E., Schwertel, U.: Reasoning in Attempto Controlled English. In: Bry, F., Henze, N., Małuszyński, J. (eds.) PPSWR 2003. LNCS, vol. 2901, Springer, Heidelberg (2003)
7. Sowa, J.: Graphics and languages for the Flexible Modular Framework. In: Wolff, K.E., Pfeiffer, H.D., Delugach, H.S. (eds.) ICCS 2004. LNCS (LNAI), vol. 3127, pp. 31–51. Springer, Heidelberg (2004)
8. Sowa, J.: Common Logic Controlled English (2004), <http://www.jfsowa.com/clce/specs.htm>
9. Kim, H.G., Ha, B.H., Lee, J.I., Kim, M.K.: Narrative Information Processing based on Controlled Natural Language in EMR Systems, International Journal of Medical Informatics (in Press, 2007)
10. Kim, H.G., Ha, B.H., Lee, J.I., Kim, M.K.: A multi-layered application for the gross description using semantic web technology. International Journal of Medical Informatics 74, 399–407 (2005)
11. Chomsky, N.: Three models for the description of language. IRE Trans. Info. Theory 2(3), 113–124 (1956)
12. Barbero, C., Lesmo, L., Lombardo, V., Merlo, P.: Integration of syntactic and lexical information in a hierarchical dependency grammar. In: Proceedings of the Workshop on Processing of Dependency-Based Grammars (ACL-COLING), pp. 58–67 (1998)
13. Carroll, G., Charniak, E.: Two experiments on learning probabilistic dependency grammars from corpora. Technical Report TR-92, Department of Computer Science, Brown University (1992)
14. Hellwig, P.: Dependency unification grammar, pp. 593–635. Dependency and Valency, Walter de Gruyter (2003)

# Web Search Personalization Via Social Bookmarking and Tagging

Michael G. Noll and Christoph Meinel

Hasso-Plattner-Institut an der Universität Potsdam, Germany

**Abstract.** In this paper, we present a new approach to web search personalization based on user collaboration and sharing of information about web documents. The proposed personalization technique separates data collection and user profiling from the information system whose contents and indexed documents are being searched for, i.e. the search engines, and uses social bookmarking and tagging to re-rank web search results. It is independent of the search engine being used, so users are free to choose the one they prefer, even if their favorite search engine does not natively support personalization. We show how to design and implement such a system in practice and investigate its feasibility and usefulness with large sets of real-world data and a user study.

## 1 Introduction

The recent emergence and success of folksonomies and the so-called *tagging* with services such as del.icio.us or Flickr have shown the great potential of this simple yet powerful approach to collect metadata about resources. Unlike traditional categorization systems, the process of tagging is nothing more than annotating documents with a flat, unstructured list of keywords called *tags*. Although the number of peer-reviewed research on tagging is still comparatively low, several studies have already analyzed the semantic aspects of tagging and why it is so popular and successful in practice [1], [2]. A common argument is that tagging works because it strikes a balance between the individual and the community: the cost of participation is low for the individual, and tagging a document benefits both the individual and the community.

In this paper, we describe and analyze a system for personalization of web search based on such tagging metadata, i.e. user-contributed information about documents. Traditional web search has been rather impersonal: returned search results are the outcome of a function applied to the entered query. From a set of documents, those items that best match the query are returned to the user. Characteristics of the user are not taken into consideration when processing a query. Personalized web search on the other hand integrates user-specific data into the process of finding the best matching documents to a search query by increasing the amount of a priori input information available to search algorithms.

Pitkow et al. [3] describe two general approaches to web search personalization. The first modifies or augments a user's original query. For instance, a query for "nyt" might be translated to "new york times". The second approach will run

the unmodified original query for all users but re-rank the returned results based on information about the individual user. For the work in this paper, we will focus on the second case, i.e. re-ranking of the list of search results returned by a search engine. The proposed personalization technique benefits from the strategy of search engines to distribute their top search results among the various meanings and topics<sup>1</sup> of a query [4].

In the next section, we present our approach to web search personalization and describe its theoretical backgrounds and concepts. Section 3 outlines the technical design and implementation for running and using such a system in practice. The experiments in section 4 analyze the feasibility and usefulness of our approach with large sets of real-word data and a user study, followed by the conclusions in section 5.

## 2 User-Driven Personalization

Our approach to web search personalization is based on social bookmarking and tagging. The proposed technique for web search personalization is independent of the search engine being used, so users are free to choose the one they prefer. Personalization in this paper is defined as re-ranking the list of search results returned by a search engine [3]. Our approach exploits the conceptual links between web search, social bookmarking and tagging. From a high-level perspective, users search the Internet via a search engine, find the web documents or information they are looking for, and if the information is worth storing for later use or sharing with other users, they bookmark the web document [5]. Our approach makes use of bookmarks combined with tagging and collaboration for learning more about users *and* the web documents which are being searched for. Whenever a user bookmarks a web document, more data is available about the user and the bookmarked document and thus for personalization. We have shown in [6] that tagging metadata contains information which is not directly contained within a document, and so we argue that integrating tagging information can help to improve personalization and retrieval techniques.

From a conceptual point of view, personalization based on tagged bookmarks is a mixture of explicit and implicit personalization. Neither are users prompted to enter their preferences directly in a special configuration step, nor are they monitored or tracked in the background without being aware of it. User data is collected rather explicitly because bookmarking a web document and adding metadata like tags are manual user tasks. Users know exactly when information is collected for personalization. However, unlike traditional explicit personalization techniques, these manual tasks are not necessarily an additional burden for users. Bookmarking web documents has a direct benefit for users even without personalization, mitigating the practical problems of explicit feedback techniques

---

<sup>1</sup> For instance, a search for “jaguar” returns links to web documents about the car, the feline and the Mac OS version of the same name in the top 10 search results in order to increase the chance that at least one of these topics matches the user’s intended search.



[7]. The recent success of social bookmarking services such as del.icio.us which has a community of more than one million registered users<sup>2</sup>, has shown that users are indeed willing to make use of bookmarking and share this information with other users. We therefore argue that expecting explicit actions from a user is reasonable as long as the “cost” and effort of the action is low compared to the user’s subjective benefits and outcomes. The emergence and success of tagging (see Sect. 2.1) has been attributed to the same reasons [1], [2]. In addition, we present an easy way for automated creation of tagged bookmarks called *tagmarking* in Sect. 2.1 in order to close the usability gap to fully implicit techniques.

Our personalization technique consists of two main elements. First, the collection and aggregation of data about users and documents, and second, the personalization of web search based on this data. Normally, these steps are performed by the search engines themselves. In our approach however, information about web documents is collaboratively collected and shared by the community of users via social bookmarking, and - together with a user’s individual profile based on her own bookmarks - used to personalize the generic search results returned by a search engine. This means it is possible to provide web search personalization independent of the search engine being used.

## 2.1 Data Collection

**Bookmarking.** When a user bookmarks a web document, she stores it for later use [5]. This observation leads to our two basic assumptions about bookmarking:

1. users only bookmark documents valuable/relevant to them (or their friends)
2. users have an incentive to add meaningful metadata to bookmarks

When a document is of no interest to a specific user, it is unlikely she would bookmark it. And when users actually do store bookmarks in order to find them again later, they have an incentive to add meaningful metadata to them, for example in the form of tags. Finding word associations for describing a document in the form of tags is a subjective user task, which should help with the differentiation of a user’s characteristics when performing personalization. It is possible to help users with entering metadata, for example by presenting the list of the most frequently used tags to annotate the document [8].

“Bookmarking” a web document by a user is defined in this paper as storing a bookmark including any additional metadata at a social bookmarking service. A social bookmarking service is a central online service which enables users to add, modify, and delete bookmarks of web documents with additional metadata. The social aspect of the bookmarking service allows a user to share this information with the community. On popular social bookmarking services like del.icio.us, users can browse the bookmark collection of others and request community information about a web document (identified via its URL) from the service. In

---

<sup>2</sup> <http://blog.del.icio.us/blog/2006/09/million.html>, last retrieved on July 24, 2007.

this paper, the prime purpose of a social bookmarking service is to collaboratively collect and share metadata about web documents in the form of tags. Use of tagging metadata implies that even such web documents may be processed for which existing content extraction and indexing techniques do not work efficiently because the depicted content is difficult to analyze (e.g., multimedia content such as videos).

**Tagging.** The recent emergence and success of tagging with services such as del.icio.us or Flickr have shown the great potential of this simple approach to add metadata to documents. Unlike traditional classification or categorization systems, the process of tagging is nothing more than annotating documents with a flat, unstructured list of keywords called “tags”. Users can browse or query documents by tags, and so-called “tag clouds”, a hyperlinked collection of most frequently used tags, provide a rudimentary but often sufficient way to find popular and interesting content. Tagging can be interpreted as a relation  $R_{tagging} \subseteq D \times U \times T$  where  $D$  is the set of documents,  $U$  the set of users and  $T$  the set of tags. In our case, the act of bookmarking a document with tags by a user creates one or more tuples as described by the relation above. Documents are identified by their URLs and users by their account name in the bookmarking service. We will use tags associated with bookmarks to collect information about documents and users alike.

Storing and organizing bookmarks with the help of tags mitigates some of the problems of hierarchical bookmark classification (for example, where to file a bookmark if it fits to more than one category) and increases findability. This is a benefit especially for users with lots of bookmarks, which gives yet another incentive to actively bookmark and tag web documents, which in turn improves personalization. As we will see in Sect. 4, even a modest amount of tagged bookmarks may lead to very good personalization performance.

**Tagmarking.** In our system prototype, we have developed a browser extension which allows users to “tagmark” pages found via search queries. Tagmarking exploits the similarity between tags and search keywords [9]. The basic idea of the Tagmark extension is to store the search query, e.g. “gutenberg poe raven”, in memory while the user evaluates search results. Whenever she finds a relevant web document, she can bookmark it with a single click on the “Tagmark” button, and the browser extension will automatically translate the search query to tags and add them to the bookmark (here: “gutenberg”, “poe”, “raven”). Tagmarking is a very convenient way to enhance the search and bookmarking experience by enabling users to store bookmarks with meaningful metadata with just a single click, and it helps to collect more input data for a user’s profile and personalization. Tagmarking reduces the cost and effort of bookmarking and tagging a document, thereby closing the usability gap to personalization approaches based on fully implicit user actions.

## 2.2 Data Aggregation

**User profiling.** A user’s bookmark collection  $R_u$  can be described as a relation  $R_u \subseteq D \times T$  (cf.  $R_{tagging} \subseteq D \times U \times T$  in section 2.1) and implemented as a tag-document matrix  $M_d$  with  $m$  tags and  $n$  documents (and thus  $n$  bookmarks).

$$M_d = \begin{bmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{m1} & \cdots & c_{mn} \end{bmatrix}, c_{ij} \in \{0, 1\}$$

A bookmark of a document  $d_j$  is a column (vector)  $\mathbf{b}_j$  with its components  $c_{ij}$  set to 1 if tag  $t_i$  is associated with  $d_j$  and 0 otherwise. The user profile  $\mathbf{p}_u$  is a vector with  $m$  components as follows:

$$\mathbf{p}_u := M_d \cdot \boldsymbol{\omega}_d = \begin{bmatrix} c_1^* \\ \vdots \\ c_m^* \end{bmatrix}, c_i^* \in \mathbb{N}_0$$

In our implementation, we define the weight  $\boldsymbol{\omega}_d^T := \mathbf{1}^T = [1 \cdots 1]$  with  $n$  dimensions, thereby assigning equal importance to all  $n$  documents. Here,  $c_i^*$  denotes the total count of tag  $t_i$  for the user’s bookmark collection. We assume that frequently used tags are more interesting and relevant to a user than rarely used tags. Building a user’s profile as described implies that it can be updated incrementally whenever a user adds a new bookmark to her collection or modifies or deletes an existing one. By this, personalization can adapt to shifts of interests over time. Table 1 shows an exemplary user profile.

**Table 1.** Exemplary profile for a user (left) and a document (right)

<b>User</b> jsmith		<b>URL</b> <a href="http://iswc.semanticweb.org/">http://iswc.semanticweb.org/</a>	
“open source”	13	“iswc”	156
“programming”	19	“computing”	48
“proprietary”	2	“programming”	66
“research”	10	“conference”	90
“security”	21	“research”	111
“semantic web”	34	“semantic web”	140

**Document profiling.** Metadata about web documents is collected by the community of users submitting bookmarks to the bookmarking service. In contrast to individual user profiles, document profiles are a collaborative work. Whenever a user creates or modifies a bookmark of a web document, the information is shared with the community and the document’s profile is updated.

Metadata about a document  $d$  can be described as a relation  $R_d \subseteq U \times T$  and implemented as a tag-user matrix  $M_u$  with  $m$  tags and  $n$  users. A bookmark of document  $d$  by user  $u_j$  is a column (vector)  $\mathbf{b}_j$  with its components  $c_{ij}$  set

to 1 if tag  $t_i$  is associated with  $d$  by user  $u_j$  and 0 otherwise. Similar to user profiles, the document profile  $\mathbf{p}_d$  is a vector with  $m$  components generated by  $\mathbf{p}_d := M_u \cdot \boldsymbol{\omega}_u$ . In our implementation  $\boldsymbol{\omega}_u^T := \mathbf{1}^T = [1 \cdots 1]$  with  $n$  dimensions, thereby assigning equal importance to all  $n$  users. Table 1 shows an exemplary document profile.

### 2.3 Personalization

After having collected data in the previous sections and transformed it into user profiles and document profiles, we can now match users and documents in the actual personalization step with the goal of re-ranking a list of search results as shown in Algorithm 1.

---

#### Algorithm 1. PERSONALIZE(*user*, *documents*)

---

**Require:** The user’s profile and a sequential list of document profiles.

**Ensure:** The personalized list of documents for the user.

```

1: for all  $d$  in documents do
2:   CALCULATE  $\text{similarity}(\text{user}, d)$ 
3: end for
4: # highest to lowest, stable sort
5: SORT documents BY SIMILARITY
6: return documents

```

---

The user-document similarity is a dimensionless score used for relative weighting and re-ranking of documents within a given list, defined as:

$$\text{similarity}(u, d) := \mathbf{p}_u^T \cdot \|\mathbf{p}_d\|$$

where the naïve “normalization” of the document profile,  $\|\mathbf{p}_d\|$ , simply sets all non-zero components of  $\mathbf{p}_d$  down to 1. The main idea is to leverage community-supplied metadata mostly for identifying commonly agreed “perceptions” of documents, and let the unmodified user profile be the key factor for personalization. At the moment, we are experimenting with more sophisticated normalization techniques for both user and document profiles.

The similarity for the exemplary user and the exemplary profile of the ISWC home page in Table 1 is 63. The described user-document similarity favors documents with tags that are frequently applied by the user herself, and the personalization algorithm tends to promote known, similar documents and demote non-similar or unknown documents. *Known* in this case means that documents have already been bookmarked and tagged by users. Thus, an important factor for the viability of this personalization approach in practice is the availability of user-supplied metadata for web documents, which we study in Sect. 4.1.

The left side of Table 2 shows an exemplary search query on Google for “security” by a user with a strong interest in information technology and network security. After personalization, the result list looks as shown on the right side of Table 2. Websites related to IT security have been promoted to the top, while

pages such as the White House’s information page about Homeland Security have been demoted to the bottom. In this example, the algorithm has confirmed the top-ranked search result of SecurityFocus for this user, so there is no change for position 1. However, the home page of CERT, a center of Internet security expertise, has been pushed from position 9 to 2. The US Department of Homeland Security lost six positions and is now at the bottom of the list. Note that the website of the US Social Security Administration has been promoted even though it is not related to IT security; this is because the user profile also shows interests in insurance matters.

**Table 2.** Google search results for “security” before (left) and after personalization. URL scheme and “www.” prefix omitted, long URLs have been truncated.

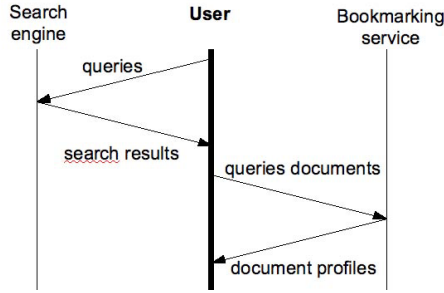
#	URL	#	Δ#	URL
1	securityfocus.com/	1	•	securityfocus.com/
2	microsoft.com/security/	2	↑ +7	cert.org/
3	microsoft.com/technet/security/def...	3	•	microsoft.com/technet/security/def...
4	dhs.gov/	4	↑ +4	w3.org/Security/
5	whitehouse.gov/homeland/	5	↑ +2	ssa.gov/
6	windowsitpro.com/WindowsSecurity/	6	↑ +4	nsa.gov/
7	ssa.gov/	7	↓ -5	microsoft.com/security/
8	w3.org/Security/	8	↓ -2	windowsitpro.com/WindowsSecurity/
9	cert.org/	9	↓ -4	whitehouse.gov/homeland/
10	nsa.gov/	10	↓ -6	dhs.gov/

## 2.4 Putting It All Together

The complete process of web search personalization works as follows:

1. the user makes a query on a search engine of her choice
2. a list of documents is returned by the search engine as result of the query
3. for each result document, the document profile is retrieved from the bookmarking service
4. the user’s profile is generated on the client side (the user’s computing device)
5. the list of documents is personalized based on user-document similarity

Steps 1-3 require communication between the user’s client and the search engine or bookmarking service. The communication flow is shown in Fig. 1. Steps 4-5, which include the actual personalization, are performed only at the client side. The proposed personalization technique has several benefits. First, bookmarking a web document will improve future web searches even if the user is not actively using a search engine. When a user bookmarks a web document recommended to her via email, it will still affect her user profile. Second, the technique allows the personalization of search results from different search engines. Because the user is in control of her user profile at any time, i.e. it is not managed by a specific search engine, she can personalize multiple (even competing) search engines with the same user data. Third, it is even possible to



**Fig. 1.** Communication flow for web search personalization

personalize a search engine which natively does not support personalization of search results. Fourth, it is relatively easy to explain users why a web document has been promoted or demoted during personalization (e.g., “...because you have tagged a lot of bookmarks with...”). Fifth, the computational expense of the algorithms for generating user and document profiles and calculating user-document similarity is very low. Personalization can easily be performed on client devices with limited energy or processing power such as mobile phones or PDA.

### 3 System Setup

We have designed and implemented a system prototype for web search personalization with the following components. First, a custom social bookmarking service for adding, modifying and querying metadata about web documents, and second, a browser extension for on-the-fly personalization of search results and managing bookmarks with tags. The third required “component” is any search engine whose search results shall be personalized.

The technical implementation of the bookmarking service follows the system we have described in detail in [10]. The service allows users to add, modify and delete bookmarks of web documents with additional metadata such as tags. For each bookmarked web document, the service aggregates metadata submitted by the user community into a document profile, which can be accessed via a web API. The browser extension is installed on the user’s computing device. The extension is responsible for carrying out steps 3-5 as described in Sect. 2.3 by communicating with the bookmarking service over its web API. It will transparently personalize the search result pages of search engines like Yahoo or Google by modifying the DOM tree of these web pages on-the-fly. It will also highlight web documents already bookmarked by the user for easier reference<sup>3</sup>. From a user perspective, web search personalization in our system setup is completely

<sup>3</sup> We are currently working on a feature that will retrieve potentially relevant bookmarks from a user’s bookmark collection based on the entered search query. This will allow us to present search results to the end user which are not in the search engine’s index at all, e.g. a bookmark to a non-public intranet web page.

transparent and happens instantly even though extra communication with the bookmarking service is required. The technical implementation of DOM tree manipulation, i.e. displaying the personalization results to the user via the browser UI, is specific to a particular search engine. On a conceptual level, however, the personalization of search results is independent of the search engine being used. The extension enhances the browser GUI with interface elements for saving tagged bookmarks to the bookmarking service and features a “tagmarking” button as described in Sect. 2.1.

## 4 Experiments and Evaluation

Personalization in this paper relies on the strength of the user community as it requires that search result documents have been tagged by users. For documents without bookmarks or tags, our personalization approach is not possible in practice because metadata about them is missing and thus document profiles cannot be generated. One if not the most important task is therefore to analyze the expected availability of metadata for search result documents in the real-world. However, the custom bookmarking service we have implemented limits the possibilities of sharing and comparison of research results. We have therefore decided to use the public bookmarking service of del.icio.us with its large user community (more than one million registered members in 2006) as information source for our experiments.

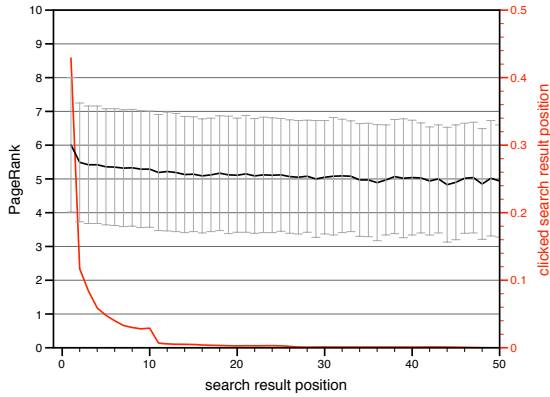
### 4.1 Quantitative Analysis

In a previous study [6], we analyzed the availability of user-contributed metadata for a random sample of 100,000 web documents from the Open Directory. One finding was a correlation between tagging metadata and a document’s popularity (measured by its Google PageRank): the more popular a web page, the more likely the page is to be bookmarked and tagged by users. We can thus infer information about bookmarking and tagging metadata of search result documents by analyzing their PageRank distribution. For the work in this paper, we combined our previous results with an analysis of the AOL500k corpus<sup>4</sup>, of which we evaluated ~1,750,000 queries with 1,000,000 clicked search results. For each clicked document, we retrieved PageRank information from Google.

First, we looked at documents and analyzed the average PageRank of web pages for each search result position. The top positions have an average PageRank of 5.4 or higher as shown by the black line in Fig. 2. The red line denotes the click frequency per search result position. The top 5 positions account for approx. 75% of all clicked search results, most of which is contributed by position 1. The drop between position 10 and 11 is caused by the default configuration of AOL search to show only 10 results per result page (similar to most popular

---

<sup>4</sup> The AOL500k corpus is a collection of ~20,000,000 search queries from ~650,000 anonymized but uniquely identifiable users sampled by AOL over a period of three months in 2006. The corpus was formerly available from <http://research.aol.com/>



**Fig. 2.** Average PageRank of result links (black) with error bars (gray) denoting standard deviations. Click frequency of users is shown by the spiked red line.

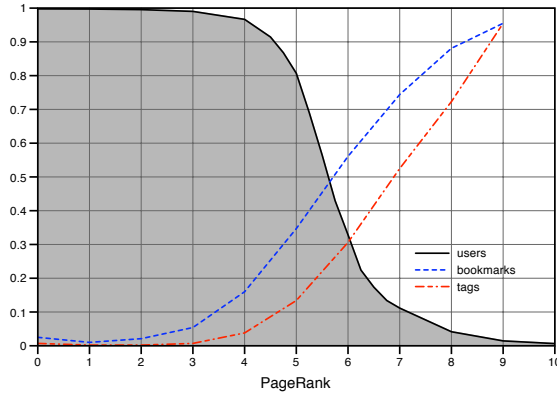
search engines), which means that users are very unlikely to look for search results beyond page 1. This result is encouraging for our personalization technique. On one hand, search results documents are likely to be bookmarked and tagged due to the high expected PageRank, and on the other hand, the re-ranking approach can prove to be efficient in practice because it is very often sufficient to personalize just the first result page.

Second, we looked at users and averaged the PageRank of clicked search results for each user in our AOL500k subset, i.e. individual click preferences regardless of search result position or result page. The black line in Fig. 3 shows the percentage of users with an average clicked PageRank of  $x$  or higher. 80% of users have an average clicked PageRank of 5 or higher, 33% a PageRank of 6 or higher. The dashed and dashed+dotted lines describe the probability of a document to be bookmarked or tagged, respectively, based on our findings in 6. While the numbers for PageRanks less than 5 might seem low at first glance, the *combined* probability of  $n$  result documents with varying PageRank to be bookmarked or tagged can be high enough in practice for good personalization results as we will see later. Additionally, the usage of social bookmarking services and collaboration platforms such as del.icio.us, on which the evaluations in 6 is based, is increasing in the Web today, and thus the availability of tagging metadata will increase over time, too.

In the next experiment, we extracted the so-called *popular tags* from del.icio.us and run searches for each tag on Google. For each document on the first result page (see above), we retrieved the document’s *common tags*<sup>5</sup> from del.icio.us,

<sup>5</sup> del.icio.us limits a document’s *common tags* to its 25 most popular tags, which means that the list of all tags attached to a document might actually be larger in practice. The reason for retrieving just the common tags of a document instead of all tags, i.e. even rarely used ones, is due to technical restrictions by del.icio.us. Still, we argue that even if all tagging information was available, it would be recommended to perform some sort of thresholding or preprocessing anyway to remove “tag noise”.





**Fig. 3.** Percentage of users with an average PageRank for clicked search results equal to or higher than  $x$ , i.e.  $P_u(\text{PageRank} \geq x)$ . The dashed and dashed+dotted lines denote the frequencies of bookmarked or tagged documents for PageRank  $x$ , respectively, i.e.  $P_d(\text{bookmarked}|\text{PageRank} = x)$  and  $P_d(\text{tagged}|\text{PageRank} = x)$ .

**Table 3.** Average number of bookmarks and tags of a document per search result position. The peak of 784 for position 10 is caused by two extreme data points in our sample; it drops to 519 when these two data points are removed.

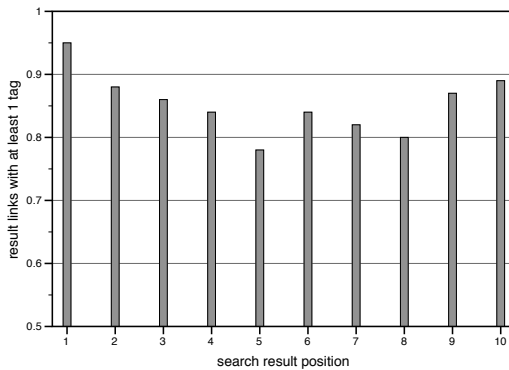
#	Bookmarks	Tags	#	Bookmarks	Tags
1	1450	19.8	6	456	13.7
2	627	16.4	7	495	13.4
3	1199	15.5	8	574	13.7
4	451	14.2	9	404	14.0
5	610	12.5	10	784	13.3

similar to steps 1-3 in Sect. 2.4. The final data set consisted of 140 tags and associated search queries with 1400 result links. A total of 981,989 user bookmarks with 20,498 tags (2,300 unique) were stored at del.icio.us for these 1400 web documents, netting 701 bookmarks and 14.6 tags per document in average. The full details are shown in Table 3. Around 9 out of 10 search results are bookmarked and 8.5 out of 10 search results are tagged by users (see Fig. 4). In other words, we can expect to be able to personalize approx. 85% of search results per query in practice - at least for popular keywords - when using the del.icio.us bookmarking service as the sole information source for tagging metadata.

### 4.2 Qualitative Analysis

To examine the usefulness of our personalized search system for individual users, we let 8 participants evaluate the top 10 search results, i.e. the first result page, for 13 queries each, totaling 104 queries. Each user had her or his personal set of bookmarks, which was used to build the individual user profile. The public

bookmark repository of del.icio.us was used for creating the document profiles. Search queries varied by user by their bookmarking history. Web search results were collected from Google Search. For each query, participants were presented two search result lists: the original, “generic” list from Google Search, and the personalized version. The experiment was conducted as a blind test, i.e. the result lists were presented in random order so as not to bias the participants. Participants were asked to determine which of the two results lists of a query was “better” tailored to them, where *better* was defined as ranking highly relevant results at the top of the lists and ranking irrelevant results at the bottom, i.e. promoting “good” results and demoting “bad” results. Participants could also vote for a draw if they could not decide which list was better. The participant’s job functions included researchers, system administrators, webmasters and software developers. All were computer literate and familiar with web search. The average number of bookmarks for a participant was 153.



**Fig. 4.** Percentage of search result links which have at least 1 associated del.icio.us “common” tag

In our experiment, the personalized list was considered better than or as good as the unmodified result list in 70.2% of the queries (63.5% and 6.7%, respectively). The unmodified result list as returned by the search engine was preferred in 29.8% of the queries. An interesting observation was the low frequency (6.7%) of the cases where users could not prefer one list over the other. The personalization had problems to improve search results for users which were only broadly interested in a particular topic, suggesting that it performs better for “expert” user profiles. For instance, a user with lots of bookmarks tagged just with “web design” will not benefit as much from personalization as a user who tags her bookmarks about web design more granularly. Previous studies such as [11] have shown that most search queries are rather short, with the average search query consisting of only one or two words. Our approach showed its strength particularly in the case of disambiguation of words and contexts (see the “security” example in Sect. 2.3), which indicates that it is especially helpful for such queries.

## 5 Conclusions

In this paper, we presented a new approach to personalization of web search by leveraging social bookmarking and tagging. We have shown how to design and implement such a system in practice and investigated its feasibility and usefulness with large sets of real-word data and a user study. Our evaluation results are encouraging and suggest that personalization based on social bookmarking and tagging is a useful addition to the web toolset and that the subject is worth further research, in particular with regard to the increasing popularity of social and collaborative services in the WWW today.

## 6 Related Work

To the best of our knowledge, this is the first study on using social bookmarking and tagging techniques for personalization of web search and its evaluation in a real-world scenario. Next to the references mentioned throughout the text, the following works are related to the work described in this paper. Bao et al. [12] use social annotations to improve page ranking in generic web search. They propose a similarity measure between social annotations and web queries, and use tagging information to measure the popularity of a web page from an end user's perspective. Next to the different research focus, an important difference to our work is that their experimental data set is restricted to web pages already stored at del.icio.us whereas our evaluation is based on a indiscriminate, random sample of web pages. Boydell and Smyth [9] describe a technique for document summarization that uses informational cues as the basis for summary construction. Social bookmarks are one of the cues used in their work, and they stress the similarity between tags and search query keywords for creating snippet texts for summarization. Integration of social bookmarking information helped to improve the quality of their system when compared with traditional summarization techniques. Sugiyama et al. [13] integrate collaborative filtering techniques into search personalization similar to the social bookmarking approach in this paper. However, the collaborative aspects focus on identifying similar users based on their daily browsing histories, not on sharing information about the documents being searched for as is the case for social bookmarking. In addition, the input data required for their collaborative filtering algorithms, i.e. detailed browsing history information about other users, is generally not available to an individual user of a search engine. Similar to tagging information supplied by end users, ranking and classification techniques may use incoming or outgoing hyperlinks of a web document to infer information about the document and its neighbors by associating terms with the web documents that are themselves not part of the documents [14]. Here, the descriptive annotations of other document authors (as opposed to the document readers in the case of social bookmarking and tagging) help to gain more knowledge about the documents at hand.

## References

1. Golder, S.A., Huberman, B.A.: Usage patterns of collaborative tagging systems. *J. Inf. Sci.* 32(2), 198–208 (2006)
2. Marlow, C., Naaman, M., Boyd, D., Davis, M.: Ht06, tagging paper, taxonomy, flickr, academic article, to read. In: *Proceedings of HT 2006*, pp. 31–40 (2006)
3. Pitkow, J., Schütze, H., Cass, T., Cooley, R., Turnbull, D., Edmonds, A., Adar, E., Breuel, T.: Personalized search. *Commun. ACM* 45(9), 50–55 (2002)
4. Wedig, S., Madani, O.: A large-scale analysis of query logs for assessing personalization opportunities. In: *Proceedings of SIGKDD 2006*, pp. 742–747 (2006)
5. Abrams, D., Baecker, R., Chignell, M.: Information archiving with bookmarks: personal web space construction and organization. In: *Proc. of SIGCHI 1998*, pp. 41–48 (1998)
6. Noll, M.G., Meinel, C.: Authors vs. readers: A comparative study of document metadata and content in the www. In: *Proceedings of ACM DocEng 2007*, pp. 177–186 (2007)
7. Carroll, J.M., Rosson, M.B.: Paradox of the active user. In: Carroll, J.M. (ed.) *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*, Bradford Books, pp. 80–111 (1987)
8. Sen, S., Lam, S.K., Rashid, A.M., Cosley, D., Frankowski, D., Osterhouse, J., Harper, F.M., Riedl, J.: tagging, communities, vocabulary, evolution. In: *Proceedings of CSCW 2006*, pp. 181–190 (2006)
9. Boydell, O., Smyth, B.: From social bookmarking to social summarization: an experiment in community-based summary generation. In: *Proceedings of IUI 2007*, pp. 42–51 (2007)
10. Noll, M.G., Meinel, C.: Design and anatomy of a social web filtering service. In: *Proceedings of CIC 2006, Hong Kong*, pp. 35–44 (2006)
11. Jansen, B.J., Pooch, U.: A review of web searching studies and a framework for future research. *J. Am. Soc. Inf. Sci. Technol.* 52(3), 235–246 (2001)
12. Bao, S., Xue, G., Wu, X., Yu, Y., Fei, B., Su, Z.: Optimizing web search using social annotations. In: *Proceedings of WWW 2007*, pp. 501–510 (2007)
13. Sugiyama, K., Hatano, K., Yoshikawa, M.: Adaptive web search based on user profile constructed without any effort from users. In: *Proceedings of WWW 2004*, pp. 675–684 (2004)
14. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: *Proceedings of WWW 7*, pp. 107–117 (1998)

# Measuring Incoherence in Description Logic-Based Ontologies

Guilin Qi<sup>1</sup> and Anthony Hunter<sup>2</sup>

<sup>1</sup>Institute AIFB  
Universität Karlsruhe  
D-76128 Karlsruhe, Germany  
[gqi@aifb.uni-karlsruhe.de](mailto:gqi@aifb.uni-karlsruhe.de)

<sup>2</sup>Department of Computer Science  
University College London, Gower Street  
London WC1E 6BT, UK  
[a.hunter@cs.ucl.ac.uk](mailto:a.hunter@cs.ucl.ac.uk)

**Abstract.** Ontologies play a core role in the success of the Semantic Web as they provide a shared vocabulary for different resources and applications. Developing an error-free ontology is a difficult task. A common kind of error for an ontology is logical contradiction or *incoherence*. In this paper, we propose some approaches to measuring incoherence in DL-based ontologies. These measures give an ontology engineer important information for maintaining and evaluating ontologies. We implement the proposed approaches using the KAON2 reasoner and provide some preliminary but encouraging empirical results.

## 1 Introduction

Ontologies play the core role for the success of the Semantic Web (SW) as they provide shared vocabularies for different domains. There are many representation languages for ontologies, such as Description Logics (DLs) [1]. High quality ontologies are important for SW technology. However, in practice, it is often difficult to construct an ontology which is error-free. A common error for an ontology is *incoherence*, i.e. whether there are *unsatisfiable* concepts which is interpreted as an empty set in all the *models* of its terminology. Incoherence can occur for several reasons, such as modeling errors when constructing an ontology and migration or merging of ontologies [16]. For example, when two upper ontologies SUMO and CYC are used in a single document, there exist over 1000 unsatisfiable concepts. Currently, there are many discussions on how to debug and diagnose terminologies in ontologies [9, 8, 17, 16]. Whilst incoherence in an ontology is often viewed negatively, by measuring it, we can get useful information for maintaining and evaluating an ontology. For example, by measuring the extent of incoherence of different ontologies, we can give a rank over them. That is, an ontology is more reliable than another one if it contains less incoherent information. Similarly, by measuring the extent of incoherence of different axioms, we get some ranking information on axioms which can be used to resolve

incoherence [8]. Furthermore, there is a trade-off in the amount of useful information in an ontology and the amount of coherent information. For example, in the extreme, we could guarantee a coherent ontology by only having the empty ontology. Obviously, this would not be acceptable, and so we need to tolerate the possibility of some incoherence in an ontology. Once we do tolerate this possibility, we need to keep track of it, perhaps as part of a process of improvement, or as a way of isolating the problematical parts of the ontology until we decide to how to fix those parts. So for these tasks, it is helpful to know where and by how much there is incoherence. This may include consideration of whether it is widespread, or localized. So the size and overlaps of incoherent subsets can be useful diagnostic tools.

Incoherence in ontologies corresponds to *inconsistency in knowledge bases* in classical logic, where a knowledge base is a finite set of classical formulae. A knowledge base is inconsistent if and only if there is no model satisfying all its formulae. We can regard the measures of inconsistency for a knowledge base as one of the following three types: **(Formula-centric measures)** These measures take into account the number of formulae required for inconsistency, and so fewer formulae in an inconsistency means a higher degree of inconsistency (e.g. [10]); **(Atom-centric measures)** These measures take into account the proportion of the language affected by inconsistency, and so more propositional atoms involved in inconsistency means a higher degree of inconsistency (e.g. [5,11]); and **(Conflict-centric measures)** These measures take into the account the number of conflicts each formula is involved in, and so if each formula is involved in more conflicts, there is a higher degree of inconsistency (e.g. [6]). Furthermore, given a measure of inconsistency of a knowledgebase, using one of the above, we can ascribe the blame or responsibility that each formula has in the set by drawing on an approach from game theory, called Shapley values, that allows for a principled assignment [7].

In this paper, we propose some approaches for measuring incoherence in DL-based ontologies that are *independent of specific DL languages*. They are based on the notion of scoring function introduced for classical logic in [6]. Whilst description logics are sub-systems of classical logic, they do incorporate a number of specialized features that are not in classical logic, and so in order to optimize the use of scoring functions for description logics, some refinement of the notion of scoring functions has been necessary. Indeed, this refinement has resulted in two classes of measures of incoherence: measures of incoherence for unsatisfiable concepts and measures of incoherence for terminologies. First, we define the scoring function for an unsatisfiable concept and use it to define a score ordering on unsatisfiable concepts. Second, we define the scoring function for a TBox and use it to define an ordering on terminology axioms and an ordering on TBoxes. We implement the proposed approaches using KAON2 reasoner<sup>1</sup> and report preliminary but encouraging experimental results.

The paper is organized as follows: We review DLs and related notions on incoherence (Section 2); We review related work (Section 3); We define our new

---

<sup>1</sup> c.f. <http://kaon2.semanticweb.org>

framework (Section 4); We discuss applications (Section 5); and We present preliminary empirical results (Section 6).

## 2 Preliminaries

### 2.1 Description Logics

We now give a brief introduction of Description Logics (DLs) and refer the reader to the DL handbook [1] for more details.

A DL-based ontology (or ontology)  $O = (\mathcal{T}, \mathcal{A})$  consists of a set  $\mathcal{T}$  of concept axioms (TBox) and role axioms, and a set  $\mathcal{A}$  of assertional axioms (ABox). Concept axioms have the form  $C \sqsubseteq D$  where  $C$  and  $D$  are (possibly complex) concept descriptions, and role axioms are expressions of the form  $R \sqsubseteq S$ , where  $R$  and  $S$  are (possibly complex) role descriptions. We call both concept axioms and role axioms as terminology axioms. The ABox contains *concept assertions* of the form  $C(a)$  where  $C$  is a concept and  $a$  is an individual name, and *role assertions* of the form  $R(a, b)$ , where  $R$  is a role and  $a$  and  $b$  are individual names.

The semantics of DLs is defined via a model-theoretic semantics, which explicates the relationship between the language syntax and the model of a domain: An interpretation  $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$  consists of a non-empty domain set  $\Delta^{\mathcal{I}}$  and an interpretation function  $\cdot^{\mathcal{I}}$ , which maps from individuals, concepts and roles to elements of the domain, subsets of the domain and binary relations on the domain, respectively.

Given an interpretation  $\mathcal{I}$ , we say that  $\mathcal{I}$  satisfies a concept axiom  $C \sqsubseteq D$  (resp., a role inclusion axiom  $R \sqsubseteq S$ ) if  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$  (resp.,  $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ ). Furthermore,  $\mathcal{I}$  satisfies a concept assertion  $C(a)$  (resp., a role assertion  $R(a, b)$ ) if  $a^{\mathcal{I}} \in C^{\mathcal{I}}$  (resp.,  $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$ ). An interpretation  $\mathcal{I}$  is called a *model* of an ontology  $O$ , iff it satisfies each axiom in  $O$ .

### 2.2 Incoherence in DL-Based Ontologies

We introduce the notion of incoherence in DL-based ontologies defined in [4].

**Definition 1 (Unsatisfiable Concept).** *A concept name  $C$  in an ontology  $O$ , is unsatisfiable iff, for each interpretation  $\mathcal{I}$  of  $O$ ,  $C^{\mathcal{I}} = \emptyset$ . The set of all unsatisfiable concept is denoted as  $US(O)$ .*

That is, a concept name is unsatisfiable in an ontology iff it is interpreted as an empty set by all models of  $O$ .

**Definition 2 (Incoherent Ontology).** *An ontology  $O$  is incoherent iff there exists an unsatisfiable concept name in  $O$ .*

For example, an ontology  $O = \{A \sqsubseteq B, A \sqsubseteq \neg B\}$  is incoherent because  $A$  is unsatisfiable in  $O$ . As pointed out in [4], incoherence does not provide the classical sense of the inconsistency because there might exist a model for an incoherent ontology. We first introduce the definition of an inconsistent ontology.

**Definition 3 (Inconsistent Ontology).** *An ontology  $O$  is inconsistent iff it has no model.*

However, incoherence and inconsistency are related to each other. According to the discussion in [4], incoherence is a potential cause of inconsistency. That is, suppose  $C$  is an unsatisfiable concept in  $O$ , by adding an instance  $a$  to  $C$  will result in an inconsistent ontology. For example, the ontology  $O = \{A \sqsubseteq B, A \sqsubseteq \neg B\}$  is incoherent but consistent (any interpretation which interprets  $A$  as an empty set and  $B$  as a nonempty set is a model of  $O$ ). However,  $O' = \{A(a), A \sqsubseteq B, A \sqsubseteq \neg B\}$  is both incoherent and inconsistent.

In most of current work on debugging ontologies, the incoherence problem is often discussed at the terminology level. That is, ABoxes are usually considered as irrelevant for incoherence. Therefore, when we talk about an axiom in an ontology, we mean only the terminology axiom.

In the following, we introduce some definitions which are useful to explain logical incoherence.

**Definition 4.** [17] *Let  $A$  be a concept name which is unsatisfiable in a TBox  $\mathcal{T}$ . A set  $\mathcal{T}' \subseteq \mathcal{T}$  is a minimal unsatisfiability-preserving sub-TBox (MUPS) of  $\mathcal{T}$  if  $A$  is unsatisfiable in  $\mathcal{T}'$ , and  $A$  is satisfiable in every sub-TBox  $\mathcal{T}'' \subset \mathcal{T}'$ . The set of all MUPS of  $\mathcal{T}$  with respect to  $A$  is denoted as  $MU_A(\mathcal{T})$ .*

A MUPS of  $\mathcal{T}$  and  $A$  is the minimal sub-TBox of  $\mathcal{T}$  in which  $A$  is unsatisfiable. For example, given an TBox  $\mathcal{T} = \{C \sqsubseteq A, A \sqsubseteq B, A \sqsubseteq \neg B\}$ .  $C$  is an unsatisfiable concept and it has one MUPS, i.e.,  $\mathcal{T}$ . Based on MUPS, we can classify unsatisfiable concepts into derived unsatisfiable concepts and root unsatisfiable concepts as follows:

**Definition 5 (Root and Derived).** [9]  *$C$  is a derived unsatisfiable concept in  $\mathcal{T}$  iff it satisfies the following condition:  $\exists i, j$  such that  $MUPS_i(C) \supseteq MUPS_j(D)$ , for an unsatisfiable concept  $D$ , where  $MUPS_i(C) \in MU_C(\mathcal{T})$  and  $MUPS_i(D) \in MU_D(\mathcal{T})$ . If  $C$  does not satisfy this condition then it is a root unsatisfiable concept.*

**Definition 6.** [17] *Let  $\mathcal{T}$  be an incoherent TBox. A TBox  $\mathcal{T}' \subseteq \mathcal{T}$  is a minimal incoherence-preserving sub-TBox (MIPS) of  $\mathcal{T}$  if  $\mathcal{T}'$  is incoherent, and every sub-TBox  $\mathcal{T}'' \subset \mathcal{T}'$  is coherent. The set of all MIPSs of  $\mathcal{T}$  is denoted as  $MI(\mathcal{T})$ .*

A MIPS of  $\mathcal{T}$  is the minimal sub-TBox of  $\mathcal{T}$  which is incoherent. Let us consider the example used to illustrate Definition 4, in which there is only one MIPS of  $\mathcal{T}$ :  $\{A \sqsubseteq B, A \sqsubseteq \neg B\}$ . We say a terminology axiom is *in conflict* in  $\mathcal{T}$  if there exists a MIPS of  $\mathcal{T}$  containing it.

Algorithms have been given to calculate MUPS and MIPS in a given ontology (see [17, 9]). It was shown in [17] that calculating MUPS is the same as the satisfiability check if a glass-box algorithm is used, i.e., the algorithm is based on the description logic tableaux reasoner.



### 3 Related Work

This work is related to the work on debugging terminologies and resolve incoherence [9,8,17,16]. The first work on debugging erroneous terminologies is reported in [17] where the authors provide a specialized algorithm for the DL  $\mathcal{ALC}$ . The notions of MUPS and MIPS are introduced to explain logical incoherences there. In [9], two orthogonal debugging approaches are proposed to detect the clash/sets of support axioms responsible for an unsatisfiable classes, and to identify root/derived unsatisfiable classes. Based on the debugging approach, in [8], the authors give a tool to repair unsatisfiable concepts in OWL ontologies. The basic idea is to rank erroneous axioms and then to generate a plan to resolve the errors in a given set of unsatisfiable concepts by taking into account the axiom ranks. Although the above-mentioned work provides potential starting points for measuring incoherence, they are not explicitly used for this purpose. The approaches to measuring incoherence in ontologies are still underdeveloped.

Recently, some approaches for measuring inconsistency in an ontology have been proposed [3,12]. In [3], some inconsistency measures are proposed by adapting the approach based on Shapley values in [7]. The approach given in [12] is defined by a four-valued semantics of DL  $\mathcal{ALC}$ . Our approaches differ from these approaches in that we consider measuring incoherence of an ontology instead of inconsistency, although the measures of incoherence implicitly provides information for degree of inconsistency of an ontology (recall that inconsistency and incoherence are two different but related notions in DLs).

## 4 Measures of Incoherence

### 4.1 Measures of Incoherence for Unsatisfiable Concepts

If an ontology is incoherent, there is at least one unsatisfiable concept in its TBox. For these unsatisfiable concepts, some are more problematic than others. For example, given a TBox  $\mathcal{T} = \{A \sqsubseteq B, B \sqsubseteq C, B \sqsubseteq \neg C\}$ .  $A$  and  $B$  are both unsatisfiable concepts. However,  $A$  is unsatisfiable because of  $B$ . That is, if  $B$  becomes satisfiable, then  $A$  is also satisfiable. So in a sense, we may regard  $B$  as more incoherent than  $A$ . However, we may argue that  $B$  is less incoherent than  $A$  because the axioms involved in the conflict for concept  $B$  are a subset of those for concept  $A$ . We develop an alternative (conflict-centric) characterization here.

We define an ordering between two unsatisfiable concepts based on the *scoring function*.

**Definition 7.** Let  $\mathcal{T}$  be an incoherent TBox, and  $A$  be an unsatisfiable concept name in  $\mathcal{T}$  and  $MU_A(\mathcal{T})$  be the set of all MUPSs of  $\mathcal{T}$  with respect to  $A$ . The scoring function for  $A$  is a function  $S_{\mathcal{T},A} : \wp(\mathcal{T}) \mapsto \mathbb{N}$  ( $\wp(\mathcal{T})$  denoting the power set of  $\mathcal{T}$ ) such that for all  $T' \in \wp(\mathcal{T})$

$$S_{\mathcal{T},A}(T') = |\{T_i \in MU_A(\mathcal{T}) : T_i \cap T' \neq \emptyset\}|.$$

The scoring function  $S_{T,A}$  for  $A$  returns for each subset  $T'$  of  $T$  the number of MUPS of  $T$  with respect to  $A$  that have overlap with  $T'$ . The scoring function adapts a version defined in [6] to compare two logical inconsistent sets of propositional formulae. It is clear that we have the following proposition.

**Proposition 1.** *Let  $T$  be an incoherent TBox, and  $A$  be an unsatisfiable concept names in  $T$  and  $MU_A(T)$  be the set of all MUPSs of  $T$  with respect to  $A$ . Suppose  $S_{T,A}$  is the scoring function for  $A$ , then for all  $T' \in \wp(T)$*

$$S_{T,A}(T') = |MU_A(T)| - |MU_A(T \setminus T')|.$$

According to Proposition 1, the scoring function for  $T'$  gives the number of MUPS that would be eliminated from  $T$  if  $T'$  were subtracted from  $T$ .

Let  $T$  be an incoherent TBox, and  $A$  and  $B$  be two unsatisfiable concept names in  $T$ . Let  $M_A = \cup_{T_i \in MU_A(T)} T_i$  and  $M_B = \cup_{T_j \in MU_B(T)} T_j$ . Suppose  $|M_A| < |M_B|$ , then we add some dummy axioms to  $M_A$  such that  $|M_A| = |M_B|$ . The dummy axioms can be constructed by using fresh concept names (i.e. names not used elsewhere). We can define a score ordering as follows:

**Definition 8.** *Assume that  $S_{T,A}$  and  $S_{T,B}$  are the scoring functions for  $A$  and  $B$  respectively.  $S_{T,A} \leq_S S_{T,B}$  iff there is a bijection  $f : \wp(M_A) \rightarrow \wp(M_B)$  such that the following condition is satisfied:*

$$\forall T' \in \wp(M_A), S_{T,A}(T') \leq S_{T,B}(f(T')).$$

As usual,  $S_{T,A} <_S S_{T,B}$  denotes  $S_{T,A} \leq_S S_{T,B}$  and  $S_{T,B} \not\leq_S S_{T,A}$ , and  $S_{T,A} \simeq_S S_{T,B}$  denotes  $S_{T,A} \leq_S S_{T,B}$  and  $S_{T,B} \leq_S S_{T,A}$ . The score ordering, denoted  $\leq$ , is defined as follows: for any two unsatisfiable concepts  $A$  and  $B$ ,

$$A \leq B \text{ iff } S_{T,A} \leq_S S_{T,B}.$$

Intuitively,  $S_{T,A} \leq_S S_{T,B}$  means that the MUPSs in  $MU_A(T)$  are less overlapping than those in  $MU_B(T)$ , and hence each formula in the first set is involved in the same or fewer conflicts than its corresponding formula in the second. So  $A$  is less incoherent than  $B$  with respect to the score ordering iff the MUPSs in  $A$  is less overlapping than those in  $B$ .

We illustrate the score ordering by the following example.

*Example 1.* Given a TBox  $T = \{A \sqsubseteq B, A \sqsubseteq C, B \sqsubseteq D, C \sqsubseteq \neg D, E \sqsubseteq F, E \sqsubseteq \neg F, F \sqsubseteq D, E \sqsubseteq \neg D\}$ , where  $A, B, C, D, E, F$  are concept names. Clearly,  $A$  and  $E$  are two root unsatisfiable concepts in  $T$ , and  $MU_A = \{T_1\}$ , s.t.  $T_1 = \{A \sqsubseteq B, A \sqsubseteq C, B \sqsubseteq D, C \sqsubseteq \neg D\}$  and  $MU_E = \{T_2, T_3\}$ , s.t.  $T_2 = \{E \sqsubseteq F, E \sqsubseteq \neg F\}$  and  $T_3 = \{E \sqsubseteq F, F \sqsubseteq D, E \sqsubseteq \neg D\}$ . So  $M_A = \{A \sqsubseteq B, A \sqsubseteq C, B \sqsubseteq D, C \sqsubseteq \neg D\}$  and  $M_E = \{E \sqsubseteq F, E \sqsubseteq \neg F, F \sqsubseteq D, E \sqsubseteq \neg D\}$ . Let  $S_{T,A}$  and  $S_{T,E}$  be the scoring function for  $A$  and  $E$  respectively, then  $S_{T,A}(T') = 1$ , for all  $T' \in \wp(M_A)$ . However,  $S_{T,E}(\{E \sqsubseteq F\}) = 2$  and  $S_{T,E}(T') \geq 1$  for all other  $T' \in \wp(M_E)$ . So  $S_{T,A} <_S S_{T,E}$  and we have  $A < E$ .

When defining the score ordering, we need to find a bijection  $f$  mapping every subset of  $M_A$  to a subset of  $M_B$ . In the following, we provide a procedure to find the bijection  $f$ . Let  $|\wp(M_A)| = |\wp(M_B)| = n$ .

Step 1: for each  $\mathcal{T}_i \in \wp(M_A)$  and each  $\mathcal{T}'_j \in \wp(M_B)$ , compute  $S_{\mathcal{T},A}(\mathcal{T}_i)$  and  $S_{\mathcal{T},B}(\mathcal{T}'_j)$ ,

Step 2: rearrange  $\mathcal{T}_1, \dots, \mathcal{T}_n$  as  $\mathcal{T}_{i_1}, \dots, \mathcal{T}_{i_n}$  ( $i_k \in \{1, \dots, n\}$ ) such that  $S_{\mathcal{T},A}(\mathcal{T}_{i_1}) \geq S_{\mathcal{T},A}(\mathcal{T}_{i_2}) \geq \dots \geq S_{\mathcal{T},A}(\mathcal{T}_{i_n})$ , and rearrange  $\mathcal{T}'_1, \dots, \mathcal{T}'_n$  as  $\mathcal{T}'_{j_1}, \dots, \mathcal{T}'_{j_n}$  ( $j_k \in \{1, \dots, n\}$ ) such that  $S_{\mathcal{T},B}(\mathcal{T}'_{j_1}) \geq S_{\mathcal{T},B}(\mathcal{T}'_{j_2}) \geq \dots \geq S_{\mathcal{T},B}(\mathcal{T}'_{j_n})$ ,

Step 3: a mapping  $f_S : \wp(M_A) \rightarrow \wp(M_B)$  is defined as follows: for each  $\mathcal{T}_{i_k} \in \wp(M_A)$ ,  $f_S(\mathcal{T}_{i_k}) = \mathcal{T}'_{j_k}$ .

It is clear that  $f_S$  is a bijection. The following proposition shows that  $f_S$  is the bijection which is used to define the score ordering.

**Proposition 2.** *Assume that  $S_{\mathcal{T},A}$  and  $S_{\mathcal{T},B}$  are the scoring functions for  $A$  and  $B$  respectively. Then  $S_{\mathcal{T},A} \leq_S S_{\mathcal{T},B}$  iff*

$$\forall \mathcal{T}' \in \wp(M_A), S_{\mathcal{T},A}(\mathcal{T}') \leq S_{\mathcal{T},B}(f_S(\mathcal{T}')).$$

**Proof:** “If” part is clear by the definition of score ordering. We show the “only if” part.

Suppose  $S_{\mathcal{T},A} \leq_S S_{\mathcal{T},B}$ , then there exists a bijection  $f$  s.t. for all  $\mathcal{T}' \in \wp(M_A)$ ,  $S_{\mathcal{T},A}(\mathcal{T}') \leq S_{\mathcal{T},B}(f(\mathcal{T}'))$ . We shown that  $S_{\mathcal{T},A}(\mathcal{T}_{i_k}) \leq S_{\mathcal{T},B}(\mathcal{T}'_{j_k})$  for all  $k = 1, \dots, n$  by induction over the index  $k$ .

Suppose  $k = 1$ . Then  $S_{\mathcal{T},A}(\mathcal{T}_{i_1}) \leq S_{\mathcal{T},B}(f(\mathcal{T}_{i_1})) \leq S_{\mathcal{T},B}(\mathcal{T}'_{j_1})$ .

Assume that  $S_{\mathcal{T},A}(\mathcal{T}_{i_k}) \leq S_{\mathcal{T},B}(\mathcal{T}'_{j_k})$  for all  $k < m$ . Suppose that  $S_{\mathcal{T},A}(\mathcal{T}_{i_m}) > S_{\mathcal{T},B}(\mathcal{T}'_{j_m})$ . Then  $S_{\mathcal{T},B}(\mathcal{T}'_{j_m}) < S_{\mathcal{T},B}(f(\mathcal{T}_{i_m}))$ . This means that there exists  $j_1 < j_m$  such that  $f(\mathcal{T}_{i_m}) = \mathcal{T}'_{j_1}$ . However, since  $S_{\mathcal{T},A}(\mathcal{T}_{i_m}) > S_{\mathcal{T},B}(\mathcal{T}'_{j_m})$ , we have that  $S_{\mathcal{T},A}(\mathcal{T}_{i_k}) > S_{\mathcal{T},B}(\mathcal{T}'_{j_m})$  for all  $k < m$ . Therefore, for any  $k < m$ , there exists  $k' < m$  such that  $f(\mathcal{T}_{i_k}) = \mathcal{T}'_{j_{k'}}$ . Therefore, it is impossible that there exists  $j_1 < j_m$  such that  $f(\mathcal{T}_{i_m}) = \mathcal{T}'_{j_1}$  (every such  $\mathcal{T}'_{j_1}$  corresponds to a  $\mathcal{T}_{i_k}$  with  $k < m$ ). This is a contradiction. So  $S_{\mathcal{T},A}(\mathcal{T}_{i_m}) \leq S_{\mathcal{T},B}(\mathcal{T}'_{j_m})$ .

**Proposition 3.** *Let  $\mathcal{T}$  be an incoherent TBox, and let  $A$  and  $B$  be two unsatisfiable concepts in it. If  $A \sqsubseteq B \in \mathcal{T}$ , then  $B \leq_S A$ .*

The proof of Proposition 3 is easy to establish. Proposition 3 tells us that if  $A$  is subsumed by  $B$  then  $A$  is more incoherent than  $B$  with respect to the score ordering. If we consider the example in the beginning of this section,  $B$  is more incoherent than  $A$  with respect to the score ordering. Therefore, our score ordering provides a different view on the extent of incoherence of a concept from the subsumption relation. Indeed, the scoring ordering gives a conflict-centric view since the axioms involved in the conflict for concept  $B$  are a subset of those for concept  $A$ . Proposition 3 also provides us with a way to improve the performance of our approach for generating the score ordering. That is, before comparing the score functions of two unsatisfiable concepts, we can first check if they have a subsumption relation in the ontology.

## 4.2 Measures of Incoherence for Terminologies

Given a TBox which may be incoherent, we propose three approaches to measuring its degree of incoherence. The first measure is defined by the ratio of the number of unsatisfiable concepts and that of all the concepts in  $\mathcal{T}$ .

**Definition 9.** Let  $\mathcal{T}$  be a TBox. Suppose  $Con(\mathcal{T})$  is the set of all concept names and  $US(\mathcal{T})$  be the set of all unsatisfiable concept names in  $\mathcal{T}$  respectively, the unsatisfiability ratio for  $\mathcal{T}$ , denoted  $d_{UR}$ , is defined as follows:

$$d_{UR}(\mathcal{T}) = \frac{|US(\mathcal{T})|}{|Con(\mathcal{T})|}.$$

The unsatisfiability ratio gives us a simple view on the incoherence of a TBox. That is, if most of concept names are unsatisfiable in a TBox, the TBox is problematic. However, the unsatisfiability ratio is misleading in some cases. For example, in an ontology such as Tambis<sup>2</sup> where there is a large number of unsatisfiable concept names, many of the unsatisfiable concept names depend on other unsatisfiable concept names. The root unsatisfiable concept names are relatively few (according to [9], in Tambis, 33 concepts names out of 144 unsatisfiable concept names are root unsatisfiable concept names) and by repairing these concept names we can get a coherent ontology. Therefore, this ontology is not “strongly” incoherent. To overcome the problem for the unsatisfiability ratio, we can consider only the root unsatisfiable concept names.

**Definition 10.** Let  $\mathcal{T}$  be a TBox. Suppose  $Con(\mathcal{T})$  is the set of all concept names and  $RU(\mathcal{T})$  be the set of all root unsatisfiable concept names in  $\mathcal{T}$  respectively, the refined unsatisfiability ratio for  $\mathcal{T}$ , denoted  $d_{RU}$ , is defined as follows:

$$d_{RU}(\mathcal{T}) = \frac{|RU(\mathcal{T})|}{|Con(\mathcal{T})|}.$$

Let us consider Example 1 again. The set of concept names are  $\{A, B, C, D, E, F\}$  and root unsatisfiable concept names are  $A$  and  $E$ , so  $d_{RU}(\mathcal{T}) = \frac{2}{6}$ .

Both the unsatisfiability ratio and the refined unsatisfiability ratio do not consider the number of terminology axioms that are in conflict. So we define another incoherence measure for TBoxes.

**Definition 11.** Let  $\mathcal{T}$  be a TBox. Suppose  $MI(\mathcal{T})$  is the set of all MIPSs of  $\mathcal{T}$ , then the incoherence ratio for  $\mathcal{T}$ , denoted  $d_{IR}$ , is defined as follows:

$$d_{IR}(\mathcal{T}) = \frac{|\cup_{\mathcal{T}_i \in MI(\mathcal{T})} \mathcal{T}_i|}{|\mathcal{T}|}.$$

The incoherence ratio measures the percentage of axioms in a TBox that are in conflict. It differentiates the root unsatisfiable concept names and derived unsatisfiable concept names. This is because any axiom whose left hand is a derived unsatisfiable concept name is not in an MIPS.

<sup>2</sup> <http://protege.cim3.net/file/pub/ontologies/tambis/tambis-full.owl>

*Example 2.* Let  $\mathcal{T} = \{A \sqsubseteq B, A \sqsubseteq \neg B, C \sqsubseteq A\}$  and  $\mathcal{T}' = \{A \sqsubseteq B, A \sqsubseteq \neg B, C \sqsubseteq \perp\}$ . Then  $US(\mathcal{T}) = US(\mathcal{T}') = \{A, C\}$  and  $d_{UR}(\mathcal{T}') = d_{UR}(\mathcal{T}) = \frac{2}{3}$ :  $\mathcal{T}$  and  $\mathcal{T}'$  have the same unsatisfiability ratio. However,  $MI(\mathcal{T}) = \{\{A \sqsubseteq B, A \sqsubseteq \neg B\}\}$  and  $MI(\mathcal{T}') = \{\mathcal{T}'\}$ . So  $d_{IR}(\mathcal{T}) = \frac{2}{3}$  and  $d_{IR}(\mathcal{T}') = 1$ .

The problem for the incoherence ratio is that it says nothing about the extent to which the MIPSs in  $MI(\mathcal{T})$  overlap.

*Example 3.* Let  $\mathcal{T} = \{A \sqsubseteq B, A \sqsubseteq \neg B, C \sqsubseteq D, C \sqsubseteq \neg D\}$  and  $\mathcal{T}' = \{A \sqsubseteq B, A \sqsubseteq \neg B, B \sqsubseteq C, A \sqsubseteq \neg C\}$  be two coherent TBoxes, where  $A, B, C, D$  are concept names. By Definition 6,  $\mathcal{T}$  has two MIPSs  $\{A \sqsubseteq B, A \sqsubseteq \neg B\}$  and  $\{C \sqsubseteq D, C \sqsubseteq \neg D\}$ , and  $\mathcal{T}'$  has two MIPSs  $\{A \sqsubseteq B, A \sqsubseteq \neg B\}$  and  $\{A \sqsubseteq B, B \sqsubseteq C, A \sqsubseteq \neg C\}$ . According to Definition 11, we have  $d_{IR}(\mathcal{T}) = d_{IR}(\mathcal{T}') = 1$ . However, MIPSs in  $MI(\mathcal{T})$  have no overlap whilst the MIPSs in  $MI(\mathcal{T}')$  have a common axiom  $A \sqsubseteq B$ . Therefore, we may conclude that  $\mathcal{T}$  is less coherent than  $\mathcal{T}'$ .

We have defined two measures and argue that they are not fine grained enough. Next, we define an incoherence measure for TBoxes which is based on the scoring functions.

**Definition 12.** Let  $\mathcal{T}$  be a TBox. The scoring function for  $\mathcal{T}$  is a function  $S_{\mathcal{T}} : \wp(\mathcal{T}) \mapsto N$  such that for all  $\mathcal{T}' \in \wp(\mathcal{T})$

$$S_{\mathcal{T}}(\mathcal{T}') = |\{\mathcal{T}_i \in MI(\mathcal{T}) : \mathcal{T}_i \cap \mathcal{T}' \neq \emptyset\}|.$$

The scoring function  $S_{\mathcal{T}}$  for  $\mathcal{T}$  returns for each subset  $\mathcal{T}'$  of  $\mathcal{T}$  the number of MIPSs of  $\mathcal{T}$  that have an overlap with  $\mathcal{T}'$ .

We have the following proposition for the scoring function.

**Proposition 4.** Let  $S_{\mathcal{T}}$  be the scoring function for  $\mathcal{T}$ . For  $\mathcal{T}_i, \mathcal{T}_j \in \wp(\mathcal{T})$ , we have  $S_{\mathcal{T}}(\mathcal{T}_i \cap \mathcal{T}_j) \leq \min(S_{\mathcal{T}}(\mathcal{T}_i), S_{\mathcal{T}}(\mathcal{T}_j))$  and  $\max(S_{\mathcal{T}}(\mathcal{T}_i), S_{\mathcal{T}}(\mathcal{T}_j)) \leq S_{\mathcal{T}}(\mathcal{T}_i \cup \mathcal{T}_j)$ .

The scoring function can be used to define an ordering between two terminology axioms.

**Definition 13.** Let  $\mathcal{T}$  be a TBox and  $S_{\mathcal{T}}$  be its scoring function. A score-based ordering on terminology axioms in  $\mathcal{T}$ , denoted  $\prec_{S_{\mathcal{T}}}$ , is defined as follows: for any  $\phi, \psi \in \mathcal{T}$ ,

$$\phi \preceq_{S_{\mathcal{T}}} \psi \text{ iff } S_{\mathcal{T}}(\{\phi\}) \leq S_{\mathcal{T}}(\{\psi\}).$$

As usual,  $\phi \prec_{S_{\mathcal{T}}} \psi$  denotes  $\phi \preceq_{S_{\mathcal{T}}} \psi$  and  $\psi \not\preceq_{S_{\mathcal{T}}} \phi$ .  $\phi \preceq_{S_{\mathcal{T}}} \psi$  means that  $\phi$  is less incoherent than  $\psi$  with respect to the scoring function. That is,  $\phi$  is contained in less MIPSs of  $\mathcal{T}$  than  $\psi$ . It is clear that  $\preceq_{S_{\mathcal{T}}}$  is a total pre-order, i.e. a pre-order which is complete.

*Example 4.* (Example 1 Continued) There are three MIPSs of  $\mathcal{T}$ :  $\mathcal{T}_1 = \{A \sqsubseteq B, A \sqsubseteq C, B \sqsubseteq D, C \sqsubseteq \neg D\}$ ,  $\mathcal{T}_2 = \{E \sqsubseteq F, E \sqsubseteq \neg F\}$  and  $\mathcal{T}_3 = \{E \sqsubseteq F, F \sqsubseteq D, E \sqsubseteq \neg D\}$ . So  $S_{\mathcal{T}}(\{E \sqsubseteq F\}) = 2$  and  $S_{\mathcal{T}}(\{\phi\}) = 1$  for all other  $\phi \in \mathcal{T}$ . Therefore,  $\phi \prec_{S_{\mathcal{T}}} E \sqsubseteq F$  for all  $\phi \in \mathcal{T}$  and  $\phi \neq E \sqsubseteq F$ .

Let  $\mathcal{T}$  and  $\mathcal{T}'$  be two TBox. Let  $M_{\mathcal{T}} = \cup_{\mathcal{T}_i \in MI(\mathcal{T})} \mathcal{T}_i$  and  $M_{\mathcal{T}'} = \cup_{\mathcal{T}_j \in MI(\mathcal{T}')} \mathcal{T}_j$ . Suppose  $|M_{\mathcal{T}}| < |M_{\mathcal{T}'}|$ , then we add some dummy axioms to  $M_{\mathcal{T}}$  such that  $|M_{\mathcal{T}}| = |M_{\mathcal{T}'}|$ . An ordering on TBoxes can be defined by the scoring functions as follows.

**Definition 14.** Assume that  $S_{\mathcal{T}}$  and  $S_{\mathcal{T}'}$  are the scoring functions for two TBoxes  $\mathcal{T}$  and  $\mathcal{T}'$  respectively.  $S_{\mathcal{T}} \preceq S_{\mathcal{T}'}$  iff there is a bijection  $f : \wp(M_{\mathcal{T}}) \rightarrow \wp(M_{\mathcal{T}'})$  such that the following condition is satisfied:

$$\forall T' \in \wp(M_1), S_{\mathcal{T}}(T') \leq S_{\mathcal{T}'}(f(T')).$$

As usual,  $S_{\mathcal{T}} \prec_S S_{\mathcal{T}'}$  denotes  $S_{\mathcal{T}} \preceq_S S_{\mathcal{T}'}$  and  $S_{\mathcal{T}'} \not\prec_S S_{\mathcal{T}}$ , and  $S_{\mathcal{T}} \equiv_S S_{\mathcal{T}'}$  denotes  $S_{\mathcal{T}} \preceq_S S_{\mathcal{T}'}$  and  $S_{\mathcal{T}'} \preceq_S S_{\mathcal{T}}$ . The score ordering, denoted  $\preceq_S$ , is defined as: for any two TBoxes  $\mathcal{T}$  and  $\mathcal{T}'$ ,

$$\mathcal{T} \preceq_S \mathcal{T}' \text{ iff } S_{\mathcal{T}} \preceq_S S_{\mathcal{T}'}$$

Intuitively,  $S_{\mathcal{T}} \preceq S_{\mathcal{T}'}$  means that the MIPSs of  $\mathcal{T}$  are less overlapping than those of  $\mathcal{T}'$ . So  $\mathcal{T}$  is less incoherent than  $\mathcal{T}'$  with respect to the score ordering iff the MIPSs of  $\mathcal{T}$  is less overlapping than those of  $\mathcal{T}'$ .

*Example 5.* Given two TBoxes  $\mathcal{T} = \{A \sqsubseteq B \sqcap C, B \sqcap C \sqsubseteq \perp\}$  and  $\mathcal{T}' = \{A \sqsubseteq B, A \sqsubseteq C, B \sqcap C \sqsubseteq \perp\}$ , it is clear that  $\mathcal{T} \equiv \mathcal{T}'$ .  $\mathcal{T}$  has only one MIPS which is  $\mathcal{T}$  and  $\mathcal{T}'$  has only one MIPS which is  $\mathcal{T}'$ . So  $M_{\mathcal{T}} = \mathcal{T}$  and  $M_{\mathcal{T}'} = \mathcal{T}'$ . Since  $|M_{\mathcal{T}}| < |M_{\mathcal{T}'}|$ , we add a dummy axiom to  $M_{\mathcal{T}}$  such that  $M_{\mathcal{T}} = \{A \sqsubseteq B \sqcap C, B \sqcap C \sqsubseteq \perp, D \sqsubseteq \top\}$ , where  $D$  is a new concept name. Let  $S_{\mathcal{T}}$  and  $S_{\mathcal{T}'}$  be the scoring functions for  $\mathcal{T}$  and  $\mathcal{T}'$  respectively, we then have

$$\begin{aligned} S_{\mathcal{T}}(\{A \sqsubseteq B \sqcap C\}) &= 1, S_{\mathcal{T}}(\{B \sqcap C \sqsubseteq \perp\}) = 1, S_{\mathcal{T}}(\{D \sqsubseteq \top\}) = 0, \text{ and} \\ S_{\mathcal{T}'}(\{A \sqsubseteq B\}) &= 1, S_{\mathcal{T}'}(\{A \sqsubseteq C\}) = 1, S_{\mathcal{T}'}(\{B \sqcap C \sqsubseteq \perp\}) = 1. \end{aligned}$$

So  $S_{\mathcal{T}} \prec S_{\mathcal{T}'}$  and  $\mathcal{T} \prec_S \mathcal{T}'$ .

According to Example 5, the scoring function defined by Definition 12 is syntax sensitive in the sense that there may exist two TBoxes  $\mathcal{T}$  and  $\mathcal{T}'$  where  $\mathcal{T} \equiv \mathcal{T}'$  and  $S_{\mathcal{T}}$  is the scoring function for  $\mathcal{T}$  and  $S_{\mathcal{T}'}$  is the scoring function for  $\mathcal{T}'$ , but  $S_{\mathcal{T}} \neq S_{\mathcal{T}'}$ . To give a more precise measure of incoherence, we can simply split the axioms in a TBox  $\mathcal{T}$  into “smaller” axioms to obtain an equivalent TBox  $\mathcal{T}_s$  using the algorithm in 8. For instance, in Example 5, the axiom  $A \sqsubseteq B \sqcap C$  can be split into  $A \sqsubseteq B$  and  $A \sqsubseteq C$ . Then it is clear that  $S_{\mathcal{T}} \equiv_S S_{\mathcal{T}'}$ .

The score ordering  $\preceq_S$  is related to the incoherence ratio.

**Proposition 5.** Let  $\mathcal{T}$  and  $\mathcal{T}'$  be two TBoxes, and  $|\mathcal{T}| = |\mathcal{T}'|$ . Suppose  $S_{\mathcal{T}}$  and  $S_{\mathcal{T}'}$  are scoring function for  $\mathcal{T}$  and  $\mathcal{T}'$  respectively, then  $S_{\mathcal{T}} \preceq_S S_{\mathcal{T}'}$  implies  $d_{IR}(\mathcal{T}) \leq d_{IR}(\mathcal{T}')$ . The converse does not hold.

Proposition 5 shows that if two TBoxes have the same cardinality, then a TBox  $\mathcal{T}$  is less coherent than another one  $\mathcal{T}'$  implies that  $\mathcal{T}$  contains fewer conflicting terminology axioms.

## 5 Applications of Measures of Incoherence

The ordering relations on unsatisfiable concepts can provide important information for repairing incoherence in ontologies. When resolving incoherence in an ontology, we may need to repair unsatisfiable concepts one by one [8]. In this case, the ordering on unsatisfiable concepts can be used to decide which unsatisfiable concepts should be dealt with first. Another strategy to resolve incoherence is to remove some unsatisfiable concepts in an ontology [8]. Based on our ordering relation  $\leq_I$ , those unsatisfiable concepts that are more incoherent *w.r.t*  $\leq_I$  should be removed before those that are less incoherent.

Based on the score ordering on the TBoxes, we can give an ordering on ontologies. That is, an ontology  $O$  is more important or reliable than another one  $O'$  if it is less coherent than  $O'$ . The ordering on ontologies provides us with important information for evaluating different ontologies.

We have applied the scoring function to define an ordering between two axioms in an ontology. Alternatively, we can compute the Shapley values of each axiom and then obtain an ordering on axioms. The ordering on axioms are important context information for dealing with incoherence and inconsistency in an ontology. It has been widely accepted that priorities play an important role in dealing with inconsistency in propositional logic. Recently, several priority-based approaches for reasoning with inconsistent ontologies have been proposed [13,14,15]. A challenging problem for these approaches is to obtain an ordering on axioms. This problem can be solved by considering our score ordering.

## 6 Evaluation

We have implemented the approaches for measuring incoherence described in the previous sections in JAVA using KAON2. All tests were performed on a laptop computer with a 1.5 GHz Intel processor, 512M of RAM (with 512M heap space allocated to JVM). The operating system was Windows XP Pro SP2. The data sets were miniTambis (we do not use Tambis because KAON2 does not support it) and a revised version of Chemical (we denoted it by Chemical\*) provided by the University of Maryland [3], and *proton\_100\_all* [4]. Some information about the ontologies is given in Table 1. The second column of the table shows the number of terminology axioms, MIPSs and the cardinality of the union of all MIPSs and the third column of the table shows the number of concepts/properties/individuals and unsatisfiable concepts in the ontology. To find all the MUPSs of an unsatisfiable concept, we implemented an approach which is based on the algorithm for finding minimal unsatisfiable subsets in [2]. We then obtained all the MIPSs of an ontology using the approach given in [17].

---

<sup>3</sup> <http://www.mindswap.org/2005/debugging/ontologies/>

<sup>4</sup> <http://wasp.cs.vu.nl/knowledgeweb/d2163/learning.html>

**Table 1.** Sample OWL Data

Ontology	#axiom/#MIPS/#MIPUnion	C/P/I/U
miniTambis	173/3/12	178/35/0/30
Chemical*	123/4/8	48/19/0/35
<i>proton_100_all</i>	1788/3/10	266/111/30/3

**Table 2.** Score ordering on unsatisfiable concepts (time is in seconds)

Ontology	Prepare time	Compare time	Total time
miniTambis	233.230	78.540	311.770
Chemical*	873.12	652.75	1525.87
<i>proton_100_all</i>	319.83	0.071	319.901

### 6.1 Score Ordering on Unsatisfiable Concepts

We implemented the approach for score ordering on unsatisfiable concepts. The performance of our approach is analyzed in Table 2. In Table 2, the second column and third column show the average runtime (in seconds) of calculating MUPS of each unsatisfiable concept (prepare time) and average runtime of generating the score ordering on unsatisfiable concepts (compare time) respectively. The fourth column is the sum of the prepare time and compare time.

For all test ontologies, the time spent on comparing concepts is less than that spent on calculating MUPS. For ontology *proton\_100\_all*, which has only three unsatisfiable concepts and the maximal cardinality of their MUPSs is 5, it takes 0.071 seconds to compare concepts. Much time has been spent on calculating MUPS because there exist 1788 axioms in this ontology. In contrast, for ontologies miniTambis and Chemical\* which contain about 30 unsatisfiable concepts, there is increasing time spent on comparison. Especially, for Chemical\* in which the cardinalities of some MUPSs are more than 10, the compare time is almost the same as the prepare time. Therefore, we can conclude that the efficiency of computation of score ordering on unsatisfiable concepts depends on the cardinalities of MUPSs. Even for a big ontology like *proton\_100\_all*, if the cardinalities of the MUPSs of its unsatisfiable concepts are small, the score ordering can still be computed quickly if we ignore the time spent on calculating MUPS.

### 6.2 Score Ordering on Terminologies

We also implemented the approaches for score orderings on terminology axioms and on ontologies. The performance of the approach for score ordering on terminology axioms is analyzed in Table 3. In Table 3, the second column and third column show the average runtime (in milliseconds) of calculating MIPS of the ontology (prepare time) and average runtime of generating the score ordering on axioms (compare time) respectively. The prepare time in Table 3 does not count the time spent on calculating all MUPSs in an ontology, which can be checked



**Table 3.** Score ordering on terminologies (time is in milliseconds)

Ontology	Prepare time	Compare time	Total time
miniTambis	171	10	181
Chemical*	511	30	541
<i>proton_100_all</i>	90	0.1	90.1

in Table 2. According to the table, it takes less than one second to calculate all the MIPSs from MUPSs for all test ontologies.

According to our experiment, the score of every terminology axiom in miniTambis is 1. This means that the MIPSs in miniTambis are pair-wise disjoint. For Chemical\* and *proton\_100\_all*, however, axioms may get different scores. For example, in Chemical\*, the axiom  $PublishedWork \sqsubseteq \neg Person$  gets score 2 and the axiom  $NerveAgentRelatedPublishedWork \sqsubseteq PublishedWork$  gets score 4 and so the former is less incoherent than the latter *w.r.t.* the score ordering. According to Table 3, the time spent on comparison is much less than that spent on calculating MIPSs.

Similar to the score ordering on terminology axioms, if we ignore the time spent on calculating MIPS, it also takes less than 1 second to obtain the score orderings on ontologies. This is reasonable because both the number of MIPS in each ontology and the size of the union of MIPS in each ontology are small. In contrast, the MUPS of an unsatisfiable concept is more likely to contain a large number of axioms and there are usually many unsatisfiable concepts in a coherent ontology. So it takes longer to obtain a score ordering on unsatisfiable concepts. The test ontologies are not comparable *w.r.t.* the score ordering  $\leq_S$ . That is, there does not exist two ontologies such that one is more incoherent than the other *w.r.t.* the score ordering.

## 7 Conclusions and Future Work

In this paper, we have defined two classes of measures for incoherent ontologies: measures of incoherence for unsatisfiable concepts and measures of incoherence for terminologies. The first class of measures gives us information on comparing unsatisfiable concept names. The second class of measures gives us information on comparing terminology axioms and comparing ontologies. These measures of incoherence can provide important information for dealing with incoherence and evaluating ontologies. Finally, we have implemented the approaches for measuring incoherence and reported on some preliminary but interesting experimental results.

For future work, we plan to apply the approaches of measuring incoherence to deal with incoherence and inconsistency in an ontology. The score ordering defined on unsatisfiable concepts is not applicable for ontologies which contain a large number of conflicting terminology axioms. We will explore some approximation techniques to solve this problem.

## Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. This work is partially supported by the EU under the IST project NeOn (IST-2006-027595, <http://www.neon-project.org/>).

## References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.: *The Description Logic Handbook: Theory, implementation and application*. Cambridge University Press, Cambridge (2003)
2. de la Banda, M.G., Stuckey, P.J., Wazny, J.: Finding all minimal unsatisfiable subsets. In: *Proceedings of the 5th ACM SIGPLAN international conference on Principles and practice of declarative programming*, pp. 32–43. ACM Press, New York (2003)
3. Deng, X., Haarslev, V., Shiri, N.: Measuring inconsistency in ontologies. In: *Proc. of ESWC 2007 (to appear, 2007)*
4. Flouris, G., Huang, Z., Pan, J.Z., Plexousakis, D., Wache, H.: Inconsistencies, negations and changes in ontologies. In: *Proc. of AAAI 2006*, pp. 1295–1300. AAAI Press, Stanford, California, USA (2006)
5. Hunter, A.: Measuring inconsistency in knowledge via quasi-classical models. In: *Proc. of AAAI 2002*, pp. 68–73. AAAI Press, Stanford, California, USA (2002)
6. Hunter, A.: Logical comparison of inconsistent perspectives using scoring functions. *Knowledge and Information Systems* 6(5), 528–543 (2004)
7. Hunter, A., Konieczny, S.: Shapley inconsistency values. In: *Proc. of KR 2006*, pp. 249–259 (2006)
8. Kalyanpur, A., Parsia, B., Sirin, E., Grau, B.C.: Repairing unsatisfiable concepts in owl ontologies. In: Sure, Y., Domingue, J. (eds.) *ESWC 2006*. LNCS, vol. 4011, pp. 170–184. Springer, Heidelberg (2006)
9. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging unsatisfiable classes in owl ontologies. *Journal of Web Semantics* 3(4), 268–293 (2005)
10. Knight, K.M.: Measuring inconsistency. *Journal of Philosophical Logic* 31, 77–98 (2001)
11. Konieczny, S., Lang, J., Marquis, P.: Quantifying information and contradiction in propositional logic through epistemic tests. In: *Proc. of IJCAI 2003*, pp. 106–111 (2003)
12. Ma, Y., Qi, G., Hitzler, P., Lin, Z.: Measuring inconsistency for description logics based on paraconsistent semantics. In: *Proc. of DL 2007 (2007)*
13. Meyer, T., Lee, K., Booth, R.: Knowledge integration for description logics. In: *Proc. of AAAI 2005*, pp. 645–650. AAAI Press, Stanford, California, USA (2005)
14. Qi, G., Liu, W., Bell, D.A.: A revision-based algorithm for handling inconsistency in description logics. In: *Proc. of NMR 2006*, pp. 124–132 (2006)
15. Qi, G., Pan, J.Z., Ji, Q.: A possibilistic extension of description logics. In: *Proc. of DL 2007 (to appear, 2007)*
16. Schlobach, S.: Diagnosing terminologies. In: *Proc. of AAAI 2005*, pp. 670–675. AAAI Press, Stanford, California, USA (2005)
17. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: *Proc. of IJCAI 2003*, pp. 355–362 (2003)

# Alternating-Offers Protocol for Multi-issue Bilateral Negotiation in Semantic-Enabled Marketplaces

Azzurra Ragone<sup>1</sup>, Tommaso Di Noia<sup>1</sup>, Eugenio Di Sciascio<sup>1</sup>,  
and Francesco M. Donini<sup>2</sup>

<sup>1</sup> SisInfLab, Politecnico di Bari, Bari, Italy  
{a.ragone,t.dinoia,disciascio}@poliba.it  
<sup>2</sup> Università della Tuscia, Viterbo, Italy  
donini@unitus.it

**Abstract.** We present a semantic-based approach to multi-issue bilateral negotiation for e-commerce. We use Description Logics to model advertisements, and relations among issues as axioms in a TBox. We then introduce a logic-based alternating-offers protocol, able to handle conflicting information, that merges non-standard reasoning services in Description Logics with utility theory to find the most suitable agreements. We illustrate and motivate the theoretical framework, the logical language, and the negotiation protocol.

## 1 Introduction

Fully automating negotiation mechanisms in e-marketplaces calls for adopting logical languages to model and reason on advertisements whenever the negotiation has to take place on complex descriptions, going beyond plain undifferentiated goods, where a single issue (usually price) is amenable to negotiation. Description Logics (DLs) are natural candidates languages for this purpose: they are the basis of Semantic Web Languages and they can be much more expressive than *e.g.*, Propositional Logic, yet they have decidable inference problems that can be useful in a number of negotiation scenarios. In this paper, in particular, we present a DL-based approach to multi-issue bilateral negotiation and introduce a novel logic-based alternating-offers protocol. The protocol merges DLs formalism and non-standard reasoning services with utility theory, to find the most suitable agreements. To this aim it takes into account existing logical relations between issues in requests and offers and related utilities of agents, expressed through logical formulas. The roadmap to the rest of this paper is as follows: next Section outlines the approach and assumptions we make. Then we move on to the logical formalism we adopt. Section 3 presents and motivates the protocol we devised. We discuss features of the protocol in Section 4. A brief analysis of relevant related work and a summary of the approach close the paper.

### 1.1 Scenario and Assumptions

We consider a marketplace of peer entities where users submit their semantically annotated descriptions, and negotiations among agents take place in a fully automated way. We start outlining: the *negotiation protocol*, *i.e.*, the set of rules that specifies how an

agreement can be reached; the *negotiation strategy*, that specifies the action to take in each situation, given an explicit set of rules specified in the negotiation protocol [15]; the *utility function* of the agents, which is used to evaluate the possible outcomes of the negotiation [11]. The assumptions characterizing the proposed negotiation mechanism are:

**one-to-many:** the negotiation is a one-to-many negotiation, since the buyer's agent will negotiate simultaneously with other  $m$  different agents – each one representing a seller, whose offer has been previously stored in the e-marketplace.

**rationality:** agents are *rational*, they behave according to their preferences and seek to maximize their utilities [11, p.19] doing in each step the minimum possible concession, *i.e.*, the concession involving the minimum utility loss, see protocol Section 3.

**incomplete information:** each agent knows its utility function and ignores the opponent disagreement thresholds and utility functions.

**conflict deal:** disagreement is better than an agreement iff the agent's utility over such an agreement is smaller than disagreement thresholds<sup>1</sup> set by the agent before negotiation starts. Therefore when the agent's utility deriving from accepting an agreement (or going on with the negotiation) and opting out it is the same, it will prefer not to opt out [11].

Here we just give a quick outlook of the protocol we propose in the framework, and will thoroughly detail it in Section 3. The protocol is inspired to Rubinstein's alternating-offers one [16]. In that setting an agent starts making an offer to its opponent, who can either accept, make a counter-offer or exit the negotiation. If a counter-offer is made, the negotiation goes on until one of the agents accepts an offer or exits the negotiation. In some cases there is a negotiation *deadline*; if such deadline is reached before one agent has accepted an offer, the negotiation ends in a conflict deal. Our protocol anyway is different from that of Rubinstein; actually we consider: *multi-issue negotiation*: buyer and seller do not negotiate on a single item or on a single bundle of items, but on many issues, which are related with each other through an ontology; such issues may also characterize more complex items (*e.g.*, in a computer store domain a notebook equipped with Wi-Fi adapter and DVD recorder). Note also that at this stage of our work we do not consider a time deadline. The protocol is sorted out by a finite set of steps<sup>2</sup>: the negotiation always terminates because either the agreement has been reached or because one agent opts out. The agent who moves first is selected randomly for each negotiation. At each step the agent who moves has two choices: *concede* or *opt out*, while the other one *stands still*. Agents are forced to concede until a *logical compatibility* is reached between the initial request and the supply, *i.e.*, until the inconsistency sources are eliminated. At each step, amongst all the allowed concessions that satisfy the concession criteria enforced by the protocol, the agent should choose the concession that gives the highest utility to itself: the *minimal concession*. Therefore a concession should be *minimal* w.r.t. the utility loss paid by the agent who makes the

<sup>1</sup> Disagreement thresholds, also called disagreement payoffs, or reservation values, are the minimum utility that each agent requires to pursue a deal [15].

<sup>2</sup> In the following, for the sake of simplicity, we always describe an interaction between only two opposite agents; although we notice that multiple negotiations can be performed at the same time, among *one* agent and *many* candidate partners.

concession [9]. The negotiation ends either if a logical compatibility is reached (*the negotiation succeeds*) or if one agent opts out (*the negotiation ends in a conflict deal*). For what concerns **strategy**, the main target of the agent is to reach the compatibility, because only through compatibility it is possible to reach an agreement. If it is its turn to move, an agent can choose to concede or opt out: if the utility of the agent at that step is smaller than its disagreement threshold, then the agent opts out and the negotiation ends immediately. Otherwise, it will do a concession: the concession is the *minimum possible concession*, that is the concession less decreasing its utility.

We define an agent's utility function over all possible outcomes [11] as:

$$u^p : \mathcal{A} \cup \{Opt\} \rightarrow \mathfrak{R} \quad (1)$$

where  $p \in \{\beta, \sigma\}$  —  $\beta$  and  $\sigma$  stand for buyer and seller respectively —  $\mathcal{A}$  is the set of all possible agreements,  $Opt$  stands for Opt out.

## 2 Logical Formalism

In this paper we will refer to  $\mathcal{AL}(D)$  [11], a fragment of OWL-DL, where besides `owl:Class` and `owl:ObjectProperty`, one is able to express `owl:DatatypeProperty`  $f$  (for **Features**) on objects such as year of building, length, weight and many others by means of *concrete domains*. Without loss of generality we assume that concrete domains we deal with are admissible [3]. In order to model the domain knowledge and represent relationships among elements, we use an ontology  $\mathcal{O}$  containing Concept Inclusion axioms of the form  $A \sqsubseteq C$  and  $A \equiv C$ , where the concept name  $A$  can appear only once on the left-hand side of axioms. We restrict  $\mathcal{O}$  to be acyclic, *i.e.*, the definition of  $A$  should not depend on  $A$  itself (see [11] for a precise definition). Using  $\mathcal{AL}(D)$  it is possible to express subclass relations, disjointness relations involving concept names. As an example consider the following axioms (hereafter we will use DL syntax which results more compact than OWL one):

$$\begin{aligned} \text{CheapPC} &\sqsubseteq \text{PC} \sqcap (\text{price} \leq 600) \quad (\textit{Subclass}) \\ \text{WinX} &\sqsubseteq \neg \text{Unix} \quad (\textit{DisjointClasses}) \end{aligned}$$

Formulas representing demands  $D$  and supplies  $S$ , are expressed as generic OWL-DL expressions. So, an example description can be the following one:

`PC`  $\sqcap$   $\neg$  `Notebook`  $\sqcap$  (`ram`  $\geq$  1024)  $\sqcap$  (`hdd`  $\leq$  160)  $\sqcap$   $\exists$  `hasOS`  $\sqcap$   $\forall$  `hasOS.Linux`  $\sqcap$   $\exists$  `monitor`  $\sqcap$   $\forall$  `monitor.(LCDmonitor`  $\sqcap$   $\exists$  `characteristics`  $\sqcap$   $\forall$  `characteristics.(inch`  $\geq$  17))  
 formally modeling this advertisement: “*personal computers, no notebooks, with a RAM of 1 Gbyte (with the possibility to extend), with an hard disk of at most 160 Gbytes and a 17” LCD monitor.*”.

Notice that for what concerns numerical properties, also range expressions are allowed in the form  $(f \geq n) \sqcap (f \leq m)$ . In order to better explain the approach, in the rest of the paper we will refer to ontology  $\mathcal{O}$  in Figure 1.

Even though subsumption and satisfiability are basic and useful reasoning tasks in a number of applications, there are typical problems related to negotiation that call for

<sup>3</sup> It is well known that  $\mathfrak{R}$  is admissible [2].

HomePC $\sqsubseteq$ PC $\sqcap$ $\exists$ hasOS $\sqcap$ $\exists$ hasProcessor;	CheapPC $\sqsubseteq$ PC $\sqcap$ (price $\leq$ 600)
Pentium $\sqsubseteq$ Intel.Processor $\sqcap$ (mhz $\geq$ 3000)	Centrino $\sqsubseteq$ $\neg$ Pentium
WinX $\sqsubseteq$ OperatingSystem;	Unix $\sqsubseteq$ OperatingSystem
Linux $\sqsubseteq$ Unix;	Unix $\sqsubseteq$ $\neg$ WinX

**Fig. 1.** The ontology used in the examples

non-standard reasoning services. For instance, suppose you have the buyer's agent  $\beta$  with her Demand represented by the concept  $D$  and the seller's agent  $\sigma$  with his Supply represented by  $S$ . In case  $\beta$ 's request  $D$  and  $\sigma$ 's offer  $S$  are in conflict with each other with respect to the domain knowledge modeled in the ontology  $\mathcal{O}$ — in formulae  $S \sqcap D \sqsubseteq_{\mathcal{O}} \perp$ — how to suggest to  $\beta$  which parts in  $D$  are in conflict with  $S$  and conversely to  $\sigma$  which parts in  $S$  are conflict with  $D$ ? The above question is very common, among others, in negotiation scenarios where you need to know “what is wrong” between  $D$  and  $S$  and negotiate on it. In order to give an answer to the previous question and provide explanations, Concept Contraction [5,8] can be exploited.

**Concept Contraction.** Given two concepts  $C_1$  and  $C_2$  and an ontology  $\mathcal{O}$ , where  $C_1 \sqcap C_2 \sqsubseteq_{\mathcal{O}} \perp$  holds, find two concepts  $K$  (for Keep) and  $G$  (for Give up) such that both  $C_1 \equiv K \sqcap G$  and  $K \sqcap C_2 \not\sqsubseteq_{\mathcal{O}} \perp$ .

In other words  $K$  represents a contraction of  $C_1$  which is satisfiable with  $C_2$ , whilst  $G$  represents some reason why  $C_1$  and  $C_2$  are not compatible with each other. With Concept Contraction, conflicting information both in  $\beta$ 's request w.r.t.  $\sigma$ 'supply can be computed and vice versa. Actually, for Concept Contraction minimality criteria have to be introduced. Following the Principle of Informational Economy [10], for  $G$  we have to give up as little information as possible. In [5,7] some minimality criteria were introduced and analyzed. In particular, if the adopted DL admits a normal form with conjunctions of concepts as  $\mathcal{AL}(D)$ ,  $G_{\exists}$  minimal irreducible solutions can be defined.

**Definition 1.** Let  $C_1$  and  $C_2$  be two concepts such that  $C_1 \sqcap C_2 \sqsubseteq_{\mathcal{O}} \perp$ . For the corresponding concept contraction problem  $\mathcal{Q}$ , we say the solution  $\langle G_{irr}, K_{irr} \rangle$  problem is  $G_{\exists}$ -irreducible minimal if the following conditions hold:

1.  $G_{irr} = \sqcap_{i=1 \dots n} G_i$  where both  $G_{irr} \sqsubseteq_{\mathcal{O}} \exists R$  and  $K_{irr} \not\sqsubseteq_{\mathcal{O}} \exists R$  iff  $C_2 \sqsubseteq \forall R. \perp$ ;
2.  $K \sqcap G_i \sqcap C_2 \sqsubseteq_{\mathcal{O}} \perp$ , for every  $i = 1 \dots n$ ;
3. if  $\langle G', K' \rangle$  is a solution to  $\mathcal{Q}$  satisfying Condition 1 then  $K' \not\sqsubseteq_{\mathcal{O}} K_{irr}$  holds.

The rationale behind the three conditions in above definition is the following:

**Condition 1.** This condition is needed in order to avoid solutions, i.e., negotiation outcomes, which could not be useful in the user perspective. Consider the following example referring to the ontology in Figure 1

$$D = \text{HomePC} \sqcap \forall \text{hasOS.Linux}$$

$$S = \text{PC} \sqcap \forall \text{hasOS.WinX}$$

Now,  $D$  and  $S$  are in conflict with each other w.r.t.  $\mathcal{O}$ ; in order to regain the compatibility,  $\beta$  may contract her request. In this case two possible solutions are:

$$\begin{aligned} \langle G_1, K_1 \rangle &= \langle \text{HomePC}, \text{PC} \sqcap \exists \text{hasProcessor} \sqcap \forall \text{hasOS.Linux} \rangle \\ \langle G_2, K_2 \rangle &= \langle \forall \text{hasOS.Linux}, \text{HomePC} \rangle \end{aligned}$$

The first solution does not satisfy Condition 1 because for the property  $\text{hasOS}$  both  $\text{HomePC} \sqsubseteq_{\mathcal{O}} \exists \text{hasOS}$  and  $\text{PC} \sqcap \exists \text{hasProcessor} \sqcap \forall \text{hasOS.Linux} \not\sqsubseteq_{\mathcal{O}} \exists \text{hasOS}$ . If  $\beta$  decides to contract her request following this solution, then she keeps all the specifications of requested characteristics (also the one on the operating system) but her choice would lead to a vacuously true agreement on the operating system specification with  $\sigma$ . In fact,  $K_1 \sqcap S$  implies  $\forall \text{hasOS}.\perp$  i.e., no operating system is admitted.

**Condition 2.** We desire to keep the number of conjuncts in  $G_{irr}$  as small as possible in order to avoid redundancies (minimality of  $G_{irr}$ ). Turning back to the previous example, another solution to the same  $\mathcal{Q}$  is  $\langle G_3, K_3 \rangle = \langle \forall \text{hasOS}.\langle \text{Linux} \sqcap \text{Unix} \rangle, \text{HomePC} \rangle$ .

**Condition 3.** Conversely, we want to keep as much information as possible in  $K_{irr}$ . In this example  $\langle G_4, K_4 \rangle = \langle \forall \text{hasOS.Linux}, \text{HomePC} \sqcap \forall \text{hasOS.OperatingSystem} \rangle$ .

With respect to the above example the only solution satisfying Definition 1 is  $\langle G_4, K_4 \rangle$ . To compute a  $G_{\exists}$ -irreducible minimal solution, the algorithm proposed in [6] can be simply adapted and used. Notice that even though within  $\mathcal{O}$  in Figure 1 the disjunction relation is between  $\text{WinX}$  and  $\text{Unix}$ , the concept contraction procedure suggests to give up  $\forall \text{hasOS.Linux}$  because  $\text{Linux}$  is a sub-class of  $\text{Unix}$ .

## 2.1 Dealing with Incomplete Information

Information about supply/demand descriptions can be, in our setting, incomplete. This may happen not only because some information may be unavailable, but also because some details have been considered irrelevant by either the seller or the buyer when they submitted their advertisements. For instance, some user may find tiresome to specify a lot of characteristics related to the brand or more technical characteristics of the product the user can be unaware of. Currently, the most common approach to this problem is avoiding incompleteness by forcing the user to fill long and tedious forms. There are several ways to deal with incomplete information and the choice may influence a negotiation. Suppose to have the following two entries in the e-marketplace:

*D: I am looking for a PC equipped with Wi-Fi and DVD recorder.*

*S: I offer a home PC with Intel Processor equipped with Linux Operating System.*

The above descriptions are then formalized as:

*D:*  $\text{PC} \sqcap \forall \text{hasDevice.WiFi} \sqcap \forall \text{hasStorageDevice.DVDrecorder}$

*S:*  $\text{HomePC} \sqcap \forall \text{hasProcessor.Intel} \sqcap \forall \text{hasOS.Linux}$

Under an *open-world assumption* we have two possible choices. First, we can keep incomplete information as *missing* information: we do not know e.g., if the buyer is not interested in a particular characteristic or he simply has forgotten to specify it. In



this case the system has to contact to buyer/seller to further refine her/his description. Asking users to refine their descriptions before the negotiation process starts seems quite unrealistic, because of the amount of descriptions that can be stored in the system itself. It appears more feasible to leave this phase to a second negotiation stage related to a small subset of supplies/demands. For instance, the ones with the highest utility product. In fact these solutions are known to be Pareto-efficient and “fair”, according to Nash [11]. Once buyer and seller have refined their descriptions it is possible to start a new negotiation (the so-called *post-negotiation* phase) where only the *updated* information is negotiated. On the other hand, still in the open-world assumption setting, a second possible choice can be to assume incomplete information as an *any-would-fit* assertion (don’t care), so the system should cope with this incompleteness as is. Therefore also this information will be presented in the final agreement. Following the above example the final agreement would be:

$A = \text{HomePC} \sqcap \forall \text{hasProcessor. Intel} \sqcap \forall \text{hasOS. Linux} \sqcap \text{PC} \sqcap$   
 $\forall \text{hasDevice. WiFi} \sqcap \forall \text{hasStorageDevice. DVDrecorder}$

In the latter case no human intervention is needed. In this paper we take this approach, with agents not caring about missing information.

### 3 Logic-Based Alternating-Offers Protocol

In this section we model an alternating-offer protocol taking into account the semantics of request and offers as well as the domain knowledge modeled within an ontology in the OWL DL fragment we identified in Section 2, exploiting Concept Contraction. For the sake of clarity and without loss of generality, from now on we consider that the agent entering the marketplace is the buyer  $\beta$  and her potential partners are the sellers’ agents  $\sigma$ . The first step of the protocol is the normalization of both  $\beta$ ’s demand  $D$  and  $\sigma$ ’s supply  $S$ . The normalization step substitutes  $A$  with  $A \sqcap C$  everywhere in a concept, if either  $A \sqsubseteq C$  or  $A \equiv C$  appears in  $\mathcal{O}$ , then considers the equivalence  $\forall R.(A \sqcap B) \equiv \forall R.A \sqcap \forall R.B$  as a recursive rewrite rule from left to right. After the normalization stage,  $D$  is a conjunction of elements in the form:

$$D = \prod_i C_i \quad (2)$$

where  $C_i \in \text{Norm}$  with

$$\text{Norm} = \left\{ \begin{array}{l} \left. \begin{array}{l} CN \\ \neg CN \end{array} \right\} - CN \text{ is a Class Name} \\ \exists R - R \text{ is an Object Property} \\ \left. \begin{array}{l} (f \leq n) \\ (f \geq n) \\ (f = n) \end{array} \right\} - \text{Constraints on numerical features} \\ \forall R.C - \text{with } C \in \text{Norm} \end{array} \right.$$

As an example consider the concept in Section 2. After normalization it is rewritten as:  
 $\text{PC} \sqcap \neg \text{Notebook} \sqcap (\text{ram} \geq 1024) \sqcap (\text{hdd} \leq 160) \sqcap \exists \text{hasOS} \sqcap \forall \text{hasOS. Linux} \sqcap \exists \text{monitor} \sqcap$   
 $\forall \text{monitor. LCDmonitor} \sqcap \forall \text{monitor.} \exists \text{characteristics} \sqcap$   
 $\forall \text{monitor.} \forall \text{characteristics.} (\text{inch} \geq 17) .$



In the normalized form  $C_i$  represents issues the user is willing to negotiate on. The buyer is able to express her utilities on single issues or on bundles of them. For instance, w.r.t. the previous request the buyer may set utility values on PC (single issue) as well as on the whole formula  $(ram \geq 1024) \sqcap \forall monitor.LCDmonitor$  (bundle of issue). We indicate these concepts with  $P_k$  — for **P**references.

Now, for each  $P_k$  the buyer  $\beta$  expresses a utility value  $u^\beta(P_k)$  such that  $\sum_i u^\beta(P_k) = 1$ . As usual, both agents' utilities are normalized to 1 to eliminate outliers, and make them comparable. Since we assume that utilities are additive, the global utility is just a suitable sum of the utilities for preferences entailed by the final agreement. In particular, given a concept expression  $A$  representing a final agreement, we define the final utility associated to the agent  $p$ , with  $p \in \{\beta, \sigma\}$ , see **(II)** as:

$$u^p(A) = \sum_k \{u^p(P_k) \mid A \sqsubseteq P_k\} \quad u^p(Opt) = t_p \tag{3}$$

where  $t_p$  is the **disagreement threshold** of agent  $p$  (see Section **(LJ)**).

### 3.1 The Protocol

Summing up, before the real negotiation starts (step 0) we have a demand  $D$  and a supply  $S$  such that:

$$D = \prod_i C_i \qquad S = \prod_j C_j$$

Based on  $C_i$  and  $C_j$ , the buyer and seller, respectively, formulate their preferences  $P_k$  (for the buyer) and  $P_h$  (for the seller) and for each of them set a utility value such that:

$$\sum_k u^\beta(P_k) = 1 \qquad \sum_h u^\sigma(P_h) = 1$$

Finally, both for  $\beta$  and  $\sigma$  we have the corresponding **disagreement thresholds** and utility functions  $t_\beta, u^\beta$  and  $t_\sigma, u^\sigma$ .

If  $D \sqcap S \sqsubseteq_{\mathcal{O}} \perp$  then demand and supply descriptions are in conflict with each other and  $\beta$  and  $\sigma$  need to negotiate on conflicting information if they want to reach an agreement. The negotiation will follow an alternating offers pattern: at each step, either  $\beta$  or  $\sigma$  gives up a portion of its conflicting information choosing the item with the minimum utility. At the beginning, both  $\beta$  and  $\sigma$  need to know what are the conflicting information. Notice that both agents  $\beta$  and  $\sigma$  know  $D$  and  $S$ , but they have no information neither on counterpart utilities nor preferences. Both  $\beta$  and  $\sigma$  will solve two Concept Contraction problems, computing a  $G_{\exists}$  minimal irreducible solution, and rewrite  $D$  and  $S$  as:

$$D = G_0^\beta \sqcap K_0^\beta \qquad S = G_0^\sigma \sqcap K_0^\sigma$$

In the above rewriting  $G_0^\beta$  and  $G_0^\sigma$  represent, respectively, some of the causes that make  $D$  in conflict with  $S$  and the reason why  $S$  is in conflict with  $D$ . At a first glance it would seem  $\beta$  needs only  $\langle G_0^\beta, K_0^\beta \rangle$  and  $\sigma$  needs  $\langle G_0^\sigma, K_0^\sigma \rangle$ . We will see later that  $\beta$

needs also information on  $\sigma$  in order to check its fairness during negotiation steps. Since we compute  $G$ -irreducible solutions we can normalize  $G_0^\beta$  and  $G_0^\sigma$ , following the same procedure for  $D$  and  $S$ , as:

$$G_0^\beta = G_{(0,1)}^\beta \sqcap G_{(0,2)}^\beta \sqcap \dots \sqcap G_{(0,n)}^\beta = \prod_{i=1}^n G_{(0,i)}^\beta$$

$$G_0^\sigma = G_{(0,1)}^\sigma \sqcap G_{(0,2)}^\sigma \sqcap \dots \sqcap G_{(0,m)}^\sigma = \prod_{j=1}^m G_{(0,j)}^\sigma$$

In the previous formulas, indexes  $(0, i)$  and  $(0, j)$  represent the  $i$ -th and  $j$ -th conjunctive element in  $G^\beta$  and  $G^\sigma$  at round 0. Due to the structure of  $D$ ,  $S$  and  $\mathcal{O}$  we have that: for each  $G_{(0,i)}^\beta$  there always exists a  $C_i$  in the normalized version of  $D$ — as represented in Equation (2) — such that  $G_{(0,i)}^\beta = C_i$ . The same relation holds between each  $G_{(0,j)}^\sigma$  and  $C_j$  in the normalized form of  $S$ . Hence, some of  $P_k$  and  $P_h$  can be partially rewritten in terms of  $G_{(0,i)}^\beta$  and  $G_{(0,j)}^\sigma$  respectively. Since the information in  $G_0^\beta$  and  $G_0^\sigma$  are the reason why an agreement is not possible, then either  $\beta$  or  $\sigma$  will start conceding one of  $G_{(0,i)}^\beta$  or  $G_{(0,j)}^\sigma$  reducing their global utility of  $u(G_{(0,i)}^\beta)$  or  $u(G_{(0,j)}^\sigma)$ , respectively.

Suppose  $\beta$  starts the negotiation and gives up  $G_{(0,2)}^\beta = C_5$  with  $P_3 \sqsubseteq_{\mathcal{O}} G_{(0,2)}^\beta$ . Then she reformulates her request as

$$D_1 = \prod_{i=1..4,6..} C_i$$

and sends it to  $\sigma$ . Notice that since  $P_3 \sqsubseteq_{\mathcal{O}} G_{(0,2)}^\beta$ , the global utility of  $\beta$  decreases to

$$u_1^\beta = \sum_{k=1..2,4..} u(P_k)$$

Now,  $\sigma$  is able to validate if  $\beta$  really changed her request to reach an agreement and did not lie. To do so,  $\sigma$  computes  $\langle G_1^\beta, K_1^\beta \rangle$  solving a concept contraction problem w.r.t. the new demand  $D_1$  and checks if  $G_1^\beta$  is more general than  $G_0^\beta$ . In formulas,  $\sigma$  checks if  $G_0^\beta \sqsubseteq_{\mathcal{O}} G_1^\beta$  holds, in case of positive answer, then  $\sigma$  knows that  $\beta$  did not lie and he continues the negotiation process. Otherwise he may decide to leave the negotiation (conflict deal) or ask  $\beta$  to reformulate her counteroffer. If the negotiation continues,  $\sigma$  computes his conflicting information w.r.t. to  $D_1$  and rewrites  $S$  as:

$$S = G_1^\sigma \sqcap K_1^\sigma \quad \text{where} \quad G_1^\sigma = \prod_{j=1}^m G_{(1,j)}^\sigma$$

Again, for each  $G_{(1,j)}$  there exists a  $C_j$  in the normalized version of  $S$ . Hence, if  $\sigma$  decides to concede  $G_{(1,j)}$ , his global utility decreases proportionally to the utility of  $P_h$  to which  $G_{(1,j)}$  belongs to. Once  $\sigma$  sends his counteroffer to  $\beta$ , this latter is able to check if

$\sigma$  lied. Similarly to  $\sigma$  in step 0,  $\beta$  computes  $\langle G_1^\sigma, K_1^\beta \rangle$  and checks if  $G_0^\sigma \sqsubseteq_{\mathcal{O}} G_1^\sigma$ . The process ends when one of the following two conditions holds:

1. the global utility of an agent is lower than its **disagreement threshold**. In this case the negotiation terminates with a conflict deal.
2. there is nothing more to negotiate on and the global utility of each agent is greater than its disagreement threshold. In this case the negotiation terminates with an agreement. **The agreement  $A$  is computed** simply as  $A = D_{last} \sqcap S_{last}$ , where  $D_{last}$  and  $S_{last}$  are the request and the offer in the last step.

### 3.2 Minimum Concession

Since users can express an utility value also on bundles, whenever they concede an issue as the **minimum concession** (in term of minimum global utility decrease), the set of all the bundles in which the issue is present has to be taken into account. They choose based on the utility of the whole set. For instance, suppose the buyer sets as preferences the following ones:

$$P_1 = \forall \text{monitor.LCDmonitor}$$

$$P_2 = (\text{hdd} \leq 200)$$

$$P_3 = \forall \text{monitor.LCDmonitor} \sqcap \forall \text{monitor.}\forall \text{characteristics.}(\text{inch} = 17)$$

with the following utilities:  $u^\beta(P_1) = 0.1$ ,  $u^\beta(P_2) = 0.4$  and  $u^\beta(P_3) = 0.5$ .

At the  $n$ -th step the conflicting information is:

$$G_n^\beta = \forall \text{monitor.LCDmonitor} \sqcap (\text{hdd} \leq 200)$$

Hence,  $\beta$  can concede whether  $\forall \text{monitor.LCDmonitor}$  or  $(\text{hdd} \leq 200)$ . If she concedes  $\forall \text{monitor.LCDmonitor}$  then her global utility decreases of  $u^\beta(P_1) + u^\beta(P_3) = 0.6$ , while conceding  $(\text{hdd} \leq 200)$  her utility decreases of  $u^\beta(P_2) = 0.4$  only. In this case the **minimum concession** is  $(\text{hdd} \leq 200)$ .

### 3.3 The Algorithm

Here we define the behavior of agents during a generic  $n$ -th round of the negotiation process. We present only the algorithm related to  $\beta$ 's behavior since the behavior of  $\sigma$  is dual w.r.t.  $\beta$ 's one.

- 1-4** If there is nothing in conflict between the old  $D_{n-1}$  and just-arrived  $S_n$ , then there is nothing more to negotiate on: the agreement is reached and returned. Notice that while computing the final agreement we use the "any-would-fit" approach to deal with *incomplete information* (see Section [2.1](#)).
- 5-11** If  $\beta$  discovers that  $\sigma$  lied on his concession, then  $\beta$  decides to exit the negotiation and terminates with a conflict deal. If we want  $\beta$  ask  $\sigma$  to concede again it is straightforward to change the protocol to deal with such a behavior.
- 13-15** If after the minimum concession, the utility of  $\beta$  is less than her **disagreement threshold**, then the negotiation ends with a conflict deal.

```

1 if  $D_{n-1} \sqcap S_n \not\sqsubseteq_{\mathcal{O}} \perp$  then
2   agreement  $A$  reached;
3   return  $A = D_{n-1} \sqcap S_n$ ;
4 end
5 if  $n > 0$  then
6   compute  $\langle G_n^\sigma, K_n^\sigma \rangle$  from  $D_{n-1}$  and  $S_n$ ;
7   if  $G_{n-1}^\sigma \not\sqsubseteq_{\mathcal{O}} G_n^\sigma$  then
8      $\sigma$  lied;
9     conflict deal: exit;
10  end
11 end
12 compute minimum concession  $G_{(n-1,i)}^\beta$ ;
13 if  $u_{n-1}^\beta < t^\beta$  then
14   conflict deal: exit;
15 end
16 formulate  $D_n$  deleting  $G_{(n-1,i)}^\beta$  from  $D_{n-1}$ ;
17 send  $D_n$  to  $\sigma$ ;

```

**Algorithm 1.** The behavior of  $\beta$  at step  $n$

### 3.4 An Illustrative Example

Keeping the computer equipment as reference domain, consider the following example: a buyer  $\beta$  looking for a “personal computer equipped with an Intel processor whose clock frequency is at least 3000 MHz, 2 Giga bytes of RAM. The computer must have Linux operating system pre-installed. A 19” LCD monitor is also requested.”. On the other side, a seller offers a “personal computer for domestic use with a 2500 MHz Pentium on board. The computer is provided with a Windows operating system, a 17” LCD monitor and a WiFi adapter.”. Both the request  $D$  and the offer  $S$  can be formalized as:

$$D = \text{PC} \sqcap \exists \text{hasProcessor} \sqcap \forall \text{hasProcessor}.(\text{Intel\_Processor} \sqcap (\text{mhz} \geq 3000)) \sqcap \\ \exists \text{hasOS} \sqcap \forall \text{hasOS}. \text{Linux} \sqcap (\text{hdd} \geq 100) \sqcap \exists \text{monitor} \sqcap \forall \text{monitor}.(\text{LCDmonitor} \sqcap \\ \exists \text{characteristics} \sqcap \forall \text{characteristics}.(\text{inch} = 19)) \sqcap (\text{ram} = 2048)$$

$$S = \text{HomePC} \sqcap \forall \text{hasProcessor}.(\text{Pentium} \sqcap (\text{mhz} = 2500)) \sqcap \forall \text{hasOS}. \text{WinX} \sqcap (\text{hdd} = 80) \sqcap \\ \exists \text{monitor} \sqcap \forall \text{monitor}.(\text{LCDmonitor} \sqcap \exists \text{characteristics} \sqcap \\ \forall \text{characteristics}.(\text{inch} = 17)) \sqcap \exists \text{hasWiFi}$$

Their normalized forms<sup>4</sup> are respectively:

$$D = \text{PC} \sqcap \exists \text{hasProcessor} \sqcap \forall \text{hasProcessor}. \text{Intel\_Processor} \sqcap \\ \forall \text{hasProcessor}.(\text{mhz} \geq 3000) \sqcap \exists \text{hasOS} \sqcap \forall \text{hasOS}. \text{Linux} \sqcap (\text{hdd} \geq 100) \sqcap \\ \exists \text{monitor} \sqcap \forall \text{monitor}. \text{LCDmonitor} \sqcap \\ \forall \text{monitor}. \exists \text{characteristics} \sqcap \forall \text{monitor}. \forall \text{characteristics}.(\text{inch} = 19) \sqcap \\ (\text{ram} = 2048)$$

<sup>4</sup> In order to keep the example compact, we do not consider here the normalization step ( $f = n \equiv (f \geq n) \sqcap (f \leq n)$ ), which should be taken into account.

$$S = \text{HomePC} \sqcap \forall \text{hasProcessor.Pentium} \sqcap \forall \text{hasProcessor.}(\text{mhz} = 2500) \sqcap \\ \forall \text{hasOS.WinX} \sqcap (\text{hdd} = 80) \sqcap \exists \text{monitor} \sqcap \forall \text{monitor.LCDmonitor} \sqcap \\ \forall \text{monitor.}\exists \text{characteristics} \sqcap \forall \text{monitor.}\forall \text{characteristics.}(\text{inch} = 17) \sqcap \\ \exists \text{hasWiFi}$$

Now both agents  $\beta$  and  $\sigma$  set their preferences with corresponding utilities and utility threshold.

$$P_1^\beta = \forall \text{monitor.LCDmonitor} \quad u^\beta(P_1^\beta) = 0.4$$

$$P_2^\beta = (\text{hdd} \geq 100) \quad u^\beta(P_2^\beta) = 0.2$$

$$P_3^\beta = \forall \text{hasOS.Linux} \quad u^\beta(P_3^\beta) = 0.2$$

$$P_4^\beta = \forall \text{monitor.LCDmonitor} \sqcap \\ \forall \text{monitor.}\forall \text{characteristics.}(\text{inch} = 19) \quad u^\beta(P_4^\beta) = 0.1$$

$$P_5^\beta = (\text{ram} = 2048) \quad u^\beta(P_5^\beta) = 0.1$$

$$t^\beta = 0.6$$

$$P_1^\sigma = \forall \text{monitor.}\forall \text{characteristics.}(\text{inch} = 17) \quad u^\sigma(P_1^\sigma) = 0.4$$

$$P_2^\sigma = \forall \text{hasProcessor.}(\text{mhz} = 2500) \quad u^\sigma(P_2^\sigma) = 0.3$$

$$P_3^\sigma = (\text{hdd} = 80) \quad u^\sigma(P_3^\sigma) = 0.2$$

$$P_4^\sigma = \forall \text{hasOS.WinX} \quad u^\sigma(P_4^\sigma) = 0.1$$

$$t^\sigma = 0.5$$

$K$  and  $G$  are computed for both  $\beta$  and  $\sigma$ .

$$K_0^\beta = \text{PC} \sqcap \exists \text{hasProcessor} \sqcap \forall \text{hasProcessor.Intel.Processor} \sqcap \exists \text{hasOS} \sqcap \\ \forall \text{hasOS.OperatingSystem} \sqcap \exists \text{monitor} \sqcap \forall \text{monitor.}\exists \text{characteristics} \sqcap \\ \forall \text{monitor.LCDmonitor} \sqcap (\text{ram} = 2048)$$

$$G_0^\beta = \forall \text{monitor.}\forall \text{characteristics.}(\text{inch} = 19) \sqcap \forall \text{hasProcessor.}(\text{mhz} \geq 3000) \sqcap \\ \forall \text{hasOS.Linux} \sqcap (\text{hdd} \geq 100)$$

$$K_0^\sigma = \text{HomePC} \sqcap \forall \text{hasProcessor.Pentium} \sqcap \exists \text{monitor} \sqcap \forall \text{monitor.LCDmonitor} \sqcap \\ \forall \text{monitor.}\exists \text{characteristics} \sqcap \exists \text{hasWiFi}$$

$$G_0^\sigma = \forall \text{monitor.}\forall \text{characteristics.}(\text{inch} = 17) \sqcap \forall \text{hasProcessor.}(\text{mhz} = 2500) \sqcap \\ (\text{hdd} = 80) \sqcap \forall \text{hasOS.WinX}$$

Now suppose that by coin tossing,  $\beta$  moves first. She starts giving up the constraint on processor clock frequency, which is her minimum concession since a utility not assigned to a characteristic is usually equivalent to a utility equal to zero. Then she computes her utility and, since it is greater than the threshold value, decides to go on with the negotiation process. In the following step we have:

$$K_1^\beta = \text{PC} \sqcap \exists \text{hasProcessor} \sqcap \forall \text{hasProcessor.Intel.Processor} \sqcap \exists \text{hasOS} \\ \sqcap \forall \text{hasOS.OperatingSystem} \sqcap \exists \text{monitor} \sqcap \forall \text{monitor.}\exists \text{characteristics} \sqcap \\ \forall \text{monitor.LCDmonitor} \sqcap (\text{ram} = 2048)$$

$$G_1^\beta = \forall \text{monitor.}\forall \text{characteristics.}(\text{inch} = 19) \sqcap \text{hasOSLinux} \sqcap (\text{hdd} \geq 100)$$

$$K_1^\sigma = \text{HomePC} \sqcap \forall \text{hasProcessor.Pentium} \sqcap \exists \text{monitor} \sqcap \forall \text{monitor.LCDmonitor} \sqcap \\ \forall \text{monitor.}\exists \text{characteristics} \sqcap \exists \text{hasWiFi} \sqcap \forall \text{hasProcessor.}(\text{mhz} = 2500)$$

$$G_1^\sigma = \forall \text{monitor.} \forall \text{characteristics.} (\text{inch} = 17) \sqcap (\text{hdd} = 80) \sqcap \forall \text{hasOS. WinX}$$

At this point,  $\sigma$  gives up  $\forall \text{hasOS. WinX}$  which is the preference with the minimum utility. The protocol continues until agents reach logical compatibility. A final agreement could then be:

$$\begin{aligned} A = & \text{HomePC} \sqcap \exists \text{hasOS} \sqcap \forall \text{hasOS. Linux} \sqcap \exists \text{hasProcessor} \sqcap \forall \text{hasProcessor.} (\text{mhz} = \\ & 2500) \sqcap \forall \text{hasProcessor. Pentium} \sqcap (\text{hdd} = 80) \sqcap (\text{ram} = 2048) \sqcap \\ & \forall \text{monitor. LCDmonitor} \sqcap \forall \text{monitor.} \exists \text{characteristics} \sqcap \\ & \forall \text{monitor.} \forall \text{characteristics.} (\text{inch} = 19) \sqcap \exists \text{hasWiFi,} \end{aligned}$$

with corresponding utilities  $u^\beta = u^\beta(P_1^\beta) + u^\beta(P_3^\beta) + u^\beta(P_4^\beta) + u^\beta(P_5^\beta) = 0.8$  for  $\beta$  and  $u^\sigma = u^\sigma(P_2^\sigma) + u^\sigma(P_3^\sigma) = 0.5$  for  $\sigma$ .

## 4 Features of the Negotiation Mechanism

We briefly point out some characteristics of the proposed negotiation protocol [15].

**Semantics.** In the proposed protocol a formal language is exploited. Using a domain ontology it is possible to discover implicit conflicting information between a demand  $D$  and a supply  $S$ . Furthermore, a general logic-based technique, *i.e.*, Concept Contraction, grounded in well-known belief revision theory [10], is adopted to compute concessions.

**Bundles.** Users (both  $\beta$  and  $\sigma$ ) can set utility values not only on single issues but also on a bundle of issues. **Strong Agreement.** During the negotiation process the agents negotiate on conflicting issues. Hence, if the negotiation does end with an outcome, no conflicting information are present in the final agreement. **Simplicity.** Interaction among agents requires low communication costs: agents do not have to guess either the preferences of their opponent or their utility function. Moreover it is possible to compute strategy in a reasonable amount of time [11]. **Distribution.** The negotiation mechanism does not require a mediator helping parties to reach an agreement during the negotiation process or managing the entire process. Agents negotiate on their own with no need of an external help. **Efficiency.** If the protocol ends with an agreement, this is on the Pareto frontier. Due to the structure of formulas  $\beta$  and  $\sigma$  and  $G$ -irreducible solutions to concept contraction problems, during round  $n$  for each  $G_{(n,i)}^\beta \in G_n^\beta$ , representing the source of information in  $D_n$ , there is always at least one (in the most general case, more than one)  $G_{(n,j)}^\sigma \in G_n^\sigma$  representing the corresponding source of information in  $S_n$ . Hence, if  $\beta$  concedes  $G_{(n,i)}^\beta$ , losing the utility of bundles containing  $G_{(n,i)}^\beta$ , then  $\sigma$  is sure to maintain the utility on bundles involving  $G_{(n,j)}^\sigma$ . In other words, if an agent concedes something decreasing its utility then the opponent's utility does not decrease. Notice that, in order to reach a surely Pareto-efficient agreement and always find a Nash bargain solution, while maintaining the same logic approach and the use of bundles, a protocol similar to the one proposed in [9] can be adopted. Nevertheless, if we want to consider the *willingness to risk conflict* of both agents, in order to decide for each round which agent will concede, it becomes hard to maintain the approach fully distributed. If the approach remains distributed then we have to hypothesize agents estimate the willingness to risk conflict of the opponent. In this case we would probably lose in simplicity.

## 5 Related Work and Summary

Several recent logic-based approaches to negotiation are based on propositional logic. In [3], Weighted Propositional Formulas (WPF) are used to express agents preferences in the allocation of indivisible goods, but no common knowledge (as our TBox) is present. Utility functions expressed through WPF are classified in [4] according to the properties of the utility function (sub/super-additive, monotone, etc.). We used the most expressive functions according to that classification, namely, weights over unrestricted formulas, but for the fact that our formulas are DL concepts—*i.e.*, non-propositional. In [17] an agreement is defined as a model for a set of formulas from both agents, but agents preferences are not taken into account. In [12] a propositional logic framework endowed of an ontology  $\mathcal{T}$  is proposed, where a one-shot protocol is exploited to reach Pareto-efficient agreements. In order to reach a Pareto-efficient agreement a trusted mediator is needed, to whom agents reveal their preferences, and which suggests to the agents the most suitable agreement. The framework in [12] was further improved in [14], extending propositional logic with concrete domains in order to handle also numerical features as price, weight, time etc. In this work, instead, no mediator is needed. The work presented in [18] adopts a kind of propositional knowledge base arbitration to choose a fair negotiation outcome. However, *common knowledge* is considered as just more entrenched preferences, that could be even dropped in some deals. Instead, the logical constraints in our ontology  $\mathcal{T}$  of formulas must *always* be enforced in the negotiation outcomes. Finally we devised a *protocol* which the agents should adhere to while negotiating; in contrast, in [18] a game-theoretic approach is taken, presenting no protocol at all, since communication between agents is not considered. Although we used a rather inexpressive DL, our approach can be easily extended up to  $\mathcal{AL}\mathcal{E}\mathcal{H}(D)$ , which can express qualified existential concepts and sub-properties. Summarizing, we have motivated and illustrated a logic-based approach to bilateral negotiation in P2P e-marketplaces; we proposed a semantic-based alternating-offers protocol exploiting Description Logics, non-standard inference services, and utility theory to find the most suitable agreements. To the best of our knowledge there is only another work exploiting DLs in negotiation scenarios [13]. In that work the more expressive  $\mathcal{SHOIN}(D)$  is used to model the logic-based negotiation protocol, and only standard inference services, such a satisfiability, are exploited in order to catch inconsistency sources between  $D$  and  $S$ . Instead, the use of a non-standard inference service, such a Concept Contraction, can be useful to provide also an explanation of “what is wrong” between  $D$  and  $S$ , *i.e.*, the reason why  $\beta$  and  $\sigma$  can not reach an agreement and *what* has to be given up in order to reach that. However, in this paper we model a scenario with *partial* incomplete information (agents know opponent preferences, but not utilities of such preferences), while in [13] a scenario with *fully* incomplete information is studied, where agents do not know anything about the opponent one (neither preferences nor utilities). Moreover, differently from the approach presented in [13], the negotiation mechanism also works without agents knowing their *exact* utilities. It is enough that each agent knows, at every round, which issue to concede next; so only partial orders on issues are needed. Work is ongoing on various directions, namely: extending the DL adopted, finding a “cheap” way to ensure that the reached agreement is Pareto-efficient, and carry out large-scale experiments with real advertisements. We also plan to apply our approach to

Semantic Web Services (SWS) contracting, for negotiating Service Level Agreements (SLA) between users and SWS providers.

## Acknowledgments

We acknowledge support of project EU-FP-6-IST-026896 "TOWL". We are grateful to one anonymous reviewer for useful suggestions.

## References

1. Baader, F., Calvanese, D., Mc Guinness, D., Nardi, D., Patel-Schneider, P.: *The Description Logic Handbook*. Cambridge University Press, Cambridge (2002)
2. Baader, F., Hanschke, P.: A schema for integrating concrete domains into concept languages. In: *proc. of IJCAI 1991*, pp. 452–457 (1991)
3. Bouveret, S., Lemaitre, M., Fargier, H., Lang, J.: Allocation of indivisible goods: a general model and some complexity results. In: *Proc. of AAMAS 2005*, pp. 1309–1310 (2005)
4. Chevaleyre, Y., Endriss, U., Lang, J.: Expressive power of weighted propositional formulas for cardinal preference modeling. In: *Proc. of KR 2006*, pp. 145–152 (2006)
5. Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., Mongiello, M.: Concept Abduction and Contraction in Description Logics. In: *Proc. of DL 2003* (2003)
6. Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., Mongiello, M.: A uniform tableaux-based method for concept abduction and contraction in description logics. In: *Proc. of ECAI 2004*, pp. 975–976 (2004)
7. Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F., Mongiello, M.: Concept Abduction and Contraction for Semantic-based Discovery of Matches and Negotiation Spaces in an E-Marketplace. *Electronic Commerce Research and Applications* 4(4), 345–361 (2005)
8. Di Noia, T., Di Sciascio, E., Donini, F.: Semantic Matchmaking as Non-Monotonic Reasoning: A Description Logic Approach. *Journal of Artificial Intelligence Research* 29, 269–307 (2007)
9. Endriss, U.: Monotonic concession protocols for multilateral negotiation. In: *Proc. of AAMAS 2006*, pp. 392–399 (2006)
10. Gärdenfors, P.: *Knowledge in Flux: Modeling the Dynamics of Epistemic States*, Bradford Books. MIT Press, Cambridge (1988)
11. Kraus, S.: *Strategic Negotiation in Multiagent Environments*. MIT Press, Cambridge (2001)
12. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.: A logic-based framework to compute pareto agreements in one-shot bilateral negotiation. In: *Proc. of ECAI 2006*, pp. 230–234 (2006)
13. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.: Description logics for multi-issue bilateral negotiation with incomplete information. In: *proc. of AAAI 2007*, pp. 477–482 (2007)
14. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.: When price is not enough: Combining logical and numerical issues in bilateral negotiation. In: *Proc. of AAMAS 2007*, pp. 97–99 (2007)
15. Rosenschein, J.S., Zlotkin, G.: *Rules of Encounter*. MIT Press, Cambridge (1994)
16. Rubinstein, A.: Perfect equilibrium in a bargaining model. *Econometrica* 50, 97–109 (1982)
17. Wooldridge, M., Parsons, S.: Languages for negotiation. In: *Proc of ECAI 2004*, pp. 393–400 (2000)
18. Zhang, D., Zhang, Y.: A computational model of logic-based negotiation. In: *Proc. of AAAI 2006*, pp. 728–733. AAAI Press, Stanford, California, USA (2006)



# An Event-Based Approach for Semantic Metadata Interoperability

Tuukka Ruotsalo and Eero Hyvönen

Semantic Computing Research Group (SeCo)  
Helsinki University of Technology (TKK), Laboratory of Media Technology  
University of Helsinki, Department of Computer Science  
firstname.lastname@tkk.fi  
<http://www.seco.tkk.fi/>

**Abstract.** This paper presents a method for making metadata conforming to heterogeneous schemas semantically interoperable. The idea is to make the knowledge embedded in the schema structures interoperable and explicit by transforming the schemas into a shared, event-based representation of knowledge about the real world. This enables and simplifies accurate reasoning services such as cross-domain semantic search, browsing, and recommending. A case study of transforming three different schemas and datasets is presented. An implemented knowledge-based recommender system utilizing the results in the semantic portal CULTURESAMPO was found useful in a preliminary user study.

## 1 Introduction

Different heterogeneous data formats, metadata schemas, and ontologies, such as Dublin Core [2], CIDOC CRM [3], ULAN<sup>1</sup>, and ABC [9], are in use for describing resources, such as documents, persons, artifacts, and web pages. The heterogeneity of metadata schemas and vocabularies causes problems when aggregating content for end-users with an integrated view of the data [7].

The problem of schema heterogeneity can be addressed on a *syntactic level* by deriving new schemas as extensions of existing ones, or by aligning metadata elements with each other. For example, VRA<sup>2</sup> extends Dublin Core elements in a compatible way by adding additional elements. CIDOC CRM [3] is an ontology developed as an underlying schema into which other metadata schemas in the cultural domain can be transformed for interoperability. On a *semantic level*, the domain ontologies whose resources can be used as values of metadata schema elements [19] can be used for enhancing interoperability [10]. To deal with problems of incompatible domain ontologies, ontology mapping and alignment or a shared upper domain ontology [4, 15] can be used.

In the semantic portal MUSEUMFINLAND [10], a method was presented for transforming heterogeneous database content into a single Dublin Core -like metadata schema for representing metadata about cultural artefacts. By mapping literal metadata element values onto resources of globally shared domain ontologies, semantic interoperability

---

<sup>1</sup> [http://www.getty.edu/research/conducting\\_research/vocabularies/ulan/](http://www.getty.edu/research/conducting_research/vocabularies/ulan/)

<sup>2</sup> <http://www.vraweb.org/projects/vracore4/index.html>

between different content sources was achieved, and intelligent services based on the shared metadata schema could be provided to end-users. When applying this approach to publishing cultural contents of various kinds in the semantic portal CULTURESAMPO [11], the following problems were encountered:

1. *Using heterogeneous metadata schemas.* In cross-domain applications the content is described using different kinds of metadata schemas that are already in everyday use in different domains. Enforcing content providers to use one standard is not feasible but rather the portal system has to make the schemas interoperable.
2. *Mismatch between metadata and knowledge representation formats.* The elements used in schemas have been designed from a content indexing and cataloguing point of view. When used for reasoning, other forms of knowledge representation would be more appropriate in many cases. For example, we may know that the *dc:creator* (*dc* refers to the Dublin Core metadata schema namespace) of a *painting* and a *house* is a certain person, say *John Smith*. However, from the knowledge representation viewpoint, *dc:creator* is not an appropriate property [6], because its meaning is relational referring to either a *painting* or a *building* event involving several participants. This knowledge is not available for the computer to reason about unless the different meanings of the binary property *dc:creator* in the different cases are explicated.
3. *Complexity of reasoning with multiple schemas.* Ontologies are developed for reasoning tasks [16]. When using multiple heterogeneous metadata schemas, the number of reasoning rules explodes if a different set of rules has to be specified for each schema separately. For example, the fact that a person is born somewhere at a certain time may be represented in metadata schemas in numerous ways, say with properties *placeOfBirth* and *timeOfBirth*, or with a *birth* event with the properties *time* and *place*. Harmonization of these representations enables simpler reasoning procedures that are independent of the metadata schemas used.

This paper presents an approach to deal with these problems. First, a new method for obtaining semantic interoperability of metadata conforming to *several heterogeneous schemas* is presented. We present a simple generic knowledge representation scheme underlying the metadata schemas based on knowledge about *events taking place in the real world*, such as painting an art work, manufacturing a chair, or being born at a place at a certain time. The idea of event-based knowledge representations has been successfully applied in many fields of artificial intelligence, such as natural language processing [122], image content description [19], and knowledge representation [20]. In our case, we employ the idea for obtaining semantic interoperability between heterogeneous metadata schemas by transforming metadata into a shared underlying event-based scheme. Second, it is shown that implicit knowledge embedded in the metadata schema structures exists. During the metadata transformation, this implicit knowledge can be made *explicit* for the machines to reason about by using the shared event-based knowledge representation scheme. It is argued that in this way more “intelligent” services to end-users can be implemented with less complex rules.

In the following, we first present a simple event-based model for representing metadata of the heterogeneous schemas. Second, methodological guidelines are presented

**Table 1.** Upper-level relations in the event-based knowledge representation schema

Relation	Meaning	Super-relation	Relation category	Domain	Range
agent	Initiates or performs the activity.	participant	thematic role	perdurant	concept
patient	Undergoes some change as a result of the activity.	participant	thematic role	perdurant	concept
instrument	Is used as an instrument in the activity.	participant	thematic role	perdurant	concept
goal	Is a goal of the activity.	participant	thematic role	perdurant	concept
place	Is a place of the activity.	participant	thematic role	perdurant	concept
time	Is a time of the activity.	participant	thematic role	perdurant	concept
participant	Other participant role of the perdurant concept.		thematic role	perdurant	concept
quality	Is a quality / qualifier of the entity		quality relation	concept	concept
partOf	Is a part of the entity		part name	concept	concept

for specifying the transformation from metadata schemas into the event-based model. A case study of transforming three different metadata schemas is presented. The knowledge explication method has been tested and used in practice in the semantic portal CULTURESAMPO [11] to enable metadata schema interoperability and for creating a semantic recommender system to demonstrate benefits of the approach in a real life application.

## 2 An Event-Based Model for Representing Metadata

In our approach a distinction is made between a *domain ontology* and *event-based metadata* conforming to an *event-based knowledge representation schema* (figure 1). The domain ontology describes the concepts specific to a certain domain, and the ontology can be divided into upper-level concepts and more specific concept hierarchies [9]. The event-based knowledge representation scheme specifies a way to represent heterogeneous metadata schemas using the domain ontology. The metadata is represented by instantiating domain ontology concepts and assigning relations between the instances with respect to the event-based knowledge representation schema.

### 2.1 Domain Ontology

For the domain ontology we use an ontology, such as DOLCE [4], SUMO [15], ABC [9] or YSO [12], which makes the distinction between major ontological upper categories such as perdurants, endurants, location concepts, and temporal concepts. Our particular interest is the distinction between perduring and enduring concepts' behavior in time [4]. Enduring concepts, such as *person*, *chair* or *car*, preserve their identity in time while perduring concepts refer to things that live in time; they are activities or events, such as running, swimming or raining. These concepts are used for instantiating events with thematic roles in the event-based knowledge representation schema.

### 2.2 Event-Based Knowledge Representation Schema

Our event-based schema introduces relations enabling representation of the original metadata as events with associated thematic roles and quality roles, an idea proposed in the fields of knowledge representation, natural language processing, and discourse

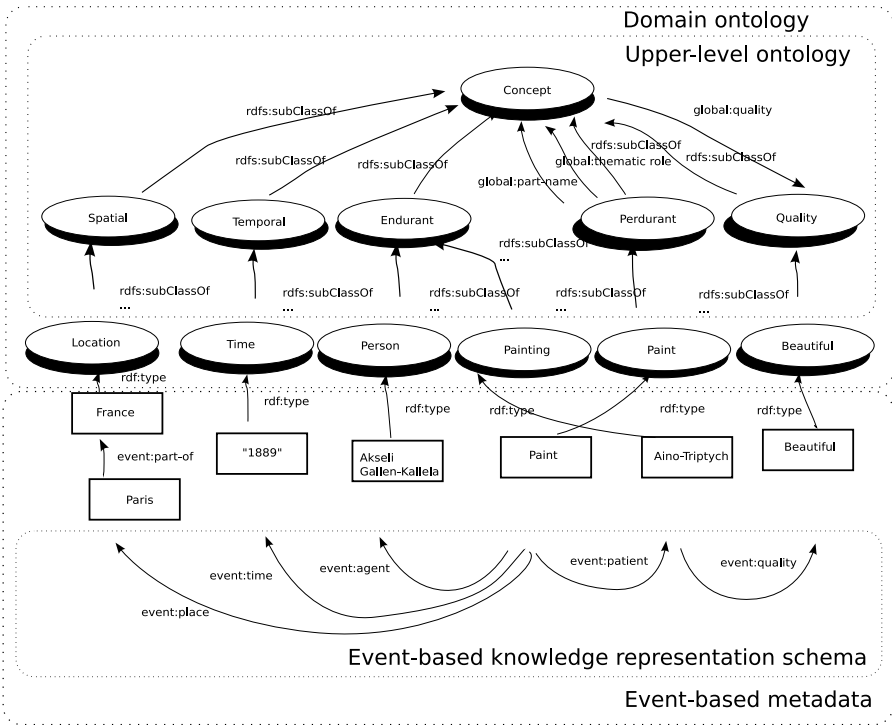


Fig. 1. Event-based model for representing metadata

modeling [11,22,20]. Table 1 presents the nine roles used in our event-based knowledge representation schema, a subset of the thematic role model of Sowa [20]. In addition to the thematic roles of perdurants, we have used the properties *partOf* and *quality* applicable to all concepts in the model.

Masolo et al. [14] propose that a concept based on a relational role is in fact a perduring concept. For example, the relation *manufacturingPlace* actually refers to the perduring concept *manufacturing* and the role *place*. Based on this notion, the relation can be represented as an event frame that consists of an instance of a perduring concept, a set of instances of participating concepts, and a set of relations between these instances. In the following a method for transforming metadata relations into events is presented.

### 3 Transforming Metadata Schemas to Event-Based Schema

In this section methodological guidelines are given to facilitate the event-based knowledge representation schema of heterogeneous metadata schema representations. First, the criteria for metadata schema classification using a set of meta-properties are given. Second, the method for schema explication is presented. Finally, the steps to perform the explication are shown.

**Table 2.** Examples of classification of relations

relation type	meta-properties	relation
non-relational	$-R - F$	person
relational	$-R + F$	teacher
quality	$+R + F$	color
part name	$+R - F$	wheelOf

### 3.1 Criteria for Relation Classification

To address the problem of semantic heterogeneity in metadata schemas we have followed the classification criteria of Guarino [6] and the closer analysis of relations by Masolo et al. [14]. These criteria are used to define the dependencies of the relations used in the metadata schemas. Guarino defines four different relation types: (1) relational role; (2) non-relational role; (3) quality and (4) part name. Two meta-properties are used to classify the relations: semantic rigidity and foundedness.

1. *Foundedness.* In order for a concept  $x$  to be founded on another concept  $y$ , any instance  $a$  of  $x$  has to be necessarily associated to an instance  $b$  of  $y$  which is not related to  $a$  by any *partOf* relation. In other words, the instances of  $x$  cannot exist as such except in a more comprehensive unity where they are associated to some other object. For example, *son* is founded since sons exist only within the framework of a family, where they are associated to their parents. On the other hand, the existence of *person* is essentially independent.
2. *Rigidity.* A concept is semantically rigid if it contributes to the very identity of its instances, in such a way that, if  $a$  is an  $x$  in a particular situation, it has to keep to be an  $x$  in any possible situation in order to keep its identity. For instance, an animal can cease to be a pup while still being a dog: *animal* and *dog* are semantically rigid, *pup* is not.

The relation types for relations are based on rigidity and foundedness of the relation. We denote rigidity with  $+R$ , anti-rigidity with  $-R$ , foundedness with  $+F$  and anti-foundedness with  $-F$ .

Table 2 shows different relation types with examples. According to Guarino [6], an entity is considered to be a non-relational role when it is a unary predicate that does not have a natural relational interpretation. More formally, a non-relational role is a relation that is anti-rigid and anti-founded. For example, the entity *person* is a non-relational role, because it is a unary predicate that does not have an extension to any other concept in its natural interpretation.

An entity is a relational role when it is a unary predicate that has natural relational interpretation. More formally, a relation is a relational role if it is founded and anti-rigid. For example, the entity *teacher* actually refers to a *teaching* activity having the person (teacher) as an *agent* and a person (student) as a *patient*, but is represented as a binary role between the two entities. A relation is considered to be a quality if it is rigid and founded and if an instance of the entity is a predicable entity [6].

A clear distinction between qualities and other types of relations is that the interpretation of a quality is that they are predicable by themselves (i.e. may be names of

predicates), but the same does not apply to other roles [6]. For example, a quality *color* can be name of a predicate and the value of the predicable instances are also qualities, such as *red*, *blue* or *green*.

part names are relations that are not founded, but are rigid. For example, a *wheel* of a *car* can exist independently of a *car*, but may be a relevant feature of a *car* in particular cases. Part names are described with a simple *partOf* relation. For a more complex meronymy we refer to [17].

### 3.2 A Method for Explicating Schema Knowledge

To enable the interoperability between the heterogeneous metadata schemas they have to be explicated using an event-based schema. The novel idea in our work is to use the domain ontology as a basis for describing—at the same time—the semantics of the metadata schema elements and the content descriptions of the resources, i.e. the values of the metadata schema slots. This approach provides interoperability between schema and domain semantics.

The method is based on what we call *explication* of metadata schemas. The input for applying the method is a set of metadata schemas  $MS$ , a domain ontology  $DO$ , and metadata  $MD$  conforming to  $MS$ . The output is event-based metadata  $EM$  that is metadata  $MD$  represented in a event-based knowledge representation scheme  $KS$  that is more suitable for reasoning tasks than  $MS$ . The method (for our case study schemas) consists of the following steps:

1. Classify each relation  $e(x, y)$  in a metadata schema  $ms$  in  $MS$  according to the foundedness ( $+/-F$ ) and rigidity ( $+/-R$ ) criteria.
2. Explication rules for each metadata schema relation  $e(x, y)$  in  $ms$  are:
  - (a) If  $e(x, y)$  is a non-relational role ( $-R - F$ ), then define  $rdf : type(x, y)$  relation such that  $y$  is a concept in  $DO$ .
  - (b) If  $e(x, y)$  is a relational role ( $-R + F$ ), then create an instance  $p$  of a selected perduring concept in  $DO$ , and create a set of thematic roles  $tr(p, y)$  or  $tr(p, x)$  or quality roles  $qr(p, y)$  such that  $y$  is an instance of a concept in  $DO$ . Add  $event : hasEvent(x, p)$ , which ensures that the description is connected to an original annotation source, e.g. a document. (In our case schemas the meaning of each  $e(x, y)$  can be explicated with one event.)
  - (c) If  $e(x, y)$  is a quality role ( $+R + F$ ) (e.g., property “colour”), and  $e$  does not exist in  $DO$ , then explicate its meaning by selecting a concept  $q$  in  $DO$  such that  $rdf : isDefinedBy(e, q)$  (e.g., class “colour”).
  - (d) If  $e(x, y)$  is a part name relation ( $+R - F$ ), then define  $partOf(x, y)$  relation and create statement  $rdfs : subPropetyOf(e(x, y), partOf(x, y))$ .
3. Transform metadata  $MD$  (conforming to  $MS$ ) into  $EM$  (conforming to  $KS$ ) by using the transformation rules.

Figure 2 illustrates an example of the metadata schema explication. The left side of the figure shows a part of an original metadata description from the ULAN dataset of the Getty Foundation. The relation *birthPlace* is first classified using the rigid and foundedness criteria and resolved to be a relational role. The explication against the

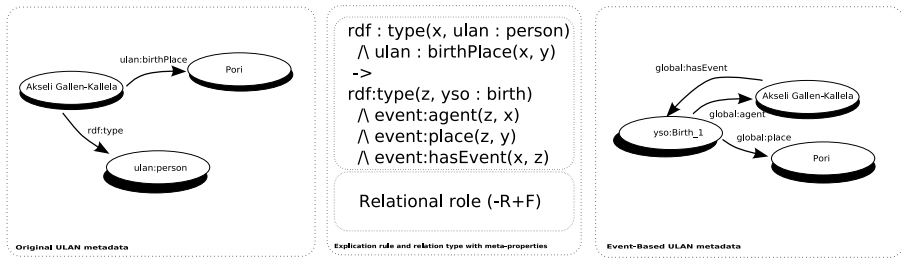


Fig. 2. An example of a metadata explication

event-based knowledge representation schema is made using the YSO [12] domain ontology. An explication rule where the instance of a perduring concept *birth* is related to the place of the birth using thematic role *place*, is derived. Finally, the right side of the figure shows the resulting event-based metadata.

### 3.3 Benefits of the Schema Explication

We argue that schema explication leads to the following benefits. (1) Semantic interoperability of syntactically different schemas can be obtained by defining the meaning of metadata schemas in terms of the underlying domain ontology concepts. This enables the usage of the transitive subsumption hierarchies of the domain ontology in reasoning. (2) It is possible to exploit additional semantic reasoning by explicating the hidden implicit semantics of metadata schemas. This is achieved by more explicit descriptions of the relational roles in terms of domain ontologies. For example, the relation *manufacturingPlace* can be explicated using the concept *manufacturing* and relation *place*. (3) Knowledge representation at a more foundational level reduces the number of different properties to be dealt with, which leads to simpler and more general reasoning. The number of relational roles in original schemas can be exponential, e.g. any perduring concept and role pair is possible. (4) The problem of aligning different metadata schemas onto each other becomes easier by using a canonical representation model. The number of pairwise mappings between  $n$  schemas is  $O(n * (n - 1)/2)$ , but there are only  $O(n)$  mappings between the schemas and the event-based knowledge representation model. To test our hypotheses, we next discuss a case study of applying the metadata explication method for three different schemas used in the semantic portal CULTURESAMPO.

## 4 Three Case Studies

A case study using three different metadata schemas and metadata was conducted: (1) Descriptions of artifacts conforming to the Dublin Core -like metadata schema of MUSEUMFINLAND, (2) descriptions of paintings conforming to the CIDOC Conceptual Reference Model (CRM) [3] used in the Finnish National Gallery and (3) descriptions of artists conforming to the ULAN. The domain ontology used was the



**Table 3.** Representative relation types and explication rules in Finnish museum dataset

row	relation	relation type	classification criteria	explication rules
1	mf:museumName(x,y)	quality	+R + F	$mf : museumName(x,y) \rightarrow$ $rdf : isDefinedBy(mf : museumName(x,y), yso : name)$
2	mf:museumUrl(x,y)	quality	+R + F	$mf : museumUrl(x,y) \rightarrow$ $rdf : isDefinedBy(mf : museumUrl(x,y), yso : identifier)$
3	mf:objectType(x,y)	non-relational	-R - F	$mf : objectType(x,y) \rightarrow rdf : type(x,y)$
4	mf:name(x,y)	quality	+R + F	$mf : name(x,y) \rightarrow rdf : isDefinedBy(mf : name(x,y), yso : name)$
5	mf:manufacturingPlace(x,y)	relational	-R + F	$rdf : type(x, mf : museumItem) \wedge mf : manufacturingPlace(x,y) \rightarrow$ $rdf : type(z, yso : manufacturing) \wedge event : place(z,y) \wedge rdf :$ $type(y, yso : place) \wedge event : patient(z,x) \wedge event : hasEvent(x,z);$ $rdf : type(x, mf : painting) \wedge mf : manufacturingPlace(x,y) \rightarrow$ $rdf : type(z, yso : paint) \wedge event : place(z,y) \wedge rdf : type(y, yso :$ $place) \wedge event : patient(z,x) \wedge event : hasEvent(x,z)$
6	mf:creator(x,y)	relational	-R + F	$rdf : type(x, mf : museumItem) \wedge mf : creator(x,y) \rightarrow$ $rdf : type(z, yso : manufacturing) \wedge event : agent(z,y) \wedge event :$ $patient(z,x) \wedge event : hasEvent(x,z)$
7	mf:creator(x,y)	relational	-R + F	$rdf : type(x, mf : painting) \wedge mf : creator(x,y) \rightarrow$ $rdf : type(z, yso : paint) \wedge event : agent(z,y) \wedge event : patient(z,x) \wedge$ $event : hasEvent(x,z)$
8	mf:manufacturing- StartTime(x,y)	relational	-R + F	$rdf : type(x, mf : museumItem)$ $\wedge mf : manufacturingStartTime(x,y) \rightarrow$ $event : time(z,k) \wedge event : startTime(k,y)$
9	mf:manufacturing- EndTime(x,y)	relational	-R + F	$rdf : type(x, mf : museumItem)$ $\wedge mf : manufacturingEndTime(x,y) \rightarrow$ $event : time(z,k) \wedge event : endTime(k,y)$
10	mf:material(x,y)	relational	-R + F	$rdf : type(x, mf : museumItem) \wedge event : material(x,y) \rightarrow$ $rdf : type(z, yso : manufacturing) \wedge event : material(z,y) \wedge event :$ $hasEvent(x,z)$
11	mf:keyword(x,y)	relational	-R + F	$rdf : type(x, mf : museumItem) \wedge mf : keyword(y) \wedge y \in yso :$ $perduring \rightarrow k = y \wedge event : hasEvent(x,k);$ $k \notin yso : perduring \rightarrow rdf : type(k, yso : perduring) \wedge event :$ $hasEvent(x,k);$ $rdf : type(x, mf : museumItem) \wedge mf : keyword(y) \wedge y \in yso :$ $enduring \rightarrow event : participant(k,x)$
12	mf:stylePeriod(x,y)	quality	+R + F	$mf : stylePeriod(x,y) \rightarrow rdf : isDefinedBy(mf$ $: stylePeriod(x,y), yso : stylePeriod)$
13	mf:inCollection(x,y)	part name	+R - F	$mf : inCollection(x,y) \rightarrow event : partOf(x,y) \wedge rdf :$ $type(yso : museumCollection, y) \wedge rdfs : subPropertyOf(mf$ $: inCollection(x,y), event : partOf(x,y))$
14	mf:part(x,y)	part name	+R - F	$mf : part(x,y) \rightarrow event : partOf(x,y) \wedge rdfs : subPropertyOf(mf :$ $part(x,y), event : partOf(x,y))$

General Finnish Ontology YSO [13]. It contains some 20,000 general concepts in ten major facets including perduring objects (e.g. events and activities), enduring objects (e.g. physical things), properties, time, and locations. This lightweight ontology was created based on the General Finnish Thesaurus YSA<sup>3</sup>. The namespace *mf* is used to refer to the MUSEUMFINLAND system, *crm* to CIDOC CRM, *ulan* to ULAN, *event* conforming to our event-based knowledge representation schema and RDF(S) to Resource Description Framework<sup>4</sup>. We use logic programming syntax to express the rules<sup>5</sup>.

**Case Study 1: Finnish Museum Data.** The Finnish museum dataset contains 4453 descriptions of museum items. We analyzed the superset of the relations occurring in the dataset and used the method to explicate the relations. Table 3 describes a selection of typical relations, the classification of the relations, and the rules defined for explication of the relations in the MuseumFinland metadata schema.

<sup>3</sup> <http://vesa.lib.helsinki.fi>

<sup>4</sup> <http://www.w3.org/RDF/>

<sup>5</sup> Dot (.) is used to indicate chained relations.



All relations in the dataset were explicated. On row 1, the relation *mf:museumName* was aligned to *iso:name* in the domain ontology. Another option would have been to extend the domain ontology to contain a sub-class of *iso:name* and align *mf:museumName* to this additional property. On rows 6 and 7 the relational role *mf:creator* is founded by the type of the object. Two separate rules were written. First, the objects typed as paintings were explicated by *iso:paint* activity while the objects typed as *mf:museumItems* were explicated by *iso:manufacturing* activity. On rows 8 and 9 the representation of interval time forced to introduce a simple time object that was able to represent *event:startTime* and *event:endTime* values. On row 11 a more complex rule was written to handle the *mf:keyword* relation. The relation was relational with respect to its values, i.e. the thematic roles were missing. A simple rule was written to predict the missing thematic roles. First, if the *mf:keyword* contained an instance that was in the sub-class hierarchy of *iso:perduring* concepts, then the instance was set to be the perduring object in the event-based description. Otherwise a superclass of this hierarchy, i.e. *iso:perdurant* was instantiated. Other values that were in the sub-class hierarchy of enduring objects were set as the value of the *event:participant* role to the perduring concept instance.

**Case Study 2: CIDOC CRM from the National Gallery of Finland.** The National gallery of Finland dataset conforms to the CIDOC CRM model and contains 553 descriptions of fine arts items. The content descriptions (*crm:isAbout* relation) were originally annotated using the ICONCLASS<sup>6</sup> vocabulary. A pre-processing stage was conducted and the descriptions were transformed to use the YSO ontology using a simple string matching alignment based on lemmatized labels of the concepts. Table 4 describes the partial but representative relations and the explication rules of the dataset.

All of the relations in the dataset were explicated. On row 7 *crm:productionEvent* is directly the perduring concept. In this case a separate alignment to YSO ontology was required. On row 8 the representation of time is again an interval and represented with a CIDOC CRM specific representation schema. On row 11 a new thematic role *technique* was introduced as a sub-property for *participant* to enable more specific correspondence with the original metadata schema. On row 15 the type property of the *crm:depicts* event has a value from CIDOC CRM ontology and therefore requires a separate alignment to the YSO ontology. As noted before, the domain ontology level alignment was performed before the explication.

**Case Study 3: ULAN of Getty Foundation.** A subset of Finnish Artists in the ULAN dataset contains 429 metadata descriptions. A preprocessing stage was conducted and the descriptions were transformed to use the YSO ontology as in case study 2. Table 5 describes partial, but representative set of relations in the dataset and the explication rules. All of the relations in Finnish Artists in ULAN dataset were explicated using the method. Some relations such as *ulan:nationalities* on row 3, *ulan:role* on row 4 and *ulan:gender* on row 5 required domain ontology alignment. This means the values of the relations were from the ULAN domain ontology and were separately aligned to YSO ontology concepts.

<sup>6</sup> <http://www.iconclass.nl/>

**Table 4.** Representative relation types and explication rules in National Gallery of Finland dataset

row	relation	relation type	classification criteria	explication rules
1	crm:hasType(x,y)	non-relational	$-R - F$	$crm : hasType(x,y) \rightarrow rdf : type(x,y)$
2	crm:consistsOf(x,y)	relational	$-R + F$	$rdf : type(x, crm : painting) \wedge crm : consistsOf(y) \rightarrow$ $rdf : type(z, yso : manufacturing) \wedge event : material(z,y) \wedge event :$ $patient(z,x) \wedge rdf : type(y, yso : material) \wedge event : hasEvent(x,z)$
3	crm:hasTitle(x,y)	quality	$+R + F$	$crm : hasTitle(x,y) \rightarrow rdf : isDefinedBy(crm : hasTitle(x,y), yso :$ $title)$
4	crm:hasDimension(x,y)	quality	$+R + F$	$crm : hasDimension(x,y) \rightarrow$ $rdf : isDefinedBy(crm : hasDimension(x,y), yso : dimension)$
5	crm:isAbout. conceptualObject(x,y)	relational	$-R + F$	$rdf : type(x, crm : painting) \wedge crm : isAbout.conceptualObject(y)$ $\wedge y \in yso : perduring \rightarrow k = y \wedge event : hasEvent(x,k);$ $k \notin yso : perduring \rightarrow rdf : type(k, yso : perduring) \wedge event :$ $hasEvent(x,k);$ $rdf : type(x, crm : painting) \wedge crm : isAbout.conceptualObject(y)$ $\wedge y \in yso : enduring \rightarrow event : participant(k,x)$
6	crm:isAbout. actor(x,y)	relational	$-R + F$	$rdf : type(x, crm : painting) \wedge crm : isAbout.actor(x,y) \rightarrow event :$ $agent(k,y) \wedge event : hasEvent(x,z)$
7	crm:isAbout. productionEvent(x,z)	relational	$-R + F$	$crm : isAbout.productionEvent(x,z) \wedge event : patient(z,x) \wedge event :$ $hasEvent(x,z)$
8	crm:isAbout. productionEvent. hasTimeSpan. atSomeTimeWithin(x,y)	quality	$+R + F$	$crm : isAbout.productionEvent.hasTimeSpan.$ $atSomeTimeWithin(x,y) \rightarrow event : time(z,y)$
9	crm:isAbout. productionEvent. tookPlaceAt(x,y)	relational	$-R + F$	$rdf : type(x, crm : painting) \wedge crm : isAbout.productionEvent.$ $tookPlaceAt(x,y) \rightarrow$ $rdf : type(z, yso : paint) \wedge event : place(z,y) \wedge event : hasEvent(x,z)$
10	crm:isAbout. productionEvent. carriedOutBy(x,y)	relational	$-R + F$	$rdf : type(x, crm : painting) \wedge isAbout.productionEvent.$ $carriedOutBy(x,y) \rightarrow$ $event : agent(z,y) \wedge event : hasEvent(x,z)$
11	crm:isAbout. productionEvent. usedGeneralTechnique(x,y)	quality	$+R + F$	$rdf : type(x, crm : painting) \wedge crm : isAbout.productionEvent.$ $usedGeneralTechnique(x,y) \rightarrow$ $event : technique(z,y) \wedge event : hasEvent(x,z)$
12	crm:wasUsedFor. activity. generalPurpose(x,y)	relational	$-R + F$	$rdf : type(x, crm : painting) \wedge crm : isAbout.$ $productionEvent.generalPurpose(x,y) \rightarrow$ $event : goal(z,y) \wedge event : hasEvent(x,z)$
13	crm:depicts(x,y)	relational	$-R + F$	$rdf : type(x, crm : painting) \wedge crm : depicts(x,y) \rightarrow$ $rdf : type(z2, yso : depict) \wedge event : patient(z2,y) \wedge event :$ $hasEvent(x,z2)$
14	crm:depicts. informationCarrier. about(x,y)	relational	$-R + F$	$crm : depicts.informationCarrier.about(x,y) \rightarrow$ $event : effector(z2,y) \wedge event : hasEvent(x,z2)$
15	crm:depicts. informationCarrier. type(x,y)	non-relational	$-R - F$	$crm : depicts.informationCarrier.type(x,k) \rightarrow$ $rdf : type(m,k)$
16	crm:isDocumentedIn. document(x,y)	relational	$-R + F$	$rdf : type(crm : painting, x) \wedge crm : isDocumentedIn.$ $document(x,y) \rightarrow$ $rdf : type(z3, yso : documenting) \wedge event : patient(z3,x) \wedge event :$ $hasEvent(x,z3)$

**Table 5.** Representative relation types and explication rules in ULAN dataset

row	relation	relation type	classification criteria	explication rules
1	ulan:name(x,y)	quality	$+R + F$	$ulan : name(x) \rightarrow rdf : isDefinedBy(ulan : name(x,y), yso : name)$
2	ulan:alternativeName(x,y)	quality	$+R + F$	$ulan : alternativeName(x,y) \rightarrow$ $rdf : isDefinedBy(ulan : alternativeName(x,y),$ $yso : additionalName)$
3	ulan:nationalities(x,y)	quality	$+R + F$	$ulan : nationalities(x,y) \rightarrow$ $rdf : isDefinedBy(ulan : nationalities(x,y), yso : nationalities)$
4	ulan:role(x,y)	non-relational	$-R - F$	$ulan : role(x,y) \rightarrow rdf : type(x,y)$
5	ulan:gender(x,y)	quality	$+R + F$	$ulan : gender(x,y) \rightarrow rdf : isDefinedBy(ulan : gender(x,y), yso :$ $gender)$
6	ulan:birthPlace(x,y)	relational	$-R + F$	$rdf : type(x, ulan : person) \wedge ulan : birthPlace(x,y) \rightarrow rdf :$ $type(z, yso : birth) \wedge event : agent(z,x) \wedge event : location(z,y) \wedge event :$ $hasEvent(x,z)$
7	ulan:deathPlace(x,y)	relational	$-R + F$	$rdf : type(x, ulan : person) \wedge ulan : deathPlace(x,y) \rightarrow rdf :$ $type(z2, yso : death) \wedge event : agent(z2,x) \wedge event : location(z2,y) \wedge$ $event : hasEvent(x,z2)$
8	ulan:studentOf(x,y)	relational	$-R + F$	$rdf : type(x, ulan : person) \wedge ulan : studentOf(x,y) \rightarrow rdf :$ $type(z3, yso : teaching) \wedge event : agent(z3,y) \wedge event : patient(z3,x) \wedge$ $event : hasEvent(x,z3)$

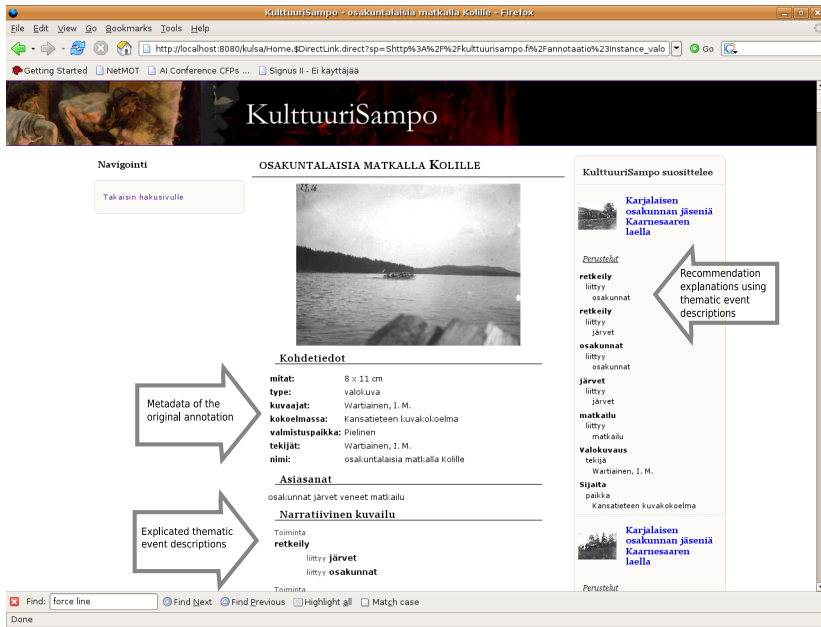


Fig. 3. User interface of CULTURESAMPO recommendation system

#### 4.1 Implemented Use Case

The method and the case studies presented above have been implemented in the CULTURESAMPO prototype portal [11]. Explication rules were written for each schema using the Java-based Prolog system Prova<sup>7</sup>.

Figure 3 illustrates the user interface of the portal showing a page about a photograph concerning a student union traveling to the Koli mountain in Karelia. On the right side the system gives recommendation links to other content items with explanations such as “hiking related to a student association” and “traveling related to a student association”. The event-based system gives these links because the image describes a “hiking” event with a “student association” and “lake” in participant roles. The method also gives links to content items that are “stored” in same collection, “photographed” by the same person, etc.

The recommendation system has been empirically evaluated by seven users and in total seventy metadata description pairs. The precision of the method using the event-based knowledge-base was 82 per cent. For a complete description of the recommendation method and the empirical evaluation and results see [18].

## 5 Discussion

Recent work on schema matching using ontologies suggests that no common or a minimal ontological commitment is needed [5] and that it is unrealistic to assume that there

<sup>7</sup> <http://www.prova.w/>

will be an agreement of one or even a small set of ontologies [16]. As a result, ontologies and metadata schemas will be developed by communities without global standardization. To overcome the interoperability problem, additional representation formalisms defining the inter-ontology or inter-metadata schema mappings have been proposed, as reviewed in [21].

In this paper we have proposed an approach that utilizes domain ontologies and an event-based knowledge representation schema to enable heterogeneous metadata interoperability. Methodological guidelines to explicate schema and metadata content in terms of events were presented and applied successfully to three highly heterogeneous metadata schemas. To utilize the resulting event-based knowledge representation, a semantic recommender system in the semantic portal CULTURESAMPO was implemented and tested [18]. In this practical use case the usefulness of the event-based approach was shown in the form of an intuitive user interface, a standardized reasoning procedure, and enhanced relevance precision.

While the case study presented in this paper confirmed that the event-based knowledge representation schema was able to represent all of the needed implicit metadata, some difficulties were encountered when using the explication method. Some of the relations referred to local domain ontology resources that had to be mapped separately onto YSO concepts. For example, the *ulan:gender* relation in the ULAN dataset referred to *ulan:female* or *ulan:male* and was mapped to the corresponding concepts in YSO. A major problem was how to enrich the metadata with new thematic roles. For example, in the National Gallery of Finland and Finnish museums datasets the content descriptions of the values contained values such as *yso:horse*, *yso:ride*, and *yso:man* without any relation to each other. Thematic roles can easily be resolved by a human annotator, e.g. that a man rides a horse, and not that a horse rides a man. However, selecting the fillers of the roles often requires tacit human knowledge and is difficult for fully automated methods. This problem is a topic of ongoing research (cf. e.g. [11]) and requires further development in the heterogeneous schema integration field.

The idea of using event-based frames for representing knowledge has been explored in many areas of research [22][20][19]. There are a number of metadata models that recognize the importance of events or actions in unambiguously describing resources and facilitating interoperability across the domains [9][3][4]. Many of these ontologies propose an upper-level class hierarchy that can be extended by the domain ontologies. Wache et al. [21] give an extensive survey of current approaches including single, global and hybrid ontology approaches. Semi-automatic methods based on statistical matching of ontologies have been studied [8][16].

An event-based canonical model for metadata in cultural heritage domain is proposed in the CIDOC CRM model [3]. It “provides the definitions and a formal structure for describing the implicit and explicit concepts and relationships used in cultural heritage documentation”<sup>8</sup>. The framework includes 81 classes, such as *crm:Man-Made Object*, *crm:Place*, and *crm:Time-Span*, and a large set of 132 properties relating the entities with each other, such as *crm:Has Time-Span* and *crm:IsIdentifiedBy*. Our approach is different in that our underlying knowledge representation does not concentrate on documentation but describes the underlying real world. The idea is to use existing ontologies

<sup>8</sup> <http://cidoc.ics.forth.gr/>

of thousands of classes describing the world in the annotations. In contrast to our model, CIDOC CRM contains many very specific properties, such as *crm:is\_documented\_in* and *crm:was\_destroyed\_by*. In our approach they are considered highly relational, and are described using events such as “documenting” and “destroying”. In our case study, CIDOC CRM was therefore considered as an example of a heterogeneous metadata schema to be made semantically interoperable with the other metadata schemas.

Our research is part of the National Finnish Ontology Project (FinnONTO) 2003-2007<sup>9</sup>, funded mainly by the National Technology Agency (Tekes) and a consortium of 37 companies and public organizations.

## References

1. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The Berkeley Framenet project. In: Proceedings of the 17th international conference on Computational linguistics, Proceedings of the 36th annual meeting on Association for Computational Linguistics, vol. 1 (1998)
2. Baker, T.: A grammar of Dublin Core. D-Lib Magazine 6(10) (2000)
3. Doerr, M.: The CIDOC conceptual reference module: an ontological approach to semantic interoperability of metadata. AI Magazine 24(3), 75–92 (2003)
4. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening ontologies with DOLCE. In: Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, pp. 166–181 (2002)
5. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing. International Journal of Human-Computer Studies 43(5-6), 907–928 (1995)
6. Guarino, N.: Concepts, attributes and arbitrary relations: some linguistic and ontological criteria for structuring knowledge bases. Data and Knowledge Engineering 8(3) (1992)
7. Hakimpour, F., Geppert, A.: Resolving semantic heterogeneity in schema integration: an ontology based approach. In: Proceedings of FOIS 2001 (2001)
8. Huma, Z., Jaffar-Ur, M., Iftikhar, N.: An ontology-based framework for semi-automatic schema integration. Journal of Computer Science and Technology 20(6), 788–796 (2005)
9. Hunter, J.: Enhancing the semantic interoperability of multimedia through a core ontology. IEEE Transactions on Circuits and Systems for Video Technology 13(1), 49–58 (2003)
10. Hyvönen, E., Mäkelä, E., Salminen, M., Valo, A., Viljanen, K., Saarela, S., Junnila, M., Kettula, S.: MuseumFinland – Finnish museums on the semantic web. Journal of Web Semantics 3(2), 25 (2005)
11. Hyvönen, E., Ruotsalo, T., Häggström, T., Salminen, M., Junnila, M., Virkkilä, M., Haaramo, M., Mäkelä, E., Kauppinen, T., Viljanen, K.: CultureSampo–Finnish culture on the semantic web: The vision and first results. In: Developments in Artificial Intelligence and the Semantic Web - Proceedings of the 12th Finnish AI Conference STeP 2006, October 26-27 2006 (2006)
12. Hyvönen, E., Valo, A., Komulainen, V., Seppälä, K., Kauppinen, T., Ruotsalo, T., Salminen, M., Ylisalmi, A.: Finnish national ontologies for the semantic web - towards a content and service infrastructure. In: DC 2005. Proceedings of International Conference on Dublin Core and Metadata Applications (November 2005)
13. Hyvönen, E., Viljanen, K., Mäkelä, E., Kauppinen, T., Ruotsalo, T., Valkeapää, O., Seppälä, K., Suominen, O., Alm, O., Lindroos, R., Känsälä, T., Henriksson, R., Frosterus, M., Tuominen, J., Sinkkilä, R., Kurki, J.: Elements of a national semantic web infrastructure - case study finland on the semantic web (invited paper). In: IEEE ICSC 2007. Proceedings of the First International Semantic Computing Conference, Irvine, California, IEEE Computer Society Press, Los Alamitos (forth-coming, 2007)

<sup>9</sup> <http://www.seco.tkk.fi/projects/finnonto/>

14. Masolo, C., Guizzardi, G., Vieu, L., Bottazzi, E., Ferrario, R.: Relational roles and quaindividuals. In: Proceedings of the AAAI Fall Symposium on Roles, an interdisciplinary perspective, Hyatt Crystal City, Arlington, Virginia. (2005)
15. Niles, I., Pease, A.: Towards a standard upper ontology. In: FOIS 2001. Proceedings of the international conference on Formal Ontology in Information Systems, Ogunquit, Maine, pp. 2–9. ACM Press, New York, NY, USA (2001)
16. Noy, N.F.: Semantic integration: a survey of ontology-based approaches. *SIGMOD Rec.* 33(4), 65–70 (2004)
17. Odell, J.J.: Six sifferent kinds of composition. *Journal Of Object-Oriented Programming* 5(8) (1994)
18. Ruotsalo, T., Hyvönen, E.: A method for determining ontology-based semantic relevance. In: DEXA 2007. Proceedings of the 18th International Conference on Database and Expert Systems Applications, Springer, Heidelberg (2007)
19. Schreiber, A.T., Dubbeldam, B., Wielemaker, J., Wielinga, B.: Ontology-based photo annotation. *IEEE Intelligent Systems* 16(3) (2001)
20. Sowa, J.F.: *Knowledge Representation: Logical, Philosophical and Computational Foundations*. Brooks/Cole, Pacific Grove, CA (2000)
21. Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Hübner, S.: Ontology-based integration of information — a survey of existing approaches. In: Stuckenschmidt, H. (ed.) *IJCAI 2001. Workshop: Ontologies and Information Sharing*, pp. 108–117 (2001)
22. Zarri, G.P.: NKRL, a knowledge representation tool for encoding the meaning of complex narrative texts. *Natural Language Engineering* 3(2) (1988)

# Evaluating the Semantic Web: A Task-Based Approach

Marta Sabou<sup>1</sup>, Jorge Gracia<sup>2</sup>, Sofia Angeletou<sup>1</sup>,  
Mathieu d'Aquin<sup>1</sup>, and Enrico Motta<sup>1</sup>

<sup>1</sup> Knowledge Media Institute (KMi), The Open University, United Kingdom  
{r.m.sabou,s.angeletou,m.daquin,e.motta}@open.ac.uk

<sup>2</sup> IIS Department, University of Zaragoza, Spain  
jgracia@unizar.es

**Abstract.** The increased availability of online knowledge has led to the design of several algorithms that solve a variety of tasks by harvesting the Semantic Web, i.e., by dynamically selecting and exploring a multitude of online ontologies. Our hypothesis is that the performance of such novel algorithms implicitly provides an insight into the quality of the used ontologies and thus opens the way to a task-based evaluation of the Semantic Web. We have investigated this hypothesis by studying the lessons learnt about online ontologies when used to solve three tasks: ontology matching, folksonomy enrichment, and word sense disambiguation. Our analysis leads to a suit of conclusions about the status of the Semantic Web, which highlight a number of strengths and weaknesses of the semantic information available online and complement the findings of other analysis of the Semantic Web landscape.

## 1 Introduction

The recent growth of the Semantic Web [19] and the appearance of semantic search engines such as Swoogle [11] and WATSON [9] that allow quick access to online knowledge has had a considerable impact on the design of Semantic Web applications. Indeed, there is a trend to move away from applications relying on a small amount of manually selected semantic sources towards a *new generation of Semantic Web tools* which dynamically select, reuse and combine a multitude of heterogeneous, online available ontologies [20,21]. This paradigm of *harvesting the Semantic Web* has also inspired novel ways of performing a variety of tasks. For example, Alani [1] proposes a method for *ontology learning* that relies on cutting and pasting modules from online ontologies relevant to a set of keywords. In [14] the authors rely on online ontologies to *disambiguate the senses* of keywords used in a search engine query. Dynamically selected online ontologies play the role of background knowledge in *ontology matching* [26,27] or can be used to *semantically enrich folksonomy tag spaces* [4,31]. The experimental evaluations of these algorithms [4,14,27] are still at an early stage, but, nevertheless, they provide strong evidence that the Semantic Web has reached a



critical point where it can be used as a valuable source of knowledge to perform a variety of tasks.

Our hypothesis is that an important benefit of such novel algorithms lies in their potential use for *evaluating the Semantic Web*. Indeed, because they reuse a multitude of online ontologies, they can provide valuable insights into the qualitative aspects of these ontologies such as their suitability for a task, the properties of their vocabularies or the quality of their conceptual structure.

Such a task-based evaluation of the Semantic Web complements current efforts for evaluating (online) ontologies. Ontology evaluation has been a core research topic from the early stages of the Semantic Web leading to a set of approaches [6,16] distributed in two major categories. On the one side, a few approaches exist, which are based on the manual assessment of a set of ontology design criteria (e.g., Onto-Clean [15]). On the other side, there are many automatic approaches, which evaluate different aspects of an ontology (e.g., vocabulary, conceptual structure) by relying on different views of what constitutes a good “quality” ontology [28]. For example, the quality of an automatically learnt ontology can be judged in terms of its similarity to a manually constructed ontology or to the corpus from which it was extracted. Or, adopting a task-based view, the quality of an ontology can also be judged with respect to the performance of a task that uses it [25].

With the growth of the Semantic Web, the focus of ontology evaluation efforts has shifted towards *online ontologies*. Ontology selection methods [28] rely on evaluating ontology aspects such as popularity [7,11,23], similarity to a domain or set of keywords [2,7,11,23] and the richness of the internal structure [2,7]. Furthermore, several overviews of the Semantic Web as a whole focus on the totality of online ontologies. Existing studies assess the size and growth rate of online knowledge [10,19], as well as emerging trends in the adoption and use of representation languages and their primitives [5,8,34]. While these findings are important, they do not give an insight into the suitability of online ontologies to be used for certain tasks. Hence, inspired by the paradigm introduced in [25], we propose to perform a task-based evaluation of the Semantic Web by analyzing the performance of novel algorithms that harvest it.

We test the feasibility and usefulness of such a task based evaluation approach by detailing the lessons we have learnt about the quality of online ontologies when they were employed to solve three different tasks: ontology matching (Section 2), folksonomy tag-space enrichment (Section 3) and query disambiguation (Section 4). We conclude in Section 5 with a number of observations about the status of the Semantic Web that support our hypothesis and are complementary to findings provided by similar studies of online ontologies [5,8,10,19,34].

## 2 Case Study 1 - Ontology Matching

Ontology matching is the task of determining the relations that hold between the entities of two ontologies [30]. In [26] we proposed a *new paradigm to ontology matching* which relies on harvesting the Semantic Web: it derives semantic mappings by dynamically selecting, exploiting, and combining multiple and



heterogeneous online ontologies. For example, when matching two concepts labeled *Researcher* and *AcademicStaff*, the matcher would 1) identify (at run-time, during matching) online ontologies that can provide information about how these two concepts inter-relate and then 2) combine this information to infer the mapping. We distinguish two different strategies for deriving mappings [26]. In strategy S1 the mapping can be provided by a *single* ontology (e.g., stating that  $Researcher \sqsubseteq AcademicStaff$ ). In strategy S2 a mapping can be derived by reasoning with information spread over several ontologies (e.g., that  $Researcher \sqsubseteq ResearchStaff$  in one ontology and that  $ResearchStaff \sqsubseteq AcademicStaff$  in another). We performed a large scale investigation and evaluation of this matching paradigm in [27] which provided a variety of insights into the quality of online ontologies as described next.

## 2.1 Experimental Data and Results

For experimental purposes we used two large, real life thesauri<sup>1</sup>. The United Nations Food and Agriculture Organization (FAO)'s **AGROVOC** thesaurus, version May 2006, consists of 28.174 descriptor terms (i.e., preferred terms) and 10.028 non-descriptor terms (i.e., alternative terms). The United States National Agricultural Library (NAL) Agricultural thesaurus **NALT**, version 2006, consists of 41.577 descriptor terms and 24.525 non-descriptor terms. We used both alternative and preferred terms in our experiments.

The matching process performed by using strategy S1 (see implementation details in [26]) resulted in a total of 6687 mappings (2330 subclass, 3710 superclass and 647 disjoint relations) obtained by dynamically selecting, exploring and combining 226 online ontologies. Fig. 1 shows the contribution of each of these ontologies to the alignment in terms of the number of mappings to which each ontology contributed and the percentage that this number represents.

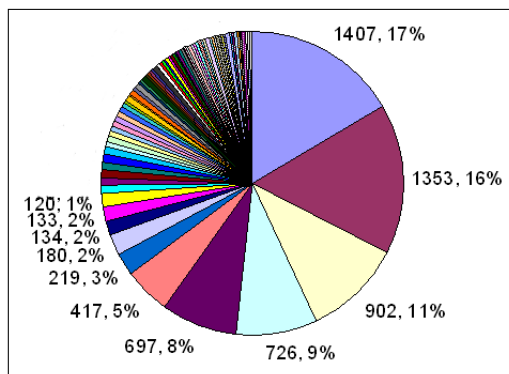


Fig. 1. Contribution of the online ontologies used by S1 to the alignment

**Conclusion C1: Online ontologies are useful for the matching task.** Based on these results, we can already conclude that online ontologies are useful to solve real life matching tasks. Indeed, if combined appropriately, they can provide a large amount of mappings between the matched ontologies. Moreover, in the next section we will show that the quality of the knowledge provided by the Semantic Web allows us to produce a performance comparable with the best performers among alternative approaches to ontology matching.

<sup>1</sup> This data set was used in the “OAEI’06 *food* Thesaurus Mapping Task”, <http://www.few.vu.nl/~wrvhage/oaiei2006/>

**Table 1.** Evaluation results

	Group 1	Group 2	Agreed by All
Correct	586	666	525
False	346	299	217
Don't know	68	35	10
Precision	63%	69%	70%

## 2.2 Quality of Online Ontologies

According to [25], the essence of a task based evaluation is that the quality of an ontology correlates with the performance of the task in which it is employed. In the case of matching this means that the precision of the alignment is an indication of the quality of online ontologies explored to derive it.

To assess the quality of the knowledge provided by online ontologies we performed a manual assessment of 1000 mappings (i.e., 15% of the alignment). We relied on six members of our lab working in the area of the Semantic Web, and thus familiar with ontologies and ontology modeling. We performed two parallel evaluations of the sample mappings (i.e., each mapping has been evaluated by two different evaluators). The participants evaluated each mapping as *Correct*, *False* or “*Don't know*” when they could not judge the correctness of the statement. We computed the precision of the obtained alignment as the ratio of *Correct* mappings over all the evaluated mappings (i.e., those evaluated either as *Correct* or *False*) and obtained precision values of 63% and 69% for the two groups (see Table 1). To level out any differences, we also computed the precision of the fraction of the alignment on which both groups agreed (i.e., 742 mappings, 74%). We consider the so obtained precision value of 70% as a typical baseline performance that can be achieved by harvesting online knowledge.

A manual inspection of the 217 false mappings on which both groups agreed revealed that 114 (i.e., 53%) are due to our simplistic anchoring mechanism (i.e., finding concepts in online ontologies that correspond to the matched concepts). For example, in Table 2, *c\_6443* labeled with *Rams* and referring to an *uncastrated adult male sheep*<sup>2</sup> is put in correspondence with a similarly labeled concept (*ram*), but which means *Random Access Memory* in the context of the online ontology. An anchoring mechanism that could prevent us from deriving these false mappings (thus reducing their number to 103) will imply an increase in precision from 70% to 87%.

To fully understand the significance of these values, it is important to compare them to the performance of other background knowledge based techniques. However, the precision values we found in the literature were reported on different data sets, therefore we can consider them only as indicative, and in addition only S-Match reports on recall values [13]. The technique of Aleksovski et al. was evaluated on a Gold Standard of mappings for 200 concepts and produced a precision of 76% [3]. The matching techniques proposed by van Hage et al. reach precision values of 53% - 75% when exploring a domain specific textual

<sup>2</sup> Definition from WordNet2.1.

**Table 2.** Examples of several types of false mappings. For each mapping we show the names and labels of the matched concepts, the reasoning which lead to deriving the mapping and the online ontology from which the mapping was derived.

Error Type	Nr./ %	Examples				
		AGROVOC Concept	Labels	Rel.	NALT Concept	Labels
Anchor	114, 53%	<i>c_6443</i>	Rams, Tups	$\sqsubseteq$	<i>memory</i>	memory
		$O_1:\text{ram} \sqsubseteq O_1:\text{memory}$				
		$O_1 = \text{http://www.arches.uga.edu/~gonen/qos_bilal.owl}$				
Subsumption as generic relation	40, 18%	<i>c_3954</i>	Irrigation	$\sqsubseteq$	<i>agriculture</i>	agriculture
		$O_1:\text{Irrigation} \sqsubseteq O_1:\text{SoilCultivation} \sqsubseteq O_1:\text{Agriculture}$				
		$O_1 = \text{http://sweet.jpl.nasa.gov/ontology/human_activities.owl}$				
Subsumption as part-whole	16, 7%	<i>c_23995</i>	Branches	$\sqsubseteq$	<i>trees</i>	trees
		$O_1:\text{Branch} \sqsubseteq O_1:\text{Tree}$				
		$O_1 = \text{http://site.uottawa.ca/~mkhedr/FuzzyOnto}$				
Subsumption as role	11, 5%	<i>c_11091</i>	Garlic	$\sqsubseteq$	<i>ingredients</i>	ingredients
		$O_1:\text{garlic} \sqsubseteq O_1:\text{vegetable} \sqsubseteq O_1:\text{ingredient}$				
		$O_1 = \text{http://cvs.sourceforge.net/viewcvs.py/instancestore/instancestore/ontologies/Attic/pizza9.daml?rev=1.2}$				
Inaccurate labeling	12, 5%	<i>c_1693</i>	Coal	$\sqsubseteq$	<i>industry</i>	industry
		$O_1:\text{coal} \sqsubseteq O_1:\text{industry}$				
		$O_1 = \text{http://www.aifb.uni-karlsruhe.de/WBS/meh/mapping/data/russia1a.rdf}$				
Different View	12, 5%	<i>c_2943</i>	Fishes	$\sqsupseteq$	<i>lobsters</i>	lobsters
		$O_1:\text{Fish} \sqsupseteq O_1:\text{MarineInvertebrate} \sqsupseteq O_1:\text{Crustacean} \sqsupseteq O_1:\text{Lobster}$				
		$O_1 = \text{http://139.91.183.30:9090/RDF/VRP/Examples/tap.rdf}$				

resource [32]. Therefore, the 70% precision value (which could be potentially increased to 87%) obtained by *dynamically selecting and combining multiple, heterogeneous and generic ontologies* correlates with the precision of the other two techniques (75% - 76%) when exploring a *single, high-quality, domain specific resource* (i.e., DICE [3], CooksRecipes.coms Cooking Dictionary [32]).

**Conclusion C2: Online ontologies have a good quality and lead to high precision alignments.** We conclude that online ontologies have a good enough quality to lead to alignments with a high precision value which can rival alignments obtained with manually selected, domain specific resources (e.g., ontologies, texts). Even more, our findings show that online ontologies don't only have a high quality when taken stand alone, but most importantly their *combined use* also results in high quality alignments.

### 2.3 Frequent Errors in Online Ontologies

While the use of online ontologies generally leads to correct mappings there are also cases when false mappings are derived. Understanding the causes of false mappings can provide another interesting insight into the quality of online ontologies, namely, *the typical errors that lead to false mappings*. Our inspection of the false mappings revealed that 91 (i.e., 42%) are a direct consequence of the following types of errors in online ontologies (see Table 2).

**Subsumption used to model generic relations.** One of the most common errors was the use of subsumption as a way to model the fact that there exists some type of relation between two concepts, e.g., *Survey*  $\sqsubseteq$  *Marketing*, *Irrigation*  $\sqsubseteq$  *Agriculture*, *Biographies*  $\sqsubseteq$  *People*. This case leads to 40 false mappings (i.e., 18%).

**Subsumption used to model part-whole relations.** Subsumption is also used in several ontologies to model part-whole relations. These ontologies resulted in 16 (7%) incorrect mappings, e.g., *Branch*  $\sqsubseteq$  *Tree*, *Leaf*  $\sqsubseteq$  *Plant*.

**Subsumption used to model roles.** We found 11 false mappings (5%) derived because roles were incorrectly modeled as subclass relations, for example, that *Garlic*, *Leek*  $\sqsubseteq$  *Ingredient* (in fact, *Leek* is a *Vegetable* but in some contexts it plays the the role of an ingredient).

**Inaccurate labeling.** We also found 12 cases (5%) when a correct subclass relation introduced errors due to the inaccurate labeling of its concepts. For example,  $O_1$ <sup>3</sup> states that *coal*  $\sqsubseteq$  *industry*, where *coal* refers to *coal industry* rather than the concept of *Coal* itself. Similarly, for *Database*  $\sqsupseteq$  *Enzyme* in  $O_1$ <sup>4</sup>, *Enzyme* refers to an *enzyme database* rather than describing the class of all enzymes.

**Different Views.** Finally, some of the explored ontologies adopted a certain view on the relation of two concepts that was not in concordance with the context of the mapping and/or the perspective of the evaluators. For example, TAP considers *lobsters* kinds of *Fishes*, a perspective with which none of the evaluators agreed.

**Conclusion C3: Online ontologies contain modeling errors which hamper the quality of the alignment.** Most errors are due to the incorrect use of subsumption to model generic relations, roles and meronymy.

## 2.4 Contradictory Statements in Online Ontologies

The novelty of techniques that harvest the Semantic Web lies in their ability to combine information from multiple, different ontologies. As such, they need to deal with potentially contradictory information supplied by different sources. For example, in the case of ontology matching, contradictory mapping relations could be derived between two concepts by relying on different ontologies. The question is how frequent this phenomenon is, i.e., *do different online ontologies lead to contradictory mappings between two given terms?*

To answer this question, we ran a modified variant of S1: for every pair of concept labels we derive mappings from *all* the online ontologies that mention them. While we have discovered mappings between a high number of label pairs (6425), the number of cases when contradictory mappings are derived is surprisingly low and accounts to only six pairs (see Table 3).

<sup>3</sup> <http://www.aifb.uni-karlsruhe.de/WBS/meh/mapping/data/russia1a.rdf>

<sup>4</sup> <http://mensa.sl.iupui.edu/ontology/Database.owl>

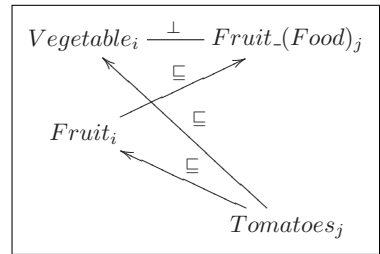
**Table 3.** Contradictory statements in online ontologies

AGROVOC label	NALT label	Nr. Subclass relations ( $\sqsubseteq$ )	Nr. Superclass relations ( $\supseteq$ )	Nr. Disjunct relations ( $\perp$ )
fruit	tomato	0	3	1
sea	river	0	1	2
energy	light	0	1	1
meat	seafood	0	2	12
mushroom	pizza	1	0	1
sea	ocean	1	1	0

**Conclusion C4: Only few online ontologies contain contradictory relations between two given concepts.** Our preliminary observations indicate that the correct mapping can normally be filtered out with simple statistical means: the most frequently derived relation is likely to be correct.

## 2.5 Inconsistencies in Multiple Mappings Drawn from Different Ontologies

In the previous section we have only looked at a rather basic form of contradiction, where contradictory relations have been explicitly stated between two items. As pointed out, these cases appear to be very infrequent. However, if we go beyond relations between two items and look at a number of mappings as a whole, then inconsistencies arise more frequently. Fig. 2 provides such an example, where  $Vegetable_i$  is discovered to be disjoint with  $Fruit\_ (Food)_j$ ,  $Tomatoes_j$  is a subclass of both concepts and thus *unsatisfiable*: there cannot be any instance of  $Tomatoes_j$ , since it would have to belong to two disjoint classes at the same time.

**Fig. 2.** Example of incoherence in mappings

As already said, this phenomenon of generating sets of inconsistent mappings was more pronounced than expected. Indeed, our automatic incoherence detection mechanism has identified 306 base incoherences that corrupted the entire alignment. A few mappings between very generic concepts with many subclasses (e.g.,  $Foods \sqsubseteq Products$ ,  $People \sqsubseteq Agents$ ) caused the majority of the incoherences. Fortunately, these can be isolated and disposed of automatically using reasoning mechanisms, thus leading to the improvement of the entire alignment.

**Conclusion C5: Because different ontologies rely on different views or different contexts, they may contain contradictory information, leading to inconsistent sets of mappings.**

### 3 Case Study 2 - Folksonomy Tagspace Enrichment

Social tagging systems<sup>5</sup> are highly successful due to the ease of the tagging process: users need neither to have prior knowledge or specific skills to use them [18,35], nor to rely on a priori agreed structure or shared vocabulary. While folksonomies (i.e., lightweight structures that emerge from the tag space) are easy to create, they only weakly support content retrieval since they are agnostic to the relations between their tags: a search for *mammal* ignores all resources that are not tagged with this specific word, even if they are tagged with semantically related terms such as *lion*, *cow*, *cat*. Most approaches which address this problem [12,29,35] identify clusters of *implicitly* related tags (e.g., that *mammal* and *lion* are related). Specia and Motta [31] go one step further by proposing to make the semantic relations between tags *explicit* (e.g., that *mammal* is more generic than *lion*). They envision a semantic enrichment algorithm which complies with the paradigm of harvesting the Semantic Web by dynamically exploring and combining multiple online ontologies to derive explicit relations among implicitly interrelated tags.

A simplified version of the enrichment algorithm has been experimentally investigated in [4] by relying on the same implementation of relation discovery as used for ontology matching in [26] (i.e., strategy S1). Given a set of implicitly related tags, the prototype identifies subsumption and disjointness relations between them and constructs a semantic structure based

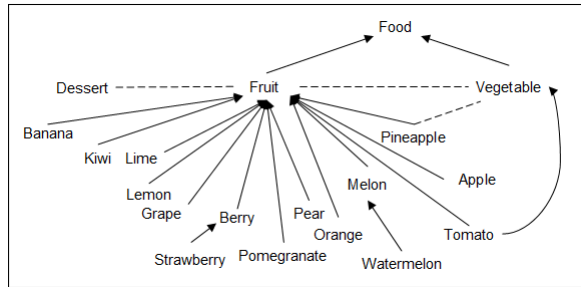


Fig. 3. Semantically enriched tag cluster for *Fruit*

on these relations. The first experiments on tag sets identified in [31] led to suboptimal results due to (1) the small size of the clusters (3-5 tags), (2) the low coverage of certain tag types in online ontologies and (3) the limitation of the software (it only identifies subsumption relations while most tags were related through generic relations). Therefore, we ran a second set of experiments on larger tag clusters identified with the Flickr API<sup>6</sup> around a handful of terms from domains that are well-covered by online ontologies. In these cases the process resulted in rich knowledge structures, such as for the tag cluster of *Fruit* in Fig. 3 (dotted lines denote disjointness). The general conclusion of the study is that while online ontologies can indeed be used to semantically enrich folksonomies, some of their characteristics hamper the process, as described next.

<sup>5</sup> E.g., Flickr (<http://flickr.com/>), del.icio.us (<http://del.icio.us/>)

<sup>6</sup> <http://www.flickr.com/services/api/flickr.tags.getRelated.html>

**Conclusion C6: Online ontologies only weakly cover certain tag categories, as follows:**

**Novel terminology.** Folksonomies are social artifacts built by large masses of people. They dynamically change to reflect the latest terminology in several domains and therefore greatly differ from ontologies which are usually developed by small groups of people and evolve much slower. As such, it is not surprising that many of the tags used in folksonomies, e.g., {`ajax`, `css`}, have not yet been integrated into ontologies<sup>7</sup>. Identifying such novel terminology has a great potential for the Semantic Web as it could represent a first step towards updating existing ontologies.

**Scientific terminology** (e.g., plant taxonomy) can only rarely be found in online ontologies. This could change however if large ontologies containing such information (e.g., AGROVOC) would be made available online.

**Multilingual tags.** Both Flickr and Del.icio.us (but especially Flickr) contain tags from a variety of languages and not only English. These tags are usually hard to find on the Semantic Web because the language coverage of the existing ontologies is rather low. Indeed, statistics<sup>8</sup> performed on a large collection of online ontologies (1177) in the context of the OntoSelect library [7] indicate that 63% of these ontologies contain English labels, while a much smaller percentage contains labels in other languages (German 13.25%, French 6.02%, Portuguese 3.61%, Spanish 3.01%).

**Photographic jargon.** Because Flickr is a photo annotation and sharing site, many tags reflect terms used in photography, such as {`nikon`, `canon`, `closeup`}. Unfortunately, this domain is weakly covered in the Semantic Web.

Our study also found that, like in the case of ontology matching (C5), online ontologies can reflect different views and when used in combination can lead to inconsistencies in the derived structures. For example, the structure in Fig. 3 states that *Fruit* is disjoint with *Dessert*. The validity of this statement depends on the point of view we adopt since some would argue that fruits are desserts. Such different views can have more serious consequences. For example, *Tomato* is considered to be both a *Fruit* and a *Vegetable*. The first statement is valid in a biological context, since a tomato is the fruit of a tomato plant, however, normally one would classify tomatoes as types of vegetables. While such different views can co-exist, the fact that *Fruit* and *Vegetable* are disjoint makes the derived semantic structure inconsistent.

## 4 Case Study 3 - Word Sense Disambiguation

The goal of the Word Sense Disambiguation (WSD) task is to identify the appropriate sense of a word in a given context. Usually this task involves identifying a set of possible senses and then filtering out the right one based on some similarity algorithms. Existing approaches [17,22,24] exploit a given lexical resource

<sup>7</sup> At the time of our experiments, March 2007.

<sup>8</sup> <http://olp.dfki.de/OntoSelect/w/index.php?mode=stats>



(mainly WordNet) or ontology (small set of ontologies) as sources of word senses and then rely on one or more particular ontologies or corpora to compute semantic measures. Because they require the selection of appropriate knowledge sources a priori, these approaches are not suitable in cases when the domain of the words to be disambiguated is determined at run time. This limitation is addressed by a novel, unsupervised, multi-ontology WSD method [14] which 1) relies on dynamically identified online ontologies as sources for candidate word senses and 2) employs algorithms that combine information available both on the Semantic Web and the Web in order to compute semantic measures.

For example, suppose that we want to disambiguate *Java* in the context of “*Indonesia Java*”. In a first step, the algorithm identifies a set of possible senses<sup>9</sup> for each keyword by exploring online ontologies<sup>10</sup> and combines highly similar senses to avoid redundancies. Table 4 shows the candidate senses for *Java* and their characterization by their synonyms (i.e., Level 0) and superterms (Level 1, in this case direct hypernyms). A second, disambiguation step consists in computing a *Google based semantic relatedness* between *Indonesia* and each involved term (e.g., *Indonesia*  $\leftrightarrow$  *Java*, *Indonesia*  $\leftrightarrow$  *Island*) and combining the obtained values into a final [0,1] range score. The highest score indicates the most appropriate sense, i.e., *Java*  $\sqsubseteq$  *Island* in our case. While the large scale evaluation of this method is still in progress, we can already summarize some qualitative conclusions about the characteristics of online ontologies.

Table 4. Disambiguation of *Java* in the context of “*Indonesia Java*”

#Sense	Type	Level 0	Level 1	Score
1	concept	Java	island	<b>0.387</b>
2	concept	java, coffee	drink	0.251
3	concept	java	programming language	0.116

**Conclusion C7: Online ontologies provide a good source for word sense definitions.** A major benefit of relying on multiple, online ontologies is that a much larger set of keyword senses can be discovered than in cases when few, predefined resources are used. For example, many traditional methods fail to disambiguate *developer* in “*UML handbook for developers*” because WordNet2.1 does not contain the word *UML* (acronym of Unified Modeling Language), neither the intended meaning of *developer* as someone who develops software. This information is however available in online ontologies: *UML* is a concept in the Book<sup>11</sup> ontology (subsumed by *SoftwareDesigns*), and *developer* is a property described as “*Developer of software*” in the DOAP<sup>12</sup> ontology. As evident from Table 5<sup>13</sup> this extra information discovered at runtime in DOAP is crucial for identifying the appropriate sense for *developer*.

<sup>9</sup> Defined by the ontological context of the term: synonyms, hypernyms etc.

<sup>10</sup> In addition to WordNet or any other local resource.

<sup>11</sup> <http://islab.hanyang.ac.kr/damls/Book.daml>

<sup>12</sup> <http://usefulinc.com/ns/doap>

<sup>13</sup> Level 1 here contains direct hypernyms for concepts and domains for properties.



**Table 5.** Disambiguation of *developer* in the context “*UML handbook developer*”

#Sense	Type	Source	Description	Level 0	Level 1	Score
1	property	DOAP	“Developer of software for the project”	developer	project	<b>0.293</b>
2	concept	WordNet	“photographic equipment ...”	developer	photographic equipment	0.239
3	concept	WordNet	“someone who develops real estate”	developer	creator	0.230

**Conclusion C8: Disambiguation results are influenced by modeling errors in online ontologies.** However, due to its nature, the algorithm, is only partially affected by the typical ontology errors described in Section 2.3 (see Table 6). Indeed, the disambiguation algorithm uses part of the ontological context that characterizes a sense (e.g., subsumption, generic relations) in order to restrict the semantic field of the sense and to distinguish it from other senses of the same word. Such ontological information is used as a basis for relatedness computation and not exploited through formal reasoning as in the case of ontology matching. Therefore, the algorithm is not affected by the quality of formal modeling. For example, to characterize *branch* in its biological sense, an incorrectly modeled part-whole relation ( $Branch \sqsubseteq Tree$ ) could lead to the same disambiguation result as using a correct subsumption ( $Branch \sqsubseteq Stalk$ ). Also, *agriculture* could be an acceptable context to distinguish *irrigation* as *supplying dry land with water* from its medical sense. We conclude that error types 1 and 2 do not affect intrinsically the algorithm. On the other hand, the last three types of errors which associate a given term with other terms that do not reflect its sense have a major influence on the algorithm. For example, in  $Enzyme \sqsubseteq Database$ , the inaccurate labeling could give unpredictable results in the computed semantic measures. Also, the user could have different views from some online ontologies (error 5) and thus obtain an undesired result.

**Table 6.** Sensitivity of disambiguation algorithm to frequent ontology errors

	Error Type	Example	Effect on algorithm?
1	Subsumption as generic relation	$Irrigation \sqsubseteq Agriculture$	No
2	Subsumption as part-whole	$Branch \sqsubseteq Tree$	No
3	Subsumption as role	$garlic \sqsubseteq ingredient$	Yes
4	Inaccurate labeling	$enzyme \sqsubseteq database$	Yes
5	Different view	$lobster \sqsubseteq fish$	Yes

**Conclusion C9: Many online ontologies have a weak internal structure** and thus hamper the performance of the method. For example, few online ontologies contain synonyms or non-taxonomic relations. We even found ontologies containing no relations at all. As a result, our algorithm can identify richly (e.g., extracted from WordNet) as well as poorly defined senses for the same word. Such uneven semantic characterization has a negative effect on the algorithm

(which was built to compare similarly rich descriptions of senses) and can lead to suboptimal results. This insight in the general quality of online ontologies lead us to envision two important future changes. First, our tool should only rely on senses extracted from semantically rich ontologies which could be identified using a ranking mechanism such as AKTiveRank [2]. Second, the semantic measures we use should adapt to ontological contexts of variable richness (e.g., glosses should be given a high importance in ontologies with a poor taxonomy but rich in descriptions).

**Conclusion C10: Parsing errors and broken links further hamper the functioning of the method.** For example, from the 602 online ontologies identified for describing 25 terms randomly extracted from a list of frequently used keywords<sup>14</sup>, 252 (42%) could not be correctly parsed into Jena<sup>15</sup> models due to parsing errors or broken links. Without being conclusive, this limited example illustrates the proportion of the problem.

## 5 Conclusions and Future Work

The hypothesis put forward in this paper is that novel algorithms which harvest online knowledge can facilitate a task based evaluation of the Semantic Web. Accordingly, we report on quality characteristics of online ontologies determined by analyzing the experimental results of three algorithms which solve divers tasks: ontology matching, folksonomy enrichment and WSD.

The major conclusion that we derive based on the content of our observations is that *online ontologies have a great potential for being used in combination to solve a variety of real life tasks*. Indeed, combining knowledge from multiple ontologies lead to a broad range of high quality mappings (C1) and to more word sense definitions during WSD (C7). In the case of ontology matching, we could also experimentally prove that the obtained alignment had a high precision, despite relying on more than 200 ontologies (C2). There are, however, some undesired effects caused by combining knowledge from multiple sources. Even if only in very few cases, contradicting statements can be obtained about two given concepts (C4). Then, the first two case studies were affected by the fact that online ontologies often reflect different views which can lead to incoherent knowledge structures when combined (C5). Overall, however, these findings deliver an important message: even at this early stage of development, the Semantic Web is a powerful source of background knowledge that can be exploited to successfully tackle real world tasks.

Besides providing task-centric conclusions, our approach also lead to observations about other aspects of online ontologies. At a syntactic level, several ontologies cannot be accessed due to parsing errors and broken links (C10). Second, regarding their vocabularies, online ontologies provide a weak coverage of certain types of folksonomy tags, such as novel terms, multilingual tags or scientific terms (C6). Third, we gained insight into major issues with the quality

<sup>14</sup> [http://www.google.com/press/zeitgeist\\_monthly.html](http://www.google.com/press/zeitgeist_monthly.html)

<sup>15</sup> <http://jena.sourceforge.net/>

of the knowledge structures of online ontologies. We found that many have a weak (or no) structure and thus hampered the WSD method (C9). Even more worryingly, we identified a set of modeling errors, mostly related to the misuse of subsumption relations, which affected (to different degrees) both the formal reasoning based matching algorithm (C3) and the WSD process (C8).

While our conclusions provide a better understanding of the current state of the Semantic Web (complementary with the conclusions of other similar studies [5,8,10,19,34]), they could further benefit the research community as follows. First, we consider them as a proof that a task based evaluation is feasible and useful, thus supporting the hypothesis of the paper. Therefore, we wish to provide a more formal model for performing evaluations in this manner. Second, our findings have highlighted the need for novel evaluation methods that are capable to automatically identify more subtle characteristics such as the quality of the modeling [33]. Finally, these findings are valuable knowledge for those who wish to (re-)design algorithms that harvest the Semantic Web in a way that they maximally benefit from this rich and growing online knowledge repository.

**Acknowledgements.** This work was funded by the OpenKnowledge IST-FF6-027253 and NeOn IST-FF6-027595 projects, the CICYT TIN2004-07999-C02-02 project and by the Government of Aragon-CAI grant IT11/06.

## References

1. Alani, H.: Position Paper: Ontology Construction from Online Ontologies. In: Proc. of WWW (2006)
2. Alani, H., Brewster, C., Shadbolt, N.: Ranking Ontologies with AKTiveRank. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
3. Aleksovski, Z., Klein, M., ten Katen, W., van Harmelen, F.: Matching Unstructured Vocabularies using a Background Ontology. In: Staab, S., Svátek, V. (eds.) EKAW 2006. LNCS (LNAI), vol. 4248, Springer, Heidelberg (2006)
4. Angeletou, S., Sabou, M., Specia, L., Motta, E.: Bridging the Gap Between Folksonomies and the Semantic Web: An Experience Report. In: Proc. of the ESWC Bridging the Gap between Semantic Web and Web 2.0 Workshop (2007)
5. Bechhofer, S., Volz, R.: Patching Syntax in OWL ontologies. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, Springer, Heidelberg (2004)
6. Brank, J., Grobelnik, M., Mladenic, D.: A survey of ontology evaluation techniques. In: Proc. of the Conf. on Data Mining and Data Warehouses (2005)
7. Buitelaar, P., Eigner, T., Declerck, T.: OntoSelect: A Dynamic Ontology Library with Support for Ontology Selection. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, Springer, Heidelberg (2004)
8. d'Aquin, M., Baldassarre, C., Gridinoc, L., Angeletou, S., Sabou, M., Motta, E.: Characterizing Knowledge on the Semantic Web with WATSON, (submitted for peer review) (2007)
9. d'Aquin, M., Sabou, M., Dzbor, M., Baldassarre, C., Gridinoc, L., Angeletou, S., Motta, E.: WATSON: A Gateway for the Semantic Web. In: ESWC Poster (2007)

10. Ding, L., Finin, T.: Characterizing the Semantic Web on the Web. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
11. Ding, L., Pan, R., Finin, T., Joshi, A., Peng, Y., Kolari, P.: Finding and Ranking Knowledge on the Semantic Web. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
12. Begelman, G., Keller, P., Smadja, F.: Automated Tag Clustering: Improving search and exploration in the tag space. In: WWW 2006. Proc. of the Collaborative Web Tagging Workshop (2006)
13. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: Discovering Missing Background Knowledge in Ontology Matching. In: ECAI 2006. LNCS (LNAI), vol. 4386, Springer, Heidelberg (2006)
14. Gracia, J., Trillo, R., Espinoza, M., Mena, E.: Querying the Web: A Multiontology Disambiguation Method. In: Proc. of ICWE (2006)
15. Guarino, N., Welty, C.A.: An Overview of OntoClean. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, Springer, Heidelberg (2004)
16. Hartmann, J., Sure, Y., Giboin, A., Maynard, D., Suarez-Figueroa, M.C., Cuel, R.: Methods for ontology evaluation. Knowledge Web Deliverable. In: Methods for ontology evaluation. Knowledge Web Deliverable D1.2.3 (2005)
17. Hassell, J., Aleman-Meza, B., Arpinar, I.B.: Ontology-Driven Automatic Entity Disambiguation in Unstructured Text. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
18. Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: Information Retrieval in Folksonomies: Search and Ranking. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)
19. Lee, J., Goodwin, R.: The Semantic Webscape: a View of the Semantic Web. IBM Research Report (2004)
20. Motta, E., Sabou, M.: Language Technologies and the Evolution of the Semantic Web. In: Proc. of the Int. Conf. on Language Resources and Evaluation (2006)
21. Motta, E., Sabou, M.: Next Generation Semantic Web Applications. In: Mizoguchi, R., Shi, Z., Giunchiglia, F. (eds.) ASWC 2006. LNCS, vol. 4185, Springer, Heidelberg (2006)
22. Navigli, R., Velardi, P.: Structural Semantic Interconnections: A Knowledge-Based Approach to Word Sense Disambiguation. IEEE Trans. Pattern Anal. Mach. Intell. 27(7), 1075–1086 (2005)
23. Patel, C., Supekar, K., Lee, Y., Park, E.K.: OntoKhoj: A Semantic Web Portal for Ontology Searching, Ranking and Classification. In: Proc. of WIDM (2003)
24. Pedersen, T., Banerjee, S., Patwardhan, S.: Maximizing Semantic Relatedness to Perform Word Sense Disambiguation, 2005. Research Report UMSI 2005/25
25. Porzel, R., Malaka, R.: A Task-based Approach for Ontology Evaluation. In: Proc. of the ECAI Workshop on Ontology Learning and Population (2004)
26. Sabou, M., d'Aquin, M., Motta, E.: Using the Semantic Web as Background Knowledge for Ontology Mapping. In: Proc. of the Ontology Matching WS (2006)
27. Sabou, M.: M. d'Aquin, and E. Motta. Using the Semantic Web as Background Knowledge for Ontology Matching: Investigating a New Paradigm (submitted for peer review, 2007)
28. Sabou, M., Lopez, V., Motta, E., Uren, V.: Ontology Selection: Ontology Evaluation on the Real Semantic Web. In: Proc. of the EON Workshop (2006)
29. Schmitz, P.: Inducing Ontology from Flickr Tags. In: WWW 2006. Proc. of the Collaborative Web Tagging Workshop (2006)

30. Shvaiko, P., Euzenat, J.: A Survey of Schema-based Matching Approaches. *Journal on Data Semantics IV* (2005)
31. Specia, L., Motta, E.: Integrating Folksonomies with the Semantic Web. In: *ESWC 2007. LNCS*, vol. 4519, Springer, Heidelberg (2007)
32. van Hage, W., Katrenko, S., Schreiber, G.: A Method to Combine Linguistic Ontology-Mapping Techniques. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005. LNCS*, vol. 3729, Springer, Heidelberg (2005)
33. Völker, J., Vrandečić, D., Sure, Y.: Automatic Evaluation of Ontologies (AEON). In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005. LNCS*, vol. 3729, Springer, Heidelberg (2005)
34. Wang, T.D., Parsia, B., Hendler, J.A.: A Survey of the Web Ontology Landscape. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006. LNCS*, vol. 4273, Springer, Heidelberg (2006)
35. Wu, X., Zhang, L., Yu, Y.: Exploring Social Annotations for the Semantic Web. In: *Proc. of WWW* (2006)

# Using Tableau to Decide Expressive Description Logics with Role Negation

Renate A. Schmidt and Dmitry Tishkovsky

School of Computer Science, The University of Manchester  
{renate.schmidt,dmitry.tishkovsky}@manchester.ac.uk

**Abstract.** This paper presents a tableau approach for deciding description logics outside the scope of OWL DL/1.1 and current state-of-the-art tableau-based description logic systems. In particular, we define a sound and complete tableau calculus for the description logic  $\mathcal{ALBO}$  and show that it provides a basis for decision procedures for this logic and numerous other description logics with full role negation.  $\mathcal{ALBO}$  is the extension of  $\mathcal{ALC}$  with the Boolean role operators, inverse of roles, domain and range restriction operators and it includes full support for nominals (individuals).  $\mathcal{ALBO}$  is a very expressive description logic which subsumes Boolean modal logic and the two-variable fragment of first-order logic and reasoning in it is  $\text{NExpTime}$ -complete. An important novelty is the use of a generic, unrestricted blocking rule as a replacement for standard loop checking mechanisms implemented in description logic systems. An implementation of our approach exists in the METTEL system.

## 1 Introduction

Mainstream description logics systems and ontology web languages provide a rich supply of concept operators, but there is currently little support for complex role operators. This places significant restrictions on the expressiveness of ontology languages and the utility of systems. For example, in the description logic  $\mathcal{ALC}$  (and other popular extensions of  $\mathcal{ALC}$ ) [1] it is possible to define a ‘spam filter’ as a mechanism for filtering out spam emails (i.e.  $x \in \text{spam-filter}$  iff  $x \in \text{mechanism} \wedge \exists y. (x, y) \in \text{filter-out} \wedge y \in \text{spam-email}$ ), and a ‘sound spam filter’ as a spam filter which filters out only spam emails (i.e.  $x \in \text{-sound-spam-filter}$  iff  $x \in \text{spam-filter} \wedge \forall y. (x, y) \in \text{filter-out} \rightarrow y \in \text{spam-email}$ ), by specifying

$$\begin{aligned} \text{spam-filter} &\stackrel{\text{def}}{=} \text{mechanism} \sqcap \exists \text{filter-out.spam-email} \\ (\dagger) \quad \text{sound-spam-filter} &\stackrel{\text{def}}{=} \text{spam-filter} \sqcap \neg \exists \text{filter-out.}\neg \text{spam-email.} \end{aligned}$$

But it is not possible to define a ‘complete spam filter’ as a spam filter which filters out every spam email, i.e.  $x \in \text{complete-spam-filter}$  iff  $x \in \text{spam-filter} \wedge \forall y. y \in \text{spam-email} \rightarrow (x, y) \in \text{filter-out}$ . This can be expressed by the following.

$$(\ddagger) \quad \text{complete-spam-filter} \stackrel{\text{def}}{=} \text{spam-filter} \sqcap \neg \exists \neg \text{filter-out.spam-email}$$

This uses the role negation operator which is not available in  $\mathcal{ALC}$  or description logics that form the basis of OWL DL/1.1. Both  $(\dagger)$  and  $(\ddagger)$  involve universal quantification but of a different kind. In  $(\dagger)$  it is the image elements of a role that are universally quantified, while in  $(\ddagger)$  it is the elements in a concept that are universally quantified. From an applications perspective there is little justification to give preference to one form of universal quantification over the other, since clearly both are useful.  $(\dagger)$  expresses the *necessity* of a property and  $(\ddagger)$  expresses the *sufficiency* of a property. Natural examples of both kinds of universal quantification can be found in many domains and every-day language.

In this paper we are interested in description logics that allow role negation (and can therefore accommodate examples such as the above), but also provide a range of other role operators not usually supported. In particular, we focus on a description logic, called  $\mathcal{ALBO}$ , which is an extension of the description logic  $\mathcal{ALB}$  [6] with singleton concepts, called nominals in modal logic.  $\mathcal{ALB}$  is the extension of  $\mathcal{ALC}$ , in which concepts and roles form a Boolean algebra, and additional operators include inverse of roles and a domain restriction operator.  $\mathcal{ALBO}$  therefore extends  $\mathcal{ALC}$  by union of roles, negation of roles, inverse of roles, and domain as well as range restriction. In addition, it provides full support for ABox individuals and singleton concepts.

None of the current state-of-the-art tableau-based description logic systems are able to handle  $\mathcal{ALBO}$  (or  $\mathcal{ALB}$ ). Because  $\mathcal{ALBO}$  allows full negation of roles, it is out of the scope of OWL DL, OWL 1.1 and most description logic systems including FACT++, KAON2, PELLET, and RACERPRO. A tableau decision procedure for the description logic  $\mathcal{ALCQTb}$  which allows for Boolean combinations of ‘safe’ occurrences of negated roles is discussed in [14]. Safeness essentially implies a ‘guardedness’ property which is violated by unsafe occurrences of role negation. Description logics with full, i.e. safe and unsafe, role negation can be decided however by translation to first-order logic and first-order resolution theorem provers such as MSPASS, SPASS, E and VAMPIRE. The paper [6] shows that the logic  $\mathcal{ALB}$  can be decided by translation to first-order logic and ordered resolution. This result is extended in [4] to  $\mathcal{ALB}$  with positive occurrences of composition of roles.  $\mathcal{ALBO}$  can be embedded into the two-variable fragment of first-order logic with equality which can be decided with first-order resolution methods [3]. This means that  $\mathcal{ALBO}$  is decidable and can be decided using first-order resolution methods.

$\mathcal{ALBO}$  is a very expressive description logic. It subsumes the Boolean modal logic and tense, hybrid versions of Boolean modal logic with the @ operator and nominals. It can also be shown that  $\mathcal{ALBO}$  subsumes the two-variable fragment of first-order logic (without equality) [7]. The following constructs and statements can be handled in  $\mathcal{ALBO}$ .

- Role negation, the universal role, the sufficiency or window operator, the image operator, cross product, and (left and right) cylindrification.
- Role inclusion axioms and role equivalence axioms in the language of  $\mathcal{ALBO}$ .
- Role assertions in the language of  $\mathcal{ALBO}$ .



- Boolean combinations of both concept and role inclusion and equivalence axioms.
- Boolean combinations of concept and role assertions, including negated role assertions.
- Disjoint roles, symmetric roles and serial roles. (It is not difficult to extend our method and results to include full equality handling including reflexive roles, identity and diversity roles, and the test operator.)

$\mathcal{ALBO}$  is in fact very close to the brink of undecidability, because we know that adding (negative occurrences of) role composition to  $\mathcal{ALB}$  takes us into undecidable territory [12].

Since  $\mathcal{ALBO}$  subsumes Boolean modal logic it follows from [9] that the satisfiability problem in  $\mathcal{ALBO}$  is NExpTime-hard. In [5] it is shown that satisfiability in the two-variable first-order fragment with equality is NExpTime-complete. It follows therefore that the computational complexity of  $\mathcal{ALBO}$ -satisfiability is NExpTime-complete.

Few tableau calculi or tableau procedures have been described for description logics with complex role operators, or equivalent dynamic modal logic versions. Ground semantic tableau calculi and tableau decision procedures are presented in [4] for the modal versions of  $\mathcal{ALC}(\sqcup, \sqcap, ^{-1})$ , i.e.  $\mathcal{ALC}$  with role union, role intersection and role inverse. These are extended with the domain restriction operator, to  $\mathcal{ALC}(\sqcup, \sqcap, ^{-1}, \uparrow)$ , in [11]. A semantic tableau decision procedure for  $\mathcal{ALC}$  with role intersection, role inverse, role identity and role composition is described in [10]. None of these tableaux make provision for the role negation operator however. In [13] a sound and complete ground semantic tableau calculus is presented for Peirce logic, which is equivalent to the extension of  $\mathcal{ALB}$  with role composition and role identity. However the tableau is not terminating because reasoning in Peirce logic is not decidable.

In this paper we develop a tableau approach which can decide description logics with the role negation operator. We present a ground semantic tableau approach which decides the description logic  $\mathcal{ALBO}$ . The style of tableau is similar to that of [4, 11, 13] but a notable difference is that our tableau calculi operate only on ground labelled *concept* expressions. This makes it easier in principle to implement the calculi as extensions of existing tableau-based description logic systems which can handle singleton concepts.

In order to limit the number of individuals in the tableau we need a mechanism for detecting periodicity in the underlying interpretations (models). *Standard loop checking* mechanisms are based on comparing sets of (labelled or unlabelled) concept expressions such as subset blocking or equality blocking. Instead of using the standard loop checking mechanisms, our approach uses a new inference rule, the *unrestricted blocking* rule, and equality reasoning. Our approach has the following advantages over standard loop checking.

- It is conceptually simple and easy to implement.
- It is universal and does not depend on the notion of a type.
- It is versatile and enables more controlled model construction in a tableau procedure. For instance, it can be used to construct small models for a



satisfiable concept, including domain minimal models. Our tableau approach has the further advantage that it constructs real models, whereas the tableau procedures for many OWL DL/1.1 description logics construct only pseudo-models (which are not always real models but can be completed to real models).

- Our blocking mechanism generalises to other logics, including full first-order logic.
- It can be simulated in first-order logic provers.

The unrestricted blocking rule corresponds to an unrestricted version of the first-order blocking rule invented by [2], simply called the *blocking rule*. The blocking rule is constrained to individuals  $\ell$  and  $\ell'$  such that the individual  $\ell$  is an ancestor of the individual  $\ell'$ . I.e. in the common branch of  $\ell$  and  $\ell'$ , the individual  $\ell'$  is obtained from  $\ell$  as a result of a sequence of applications of the existential restriction rule. In this form, the rule can be used to simulate standard loop checking mechanisms such as subset blocking and equality blocking.

The structure of the paper is as follows. The syntax and semantics of  $\mathcal{ALBO}$  is defined in Section 2. In Section 3 we define a tableau calculus for  $\mathcal{ALBO}$  and prove that it is sound and complete without the unrestricted blocking rule. Section 4 introduces our blocking mechanism and proves soundness, completeness and termination of the extended tableau calculus. This allows us to define general criteria for decision procedures for  $\mathcal{ALBO}$  and its sublogics which are discussed in Section 5. We conclude in Section 6.

## 2 Syntax and Semantics of $\mathcal{ALBO}$

The syntax of  $\mathcal{ALBO}$  is defined over the signature  $\sigma = (O, C, R)$  of three disjoint alphabets:  $O = \{\ell_0, \ell_1, \dots\}$  the alphabet of individuals (object names),  $C = \{p_0, p_1, \dots\}$  the alphabet of concept symbols, and  $R = \{r_0, r_1, \dots\}$  the alphabet of role symbols. The logical connectives are:  $\neg$  (*negation*),  $\sqcup$  (*union*),  $\exists$  (*existential concept restriction*),  $^{-1}$  (*role inverse*),  $\upharpoonright$  (*domain restriction*),  $\downharpoonright$  (*range restriction*). *Concept expressions* (or *concepts*) and *role expressions* (or *roles*) are defined as follows:

$$\begin{aligned} C &\stackrel{\text{def}}{=} p \mid \{\ell\} \mid \neg C \mid C \sqcup D \mid \exists R.C, \\ R &\stackrel{\text{def}}{=} r \mid R^{-1} \mid \neg R \mid R \sqcup S \mid R \upharpoonright C \mid R \downharpoonright C. \end{aligned}$$

$p$  ranges over the set  $C$ ,  $\ell$  ranges over  $O$ , and  $r$  ranges over  $R$ . The intersection operator  $\sqcap$  on concepts and roles is defined as usual in terms of  $\neg$  and  $\sqcup$ , and the top and bottom concepts are defined by  $\top \stackrel{\text{def}}{=} p \sqcup \neg p$  and  $\perp \stackrel{\text{def}}{=} p \sqcap \neg p$ , respectively, for some concept name  $p$ . The universal restriction operator  $\forall$  is a dual to the existential restriction operator  $\exists$ , specified by  $\forall R.C \stackrel{\text{def}}{=} \neg \exists R.\neg C$ .

Next, we define the semantics of  $\mathcal{ALBO}$ . A *model* (or an *interpretation*)  $\mathcal{I}$  of  $\mathcal{ALBO}$  is a tuple  $\mathcal{I} = (\Delta^{\mathcal{I}}, p_0^{\mathcal{I}}, \dots, \ell_0^{\mathcal{I}}, \dots, r_0^{\mathcal{I}}, \dots)$ , where  $\Delta^{\mathcal{I}}$  is a non-empty

set,  $p_i^{\mathcal{I}}$  is a subset of  $\Delta^{\mathcal{I}}$ ,  $\ell_i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ , and  $r_0^{\mathcal{I}}$  is a binary relation over  $\Delta^{\mathcal{I}}$ . The semantics of concepts and roles in the model  $\mathcal{I}$ , i.e.  $C^{\mathcal{I}}$  and  $R^{\mathcal{I}}$ , is specified by:

$$\begin{aligned} \{\ell\}^{\mathcal{I}} &\stackrel{\text{def}}{=} \{\ell^{\mathcal{I}}\}, & (R^{-1})^{\mathcal{I}} &\stackrel{\text{def}}{=} (R^{\mathcal{I}})^{-1} = \{(x, y) \mid (y, x) \in R^{\mathcal{I}}\}, \\ (\neg C)^{\mathcal{I}} &\stackrel{\text{def}}{=} \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}, & (\neg R)^{\mathcal{I}} &\stackrel{\text{def}}{=} (\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}) \setminus R^{\mathcal{I}}, \\ (C \sqcup D)^{\mathcal{I}} &\stackrel{\text{def}}{=} C^{\mathcal{I}} \cup D^{\mathcal{I}}, & (R \sqcup S)^{\mathcal{I}} &\stackrel{\text{def}}{=} R^{\mathcal{I}} \cup S^{\mathcal{I}}, \\ (R \upharpoonright C)^{\mathcal{I}} &\stackrel{\text{def}}{=} \{(x, y) \mid x \in C^{\mathcal{I}} \text{ and } (x, y) \in R^{\mathcal{I}}\}, \\ (R \downharpoonright C)^{\mathcal{I}} &\stackrel{\text{def}}{=} \{(x, y) \mid y \in C^{\mathcal{I}} \text{ and } (x, y) \in R^{\mathcal{I}}\}, \\ (\exists R.C)^{\mathcal{I}} &\stackrel{\text{def}}{=} \{x \mid \exists y \in C^{\mathcal{I}} (x, y) \in R^{\mathcal{I}}\}. \end{aligned}$$

A *TBox* (respectively *RBox*), is a (finite) set of inclusion statements  $C \sqsubseteq D$  (respectively  $R \sqsubseteq S$ ) which are interpreted in any model  $\mathcal{I}$  as subset relationships, namely  $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$  (respectively  $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ ). An *ABox* is a (finite) set of statements of the form  $\ell : C$  or  $(\ell, \ell') : R$ , called concept assertions or role assertions. A *knowledge base* is a tuple  $(T, R, A)$  of a TBox  $T$ , an RBox  $R$ , and an ABox  $A$ .

In  $\mathcal{ALBO}$  various additional operators can be defined, including:

Top role:	$\nabla \stackrel{\text{def}}{=} r \sqcup \neg r$ (for some role symbol $r$ )
Bottom role:	$\Delta \stackrel{\text{def}}{=} \neg \nabla$
Universal modality:	$\Box C \stackrel{\text{def}}{=} \forall \nabla.C$
Image operator:	$\exists^{-1} R.C \stackrel{\text{def}}{=} \exists R^{-1}.C$
Sufficiency, or window, operator:	$\bar{\nabla} R.C \stackrel{\text{def}}{=} \neg \exists \neg R.C$
Left cylindrification:	$C^c \stackrel{\text{def}}{=} \nabla \downharpoonright C$
Right cylindrification:	${}^c C \stackrel{\text{def}}{=} \nabla \upharpoonright C$
Cross product:	$C \times D \stackrel{\text{def}}{=} C^c \sqcap {}^c D$

Concept assertions can be expressed as concept expressions as follows:  $\ell : C \stackrel{\text{def}}{=} \exists \nabla.(\{\ell\} \sqcap C)$ . (It is clear that  $(\ell : C)^{\mathcal{I}} \neq \emptyset$  iff  $\ell^{\mathcal{I}} \in C^{\mathcal{I}}$  in every model  $\mathcal{I}$ .) A role assertion  $(\ell, \ell') : R$  can be expressed as a concept expression, namely  $(\ell, \ell') : R \stackrel{\text{def}}{=} \ell : \exists R.\{\ell'\}$ , or by the above as a concept expression. We refer to  $\ell : \exists R.\{\ell'\}$ , or a role assertion, as a *link* (between the individuals  $\ell$  and  $\ell'$ ). In addition, concept and role inclusion axioms are definable as concept expressions.

$$\begin{aligned} C \sqsubseteq D &\stackrel{\text{def}}{=} \forall \nabla.(\neg C \sqcup D) \\ R \sqsubseteq S &\stackrel{\text{def}}{=} \forall \nabla.\forall \nabla.(\neg R \sqcup S).\perp \end{aligned}$$

Thus, Boolean combinations of inclusion and assertion statements of concepts and roles are also expressible in  $\mathcal{ALBO}$  as the corresponding Boolean combinations of the concepts which represent these statements. As usual, concept satisfiability in  $\mathcal{ALBO}$  with respect to any knowledge base can be reduced to concept satisfiability with respect to a knowledge base where all TBox, RBox, and ABox are empty. Without loss of generality we therefore focus on the problem of concept satisfiability in  $\mathcal{ALBO}$ .

Often description logics are required to satisfy the unique name assumption. We do not assume it for  $\mathcal{ALBO}$ . This is inconsequential, because the unique name assumption can be enforced by disjointness statements of the form  $\{\ell\} \cap \{\ell'\} \sqsubseteq \perp$  for every distinct pair of nominals that occur in the given knowledge base.

Above we defined the cylindrification operators and cross product in terms of the domain and range restriction operators. In fact, each of the operators in  $\{\uparrow, \downarrow, \cdot^c, \cdot^{\bar{c}}, \times\}$  are interdefinable. Thus, we could have defined  $\mathcal{ALBO}$  as an extension of  $\mathcal{ALCO}(\neg, \sqcup, \neg^1)$  with one of these operators. It can be shown that regardless as to which of these is used to define  $\mathcal{ALBO}$ , problems in  $\mathcal{ALBO}$  are linearly reducible to problems in  $\mathcal{ALCO}(\neg, \sqcup, \neg^1)$ . For instance, suppose  $\mathcal{ALBO}$  is defined as the extension of  $\mathcal{ALCO}(\neg, \sqcup, \neg^1)$  with left cylindrification. The satisfiability of a concept  $C$ , say, in  $\mathcal{ALBO}$  can then be encoded in  $\mathcal{ALCO}(\neg, \sqcup, \neg^1)$  by replacing all occurrences of  $D^c$  in  $C$  by a new role symbol  $q_{D^c}$  uniquely associated with  $D^c$  and adding the definitions  $\neg D \sqsubseteq \forall q_{D^c} \perp$  and  $D \sqsubseteq \neg \exists \neg q_{D^c} \top$  to the knowledge base. In a similar way, problems involving the other operators from  $\{\uparrow, \downarrow, \cdot^c, \cdot^{\bar{c}}, \times\}$  can be linearly encoded in  $\mathcal{ALCO}(\neg, \sqcup, \neg^1)$ . We leave it to the reader to make sharper observations concerning the definability of these operators in description logics.

Let us give our first technical result. The following theorem can be proved using a filtration argument.

**Theorem 1 (Finite Model Property).**  *$\mathcal{ALBO}$  has the finite model property, i.e., if a concept  $C$  is satisfiable, then it has a finite model.*

### 3 Tableau Calculus

Let  $T$  denote a tableau calculus and  $C$  a concept. We denote by  $T(C)$  a finished tableau built using the rules of the calculus  $T$  starting with the concept  $C$  as input. I.e. we assume that all branches in the tableau are expanded and all applicable rules of  $T$  have been applied in  $T(C)$ . As usual we assume that all the rules of the calculus are applied *non-deterministically, to a tableau*. A branch of a tableau is *closed* if a contradiction has been derived in this branch, otherwise the branch is called *open*. The tableau  $T(C)$  is *closed* if all its branches are closed and  $T(C)$  is *open* otherwise. We say that  $T$  is *terminating* (for satisfiability) iff for every concept  $C$  either  $T(C)$  is finite whenever  $T(C)$  is closed or  $T(C)$  has a finite open branch if  $T(C)$  is open.  $T$  is *sound* iff  $C$  is unsatisfiable whenever  $T(C)$  is closed for all concepts  $C$ .  $T$  is *complete* iff for any concept  $C$ ,  $C$  is satisfiable (has a model) whenever  $T(C)$  is open.

Let  $T_{\mathcal{ALBO}}$  be the tableau calculus consisting of the rules listed in Figure [□](#). Inference steps are performed in the usual way. A rule is applied to a set of expressions (often just one expression) in a branch of a tableau, if the expressions are instances of the premises of the rule. Then, in the case of a non-branching rule, the corresponding instances of the conclusions of the rule are added to the branch. A branching rule splits the branch into several branches (here two) and adds the corresponding instances of the conclusions to each branch.

Rules for  $\mathcal{ALCO}$ :

$$\begin{array}{c}
(\perp): \frac{\ell : C, \ell : \neg C}{\perp} \\
(\neg\sqcup): \frac{\ell : \neg(C \sqcup D)}{\ell : \neg C, \ell : \neg D} \\
(\text{sym}): \frac{\ell : \{\ell'\}}{\ell' : \{\ell\}} \quad (\neg\text{sym}): \frac{\ell : \neg\{\ell'\}}{\ell' : \neg\{\ell\}} \\
(\exists): \frac{\ell : \exists R.C}{\ell : \exists R.\{\ell'\}, \ell' : C} (\ell' \text{ is new}) \\
(\text{bridge}): \frac{\ell : \exists R.\{\ell'\}, \ell' : \{\ell''\}}{\ell : \exists R.\{\ell''\}}
\end{array}
\qquad
\begin{array}{c}
(\neg\neg): \frac{\ell : \neg\neg C}{\ell : C} \\
(\sqcup): \frac{\ell : (C \sqcup D)}{\ell : C \mid \ell : D} \\
(\text{mon}): \frac{\ell : \{\ell'\}, \ell' : C}{\ell : C} \quad (\text{refl}): \frac{\ell : C}{\ell : \{\ell\}} \\
(\neg\exists): \frac{\ell : \neg\exists R.C, \ell : \exists R.\{\ell'\}}{\ell' : \neg C}
\end{array}$$

Rules for complex roles:

$$\begin{array}{c}
(\exists\sqcup): \frac{\ell : \exists(R \sqcup S).\{\ell'\}}{\ell : \exists R.\{\ell'\} \mid \ell : \exists S.\{\ell'\}} \\
(\exists^{-1}): \frac{\ell : \exists R^{-1}.\{\ell'\}}{\ell' : \exists R.\{\ell\}} \\
(\exists\mid): \frac{\ell : \exists(R \mid C).\{\ell'\}}{\ell : C, \ell : \exists R.\{\ell'\}} \\
(\exists\setminus): \frac{\ell : \exists(R \setminus C).\{\ell'\}}{\ell' : C, \ell : \exists R.\{\ell'\}} \\
(\exists\neg): \frac{\ell : \exists\neg R.\{\ell'\}}{\ell : \neg\exists R.\{\ell'\}}
\end{array}
\qquad
\begin{array}{c}
(\neg\exists\sqcup): \frac{\ell : \neg\exists(R \sqcup S).C}{\ell : \neg\exists R.C, \ell : \neg\exists S.C} \\
(\neg\exists^{-1}): \frac{\ell : \neg\exists R^{-1}.C, \ell' : \exists R.\{\ell\}}{\ell' : \neg C} \\
(\neg\exists\mid): \frac{\ell : \neg\exists(R \mid C).D}{\ell : \neg C \mid \ell : \neg\exists R.D} \\
(\neg\exists\setminus): \frac{\ell : \neg\exists(R \setminus C).D}{\ell : \neg\exists R.\neg(\neg C \sqcup \neg D)} \\
(\neg\exists\neg): \frac{\ell : \neg\exists\neg R.C, \ell' : \{\ell'\}}{\ell : \exists R.\{\ell'\} \mid \ell' : \neg C}
\end{array}$$

Fig. 1. Tableau calculus  $T_{\mathcal{ALCO}}$  for  $\mathcal{ALCO}$

The first group of rules are standard for  $\mathcal{ALC}$  and reasoning with individuals. The  $(\sqcup)$  rule is the closure rule. The  $(\neg\neg)$  rule removes occurrences of double concept negation. (The rule is superfluous if double negations are eliminated using on-the-fly rewrite rules.) The  $(\sqcup)$  and  $(\neg\sqcup)$  rules are standard rules for handling concept disjunctions. The  $(\text{sym})$ ,  $(\text{mon})$ ,  $(\text{refl})$ , and  $(\text{bridge})$  rules are the equality rules for individuals, which are familiar from hybrid logic tableau systems, and can be viewed as versions of standard rules for first-order equality. The  $(\text{refl})$  rule is formulated perhaps a bit unusually, but the purpose of the premise is to ensure that the rule is realised only for individuals actually occurring in the branch. The  $(\neg\text{sym})$  rule is needed to ensure that any negated singleton concept will eventually appear as a label in a concept assertion. As usual, and in accordance with the semantics of the existential restriction operator, for every existentially restricted concept the  $(\exists)$  rule creates a new individual with this concept and adds a link to the new individual. It is the only rule which generates new individuals in the calculus. The  $(\neg\exists)$  rule is equivalent to the standard propagation rule for universally restricted concept expressions.

The rules in the second group are rules for decomposing complex role expressions. They can be divided into two subgroups: rules for positive existential role occurrences and rules for negated existential role occurrences (in the left and right columns, respectively). Due to the presence of the  $(\exists)$  rule, the rules for positive existential roles can be restricted to role assertions. (On the side we note that the rule  $(\exists\exists)$  can be replaced by this closure rule:  $\ell : \exists\neg R.\{\ell'\}, \ell : \exists R.\{\ell'\}/\perp$ .) Among the rules for negated existential roles, the  $(\exists\exists^{-1})$  rule and the  $(\exists\exists\exists)$  rule are special. The  $(\exists\exists^{-1})$  rule allows the backward propagation of concept expressions along inverted links (ancestor links). The  $(\exists\exists\exists)$  rule is the rule for the sufficiency operator. It expands a universally restricted concept in which the role is negated according to the semantics:  $x \in (\neg\exists\neg R.C)^{\mathcal{I}}$  iff  $\forall y((x, y) \in R^{\mathcal{I}} \vee y \in (\neg C)^{\mathcal{I}})$ . That is,  $\ell'$  is implicitly quantified by a universal quantifier. The effect of the second premise,  $\ell' : \{\ell'\}$ , is to instantiate  $\ell'$  with individuals that occur in the branch. The remaining rules in this subgroup are based on obvious logical equivalences in  $\mathcal{ALBO}$ .

Tableau rules which do not produce new individuals are called *type-completing* rules. In the case of  $T_{\mathcal{ALBO}}$ , with the exception of the  $(\exists)$  rule, all rules are type-completing.

Now, given an input concept  $C$ , a tableau derivation is constructed as follows. First, preprocessing is performed which pushes the role inverse operators toward atomic concepts by exhaustively applying the following role equivalences from left to right.

$$\begin{aligned} (\neg R)^{-1} &= \neg(R^{-1}), & (R \sqcup S)^{-1} &= R^{-1} \sqcup S^{-1}, \\ (R \upharpoonright C)^{-1} &= R^{-1} \upharpoonright C, & (R \downharpoonright C)^{-1} &= R^{-1} \downharpoonright C, & (R^{-1})^{-1} &= R. \end{aligned}$$

Next, the preprocessed input concept  $C$  is tagged with a fresh individual name  $\ell$  that does not occur in  $C$ . Then we build a complete tableau  $T_{\mathcal{ALBO}}(C)$  by applying the rules of  $T_{\mathcal{ALBO}}$  to the concept assertion  $\ell : C$  as described above. It is however important to note that  $\ell : C$  and all labelled expressions and assertions really denote concept expressions.

We turn to proving soundness and completeness of the calculus. Because every rule preserves the satisfiability of concept assertions, it is easy to see that the calculus  $T_{\mathcal{ALBO}}$  is sound for  $\mathcal{ALBO}$ .

For proving completeness, suppose that a tableau  $T_{\mathcal{ALBO}}(C)$  for the given concept  $C$  is open, i.e. it contains an open branch  $\mathcal{B}$ . We construct a model  $\mathcal{I}$  for the satisfiability of  $C$  as follows. By definition, let  $\ell \sim \ell' \stackrel{\text{def}}{\iff} \ell : \{\ell'\} \in \mathcal{B}$ . It is clear that the rules  $(\text{sym})$ ,  $(\text{mon})$ , and  $(\text{refl})$  ensure that  $\sim$  is an equivalence relation on individuals. The equivalence class  $\|\ell\|$  of a representative  $\ell$  is defined by:  $\|\ell\| \stackrel{\text{def}}{=} \{\ell' \mid \ell \sim \ell'\}$ . We set

$$\begin{aligned} \Delta^{\mathcal{I}} &\stackrel{\text{def}}{=} \{\|\ell\| \mid \ell : \{\ell\} \in \mathcal{B}\}, & r^{\mathcal{I}} &\stackrel{\text{def}}{=} \{(\|\ell\|, \|\ell'\|) \mid \ell : \exists r.\{\ell'\} \in \mathcal{B}\}, \\ p^{\mathcal{I}} &\stackrel{\text{def}}{=} \{\|\ell\| \mid \ell : p \in \mathcal{B}\}, & \ell^{\mathcal{I}} &\stackrel{\text{def}}{=} \begin{cases} \|\ell\|, & \text{if } \ell : \{\ell\} \in \mathcal{B}, \\ \|\ell'\| & \text{for some } \|\ell'\| \in \Delta^{\mathcal{I}}, \text{ otherwise.} \end{cases} \end{aligned}$$

- |  |  |
|--|--|
| 1. $\text{TBox} = \{\exists r.p\}$ .....given                  | 6. $\ell_0 \not\sim \ell_1$ .....                              |
| 2. $\ell_0 : \exists r.p$ ..... $\text{TBox} \boxed{1}$        | .....Loop checking: $p$ is a difference                        |
| 3. $\ell_1 : p$ ..... $\boxed{2} \boxed{2}$                    | 7. $\ell_2 : p$ ..... $\boxed{2} \boxed{5}$                    |
| 4. $\ell_0 : \exists r.\{\ell_1\}$ ..... $\boxed{2} \boxed{2}$ | 8. $\ell_1 : \exists r.\{\ell_2\}$ ..... $\boxed{2} \boxed{5}$ |
| 5. $\ell_1 : \exists r.p$ ..... $\text{TBox} \boxed{1}$        | 9. $\ell_2 : \exists r.p$ ..... $\text{TBox} \boxed{1}$        |
|  | 10. $\ell_1 \sim \ell_2$ .....Loop checking                    |

**Fig. 2.** Standard loop checking mechanism in  $\mathcal{ALCO}$

The rules  $\boxed{\text{sym}}$ ,  $\boxed{\text{mon}}$ ,  $\boxed{\text{refl}}$ , and  $\boxed{\text{bridge}}$  ensure that the definition of  $\mathcal{I}$  does not depend on representatives of the equivalence classes.

The following lemma is proved by induction over the structure  $C$ .

**Lemma 1.** (1) If  $\ell : D \in \mathcal{B}$  then  $\|\ell\| \in D^{\mathcal{I}}$  for any concept  $D$ .

(2) For every role  $R$  and every concept  $D$

(2a)  $\ell : \exists R.\{\ell'\} \in \mathcal{B}$  implies  $(\|\ell\|, \|\ell'\|) \in R^{\mathcal{I}}$ ,

(2b) if  $(\|\ell\|, \|\ell'\|) \in R^{\mathcal{I}}$  and  $\ell : \neg \exists R.D \in \mathcal{B}$  then  $\ell' : \neg D \in \mathcal{B}$ .

A consequence of this lemma is completeness of the tableau calculus.

**Theorem 2.**  $T_{\mathcal{ALCO}}$  is a sound and complete tableau calculus for satisfiability in  $\mathcal{ALBO}$ .

## 4 Blocking

There are satisfiable concepts which result in an infinite  $T_{\mathcal{ALCO}}$ -tableau, where all open branches are infinite. The concept  $\neg \exists (s \sqcup \neg s). \neg \exists r.p$  is such an example. Indeed, since the prefix  $\neg \exists (s \sqcup \neg s). \neg$  is equivalent to the universal modality, the concept  $\ell : \exists r.p$  is propagated to every individual  $\ell$  in every branch of the tableau. The concept  $\ell : \exists r.p$  itself, each time triggers the creation of a new individual with the  $\boxed{2}$  rule. Thus, any branch of the tableau contains infinitely many individuals. The branches have however a regular structure that can be detected with loop detection or blocking mechanisms.

Observe that satisfiability of the concept  $\neg \exists (s \sqcup \neg s). \neg \exists r.p$  corresponds to satisfiability of the  $\text{TBox} \{\exists r.p\}$  in the description logic  $\mathcal{ALCO}$ . Figure 2 demonstrates how *standard loop checking* (subset blocking) for the description logic  $\mathcal{ALCO}$  with general  $\text{TBoxes}$  detects a loop in this example. (In the figure each line in the derivation is numbered on the left. The rule applied and the number of the premise(s) to which it was applied to produce the labelled concept expression (assertion) in each line is specified on the right.) Loop checking tests are performed after all the type-completing rules have been applied to all concept expressions labelled with a specific individual relative to an ancestor individual. In the tableau in Figure 2, two loop checking tests are performed, namely in step 6 and step 10. Take step 10. All the type-completing rules have been applied to all concept expressions of the form  $\ell_2 : C$  and  $\ell_1 : C$ . This means the  $\exists r.p$ -types of the individuals  $\ell_1$  and  $\ell_2$ , of the partial model constructed so far, are complete. Comparison of the types shows that they coincide and, consequently, the individuals  $\ell_1$  and  $\ell_2$  can be identified. At step 6 the  $\exists r.p$ -types of

1. $\ell_0 : \exists r.p.$ .....given	13. $\ell_3 : \neg\exists t.\neg\exists s^{-1}.(p \sqcup \neg p)$ .. (□) 4,12
2. $\ell_0 : \exists s.p.$ .....given	14. $\blacktriangleright \ell_3 : \exists t.\{\ell_2\}$ ..... (□) 13,10
3. $\ell_0 : \exists t.\neg\exists t.\exists s^{-1}.(p \sqcup \neg p)$ ..... given	15. $\ell_2 : \neg\exists s^{-1}.(p \sqcup \neg p)$ ..... (□) 11,14
4. $\ell_0 : \neg\exists t.\exists t.\neg\exists s^{-1}.(p \sqcup \neg p)$ ... given	16. $\ell_0 : \neg(p \sqcup \neg p)$ ..... (□) 15,9
5. $\ell_1 : p$ ..... (□) 11	17. Unsatisfiable.....after a few steps
6. $\ell_0 : \exists r.\{\ell_1\}$ ..... (□) 11	18. $\blacktriangleright \ell_2 : \neg\neg\exists s^{-1}.(p \sqcup \neg p)$ .. (□) 13,10
7. $\ell_1 : \{\ell_1\}$ ..... (refl) 5	19. $\ell_2 : \exists s^{-1}.(p \sqcup \neg p)$ ..... (□) 18
8. $\ell_2 : p$ ..... (□) 2	20. $\blacktriangleright \ell_3 : \exists t.\{\ell_1\}$ ..... (□) 13,7
9. $\ell_0 : \exists s.\{\ell_2\}$ ..... (□) 2	21. $\ell_1 : \neg\exists s^{-1}.(p \sqcup \neg p)$ ... (□) 11,20
10. $\ell_2 : \{\ell_2\}$ ..... (refl) 8	...
11. $\ell_3 : \neg\exists t.\exists s^{-1}.(p \sqcup \neg p)$ ..... (□) 3	22. $\blacktriangleright \ell_1 : \neg\neg\exists s^{-1}.(p \sqcup \neg p)$ .. (□) 13,7
12. $\ell_0 : \exists t.\{\ell_3\}$ ..... (□) 3	...

**Fig. 3.** Global effect of the introduction of a new individual

$\ell_0$  and  $\ell_1$  are different because  $\ell_0 : p$  is not present in the branch. The derivation therefore cannot yet stop, but does in step 10.

This example illustrates the simplest form of standard loop checking used in description and modal logic tableau procedures. This form of loop checking is too simple to handle role negation though. The problem is that the introduction of a new individual in a tableau has, in general, a global effect in the provisional model constructed so far. This global effect is illustrated by the example in Figure 3. (The black triangles in the figure denote branching points in the derivation. A branch expansion after a branching point is indicated by appropriate indentation.) At step 10 none of the type-completing rules need to be applied to concepts labelled with  $\ell_1$  and  $\ell_2$ . Although at this point  $\ell_1$  and  $\ell_2$  are labels of the same subconcepts of the given concepts, we cannot make them equal. The reason is that at step 11 a new individual is introduced which causes a few applications of the (□) rule, and as a result, at step 21, the types of  $\ell_1$  and  $\ell_2$  are now distinguished by the concept  $\exists s^{-1}.(p \sqcup \neg p)$ .

The examples illustrate that the reason for non-termination of  $T_{\mathcal{AL}\mathcal{CO}}$  is the possible infinite generation of labels. The following lemma holds, where  $\#\exists(\mathcal{B})$  denotes the number of applications of the (□) rule in a branch  $\mathcal{B}$ .

**Lemma 2.** *If  $\#\exists(\mathcal{B})$  is finite then  $\mathcal{B}$  is finite.*

In order to turn the calculus  $T_{\mathcal{AL}\mathcal{CO}}$  into a terminating calculus for  $\mathcal{AL}\mathcal{CO}$ , we introduce a new, different approach to blocking.

Let  $<$  be an ordering on individuals in the branch which is a linear extension of the order of introduction of the individuals during the derivation. I.e. let  $\ell < \ell'$ , whenever the first appearance of individual  $\ell'$  in the branch is strictly later than the first appearance of individual  $\ell$ . We add the following rule, called the *unrestricted blocking* rule, to the calculus.

$$(\text{ub}): \frac{\ell : \{\ell\}, \ell' : \{\ell'\}}{\ell : \{\ell'\} \mid \ell : \neg\{\ell'\}}$$

Moreover, we require that the following conditions hold.

- (c1) Any rule is applied at most once to the same set of premises.
- (c2) The  $(\exists)$  rule is applied only to expressions of the form  $\ell : \exists R.C$ , when  $C$  is not a singleton, i.e.  $C \neq \{\ell''\}$  for some individual  $\ell''$ .
- (c3) If  $\ell : \{\ell'\}$  appears in a branch and  $\ell < \ell'$  then all further applications of the  $(\exists)$  rule to expressions of the form  $\ell' : \exists R.C$  are not performed within the branch.
- (c4) In every open branch there is some node from which point onwards before any application of the  $(\exists)$  rule all possible applications of the  $(\text{ub})$  rule have been performed.

We use the notation  $T_{\mathcal{ALBO}}(\text{ub})$  for the extension of  $T_{\mathcal{ALBO}}$  with this rule and this blocking mechanism.

The blocking requirements  $(\text{c1})$ – $(\text{c4})$  are sound in the sense that they cannot cause an open branch to become closed. The  $(\text{ub})$  rule is sound in the usual sense. Thus, the following theorem holds.

**Theorem 3.**  $T_{\mathcal{ALBO}}(\text{ub})$  is a sound and complete tableau calculus for  $\mathcal{ALBO}$ .

Let  $\mathcal{B}$  be the *leftmost open branch with respect to the rule  $(\text{ub})$*  in the  $T_{\mathcal{ALBO}}(\text{ub})$  tableau for a given concept  $C$ . Assume that  $\mathcal{I}$  is a model constructed from  $\mathcal{B}$  as in Section 3.

It can be shown that the tableau procedure mimics the construction of a finite model used in the proof of Theorem 1. Thus the following lemma holds.

**Lemma 3.**  $\Delta^{\mathcal{I}}$  is finite.

For every  $\|\ell\| \in \Delta^{\mathcal{I}}$ , let  $\#\exists(\|\ell\|)$  denote the number of applications of the  $(\exists)$  rule to concepts of the form  $\ell' : \exists R.D$  with  $\ell' \in \|\ell\|$ .

**Lemma 4.** (1)  $\#\exists(\|\ell\|)$  is finite for every  $\|\ell\| \in \Delta^{\mathcal{I}}$ .

(2)  $\#\exists(\mathcal{B})$  is finite.

**Corollary 1.** If the leftmost branch with respect to the rule  $(\text{ub})$  in a  $T_{\mathcal{ALBO}}(\text{ub})$  tableau is open then the branch is finite.

The termination theorem is an immediate consequence.

**Theorem 4 (Termination).**  $T_{\mathcal{ALBO}}(\text{ub})$  is a terminating tableau calculus for satisfiability in  $\mathcal{ALBO}$ .

Notice that condition  $(\text{c4})$  is essential for ensuring termination of a  $T_{\mathcal{ALBO}}(\text{ub})$  derivation. Indeed, it is easy to see that without  $(\text{c4})$  the  $T_{\mathcal{ALBO}}(\text{ub})$  tableau for the concept  $\neg(\exists(s \sqcup \neg s). \neg \exists r.p \sqcup \exists(s \sqcup \neg s). \neg \exists r. \neg p)$  would not terminate because new individuals are generated more often than their equality check is performed via the rule  $(\text{ub})$ .

## 5 Decision Procedures

When turning the presented calculus  $T_{\mathcal{ALBO}}(\text{ub})$  into a deterministic decision procedure it is crucial that this is done in a *fair* way. A procedure is fair if, when



1. $\ell_0 : C$ ..... given	21.	$\blacktriangleright \ell_0 : \exists s.\{\ell_2\}$ ..... (□⇒) 7,20
2. $\ell_0 : \{\ell_0\}$ ..... (refl) 11	22.	$\ell_2 : \neg\neg\exists r.p$ ..... (□⇒) 6,21
3. $\ell_0 : \neg\exists(s \sqcup \neg s).\neg\exists r.p$ ..... (□⇒) 11	23.	Unsatisfiable..... (□) 18,22
4. $\ell_0 : \neg\neg\exists t.\neg\exists r.p$ ..... (□⇒) 11	24.	$\blacktriangleright \ell_2 : \neg\neg\exists r.p$ ..... (□⇒) 7,20
5. $\ell_0 : \exists t.\neg\exists r.p$ ..... (□⇒) 4	25.	Unsatisfiable..... (□) 18,24
6. $\ell_0 : \neg\exists s.\neg\exists r.p$ ..... (□⇒□) 3	26.	$\blacktriangleright \ell_0 : \neg\{\ell_1\}$ ..... (nb)
7. $\ell_0 : \neg\neg\exists s.\neg\exists r.p$ ..... (□⇒□) 3	27.	$\ell_2 : p$ ..... (□) 16
8. $\blacktriangleright \ell_0 : \exists s.\{\ell_0\}$ ..... (□⇒) 7,2	28.	$\ell_1 : \exists r.\{\ell_2\}$ ..... (□) 16
9. $\ell_0 : \neg\neg\exists r.p$ ..... (□⇒) 8,6	29.	$\ell_2 : \{\ell_2\}$ ..... (refl) 27
10. $\ell_0 : \exists r.p$ ..... (□⇒) 9	30.	$\blacktriangleright \ell_0 : \exists s.\{\ell_2\}$ ..... (□⇒) 7,29
11. $\ell_1 : p$ ..... (□) 10	31.	$\ell_2 : \neg\neg\exists r.p$ ..... (□⇒) 30,6
12. $\ell_0 : \exists r.\{\ell_1\}$ ..... (□) 10	32.	$\ell_2 : \exists r.p$ ..... (□⇒) 31
13. $\ell_1 : \{\ell_1\}$ ..... (refl) 11	33.	<b>Non-terminating</b> .....
14. $\blacktriangleright \ell_0 : \exists s.\{\ell_1\}$ ..... (□⇒) 7,13		..... Repetition of 16-32
15. $\ell_1 : \neg\neg\exists r.p$ ..... (□⇒) 14,6	34.	$\blacktriangleright \ell_2 : \neg\neg\exists r.p$ ..... (□⇒) 7,29
16. $\ell_1 : \exists r.p$ ..... (□⇒) 15	35.	..... Similarly to 30-33
17. $\blacktriangleright \ell_0 : \{\ell_1\}$ ..... (nb)	36.	$\blacktriangleright \ell_1 : \neg\neg\exists r.p$ ..... (□⇒) 7,13
18. $\ell_2 : \neg\exists r.p$ ..... (□) 5	37.	..... Similarly to 14-35
19. $\ell_0 : \exists t.\{\ell_2\}$ ..... (□) 5	38.	$\blacktriangleright \ell_0 : \neg\neg\exists r.p$ ..... (□⇒) 7,2
20. $\ell_2 : \{\ell_2\}$ ..... (refl) 18	39.	..... Similarly to 8-37

Fig. 4. An infinite derivation, due to unfair selection of concepts

an inference is possible forever, then it is performed eventually. This means a deterministic tableau procedure based on  $T_{ALBO}$  (nb) may not defer the use of an applicable rule indefinitely. Note that we understand fairness in a ‘global’ sense. That is, a tableau procedure has to be fair not only to expressions in a particular branch but to expressions in *all* branches of a tableau. In other words, a procedure is fair if it makes both the branch selection, and the selection of expressions to which to apply a rule to, in a fair way.

**Theorem 5.** *Any fair tableau procedure based on  $T_{ALBO}$  (nb) is a decision procedure for  $ALBO$  and all its sublogics.*

Note that we do not assume that the branches are expanded in a depth-first left-to-right order. Nevertheless, it also follows from our results that:

**Theorem 6.** *Any fair tableau procedure based on  $T_{ALBO}$  (nb) which uses a depth-first and left-to-right strategy, with respect to branch selection of the (nb) rule, is a decision procedure for  $ALBO$  and all its sublogics.*

To illustrate the importance of fairness we give two examples. The concept

$$C \stackrel{\text{def}}{=} \neg(\exists(s \sqcup \neg s).\neg\exists r.p \sqcup \neg\exists t.\neg\exists r.p)$$

is not satisfiable. Figure 4 gives a depth-first left-to-right derivation which is unfair and does not terminate. It can be seen that the derivation is infinite because the application of the (□) rule to  $\ell_0 : \exists t.\neg\exists r.p$  is deferred forever and, consequently, a contradiction is not found. This illustrates the importance of fairness for (refutational) completeness.

1. $\ell_0 : \neg\exists(s \sqcup \neg s). \neg\exists r.p$ ..... given	12. $\blacktriangleright \ell_0 : \neg\{\ell_1\}$ ..... (ub) 2,9
2. $\ell_0 : \{\ell_0\}$ ..... (refl) 1	13. $\ell_2 : p$ ..... (E) 11
3. $\ell_0 : \neg\exists s. \neg\exists r.p$ ..... (E) 1	14. $\ell_1 : \exists r.\{\ell_2\}$ ..... (E) 11
4. $\ell_0 : \neg\exists\neg s. \neg\exists r.p$ ..... (E) 1	15. <b>Non-terminating</b> .....
5. $\blacktriangleright \ell_0 : \neg\neg\exists r.p$ ..... (E) 4,2	..... Repetition of 7-14
6. $\ell_0 : \exists r.p$ ..... (E) 5	16. $\blacktriangleright \ell_0 : \{\ell_1\}$ ..... (ub) 2,9
7. $\ell_1 : p$ ..... (E) 6	17. .... Never expanded
8. $\ell_0 : \exists r.\{\ell_1\}$ ..... (E) 6	18. $\blacktriangleright \ell_0 : \exists s.\{\ell_1\}$ ..... (E) 4,9
9. $\ell_1 : \{\ell_1\}$ ..... (refl) 7	19. .... Never expanded
10. $\blacktriangleright \ell_1 : \neg\neg\exists r.p$ ..... (E) 4,9	20. $\blacktriangleright \ell_0 : \exists s.\{\ell_0\}$ ..... (E) 4,2
11. $\ell_1 : \exists r.p$ ..... (E) 5	21. .... Never expanded

Fig. 5. An infinite derivation, due to unfair selection of branches

The next example illustrates the importance of fairness for decidability. The concept  $\neg\exists(s \sqcup \neg s). \neg\exists r.p$  is satisfiable. The derivation in Figure 5 is obtained with a depth-first *right-to-left* strategy. However, the repeated selection of the right branch at (ub) choice points causes all the individuals in the branch to be pairwise non-equal. The concept  $\ell : \exists r.p$  re-appears repeatedly, for every individual  $\ell$  in the branch. This triggers the repeated generation of a new individual by the (E) rule, resulting in an infinite derivation. This strategy is unfair because all branches except for the rightmost branch get ignored.

In an implemented prover, optimisations, good heuristics and clever backtracking techniques are important. The standard optimisations such as simplification, backjumping, dynamic backtracking, different heuristics for branch selection and rule selection, lemma generation, et cetera, are all compatible with the presented calculi and procedures. An obvious simplification, for example, is the on-the-fly removal of double negations from concepts, and especially from roles, as this reduces a number of applications of the (E) rule.

Since the presented tableaux operate only on ground labelled *concept* expressions, they can in principle be implemented as extensions of existing tableau-based description logic systems which can handle singleton concepts. We have implemented the unrestricted blocking rule as a plug-in to the METTEL tableau prover [8], and tests with various description logics are encouraging.

## 6 Conclusion

We have presented a new, general tableau approach for deciding expressive description logics with complex role operators, including ‘non-safe’ occurrences of role negation. The tableau decision procedures found in the description logic literature, and implemented in existing tableau-based description logic systems, can handle a large class of description logics, but cannot currently handle description logics with full role negation such as *ALCB* or *ALCBO*. The present paper closes this gap. An important novelty of our approach is the use of a blocking mechanism based on small inference steps rather than ‘big’ tests performed on sets of expressions or assertions which are often tailored toward specific logics.

Our techniques are versatile and are not limited to  $\mathcal{ALBO}$  or its sublogics, but carry over to all description logics and also other logics including first-order logic.

*Acknowledgements.* We thank the referees for valuable comments. The work is supported by EPSRC research grant EP/D056152/1.

## References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: Description Logic Handbook. Cambridge Univ. Press, Cambridge (2003)
2. Baumgartner, P., Schmidt, R.A.: Blocking and other enhancements for bottom-up model generation methods. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 125–139. Springer, Heidelberg (2006)
3. De Nivelle, H., Pratt-Hartmann, I.: A resolution-based decision procedure for the two-variable fragment with equality. In: Goré, R.P., Leitsch, A., Nipkow, T. (eds.) IJCAR 2001. LNCS (LNAI), vol. 2083, pp. 211–225. Springer, Heidelberg (2001)
4. De Nivelle, H., Schmidt, R.A., Hustadt, U.: Resolution-based methods for modal logics. Log. J. IGPL 8(3), 265–292 (2000)
5. Grädel, E., Kolaitis, P.G., Vardi, M.Y.: On the decision problem for two-variable first-order logic. Bull. Sect. Log. 3, 53–69 (1997)
6. Hustadt, U., Schmidt, R.A.: Issues of decidability for description logics in the framework of resolution. In: Caferra, R., Salzer, G. (eds.) Automated Deduction in Classical and Non-Classical Logics. LNCS (LNAI), vol. 1761, pp. 191–205. Springer, Heidelberg (2000)
7. Hustadt, U., Schmidt, R.A., Georgieva, L.: A survey of decidable first-order fragments and description logics. J. Relat. Methods Comput. Sci. 1, 251–276 (2004)
8. Hustadt, U., Tishkovsky, D., Wolter, F., Zakharyashev, M.: Automated reasoning about metric and topology (System description). In: Fisher, M., van der Hoek, W., Konev, B., Lisitsa, A. (eds.) JELIA 2006. LNCS (LNAI), vol. 4160, pp. 490–493. Springer, Heidelberg (2006)
9. Lutz, C., Sattler, U.: The complexity of reasoning with Boolean modal logics. In: Advances in Modal Logics, vol. 3, CSLI Publ., Stanford (2002)
10. Massacci, F.: Decision procedures for expressive description logics with intersection, composition, converse of roles and role identity. In: Proc. IJCAI 2001, pp. 193–198. Morgan Kaufmann, San Francisco (2001)
11. Schmidt, R.A.: Developing modal tableaux and resolution methods via first-order resolution. In: Advances in Modal Logic, vol. 6, pp. 1–26. College Publ. (2006)
12. Schmidt, R.A., Hustadt, U.: Mechanised reasoning and model generation for extended modal logics. In: de Swart, H., Orłowska, E., Schmidt, G., Roubens, M. (eds.) Theory and Applications of Relational Structures as Knowledge Instruments. LNCS, vol. 2929, pp. 38–67. Springer, Heidelberg (2003)
13. Schmidt, R.A., Orłowska, E., Hustadt, U.: Two proof systems for Peirce algebras. In: Berghammer, R., Möller, B., Struth, G. (eds.) Relational and Kleene-Algebraic Methods in Computer Science. LNCS, vol. 3051, pp. 238–251. Springer, Heidelberg (2004)
14. Tobies, S.: Decidability Results and Practical Algorithms for Logics in Knowledge Representation. PhD dissertation, RWTH Aachen (2001)

# Instance Migration in Heterogeneous Ontology Environments

Luciano Serafini and Andrei Tamilin

Foundation B. Kessler - IRST (formerly ITC-IRST)  
Via Sommarive 18, 38050 Trento, Italy  
{serafini,tamilin}@itc.it

**Abstract.** In this paper we address the problem of migrating instances between heterogeneous overlapping ontologies. The instance migration problem arises when one wants to reclassify a set of instances of a source ontology into a semantically related target ontology. Our approach exploits mappings between ontologies, which are used to reconcile both conceptual and individual level heterogeneity, and further used to draw the migration process. We ground the approach on a distributed description logic (DDL), in which ontologies are formally encoded as DL knowledge bases and mappings as bridge rules and individual correspondences. From the theoretical side, we study the task of reasoning with instance data in DDL composed of *SHIQ* ontologies and define a correct and complete distributed tableaux inference procedure. From the practical side, we upgrade the DRAGO DDL reasoner for dealing with instances and further show how it can be used to drive the migration of instances between heterogeneous ontologies.

## 1 Introduction

The semantic web steadily evolves growing into a container of multiple distributed ontologies. Although ontologies are supposed to provide a consensual model of the world, those found in practice are far from being so. They rather formalize a subjective view in accordance with diverse assumptions, such as target goals, available background knowledge, biases, etc. This fact inevitably leads to a situation in which the same domain is represented by more than one ontology in *heterogeneous* ways.

The classical approach to the problem of *reconciling heterogeneity* between ontologies consists of two essential steps: matching and reasoning. By matching one creates a set of *mappings*—comprising semantic correspondences—between elements of different ontologies. For example, a mapping can express the fact that the concept **Scholar** in one ontology is equivalent to the concept **Student** in another ontology, or that the individual **Samuel Clemens** in one ontology corresponds to the individual **Mark Twain** in the other ontology. Mappings can be created manually, by domain experts, or discovered (semi-)automatically by existing matching techniques [8]. Once mappings are stated, it is necessary to provide a method for *reasoning* with them. Formally, this amounts to evaluating logical consequences of mappings on the mapped ontologies.

Mappings from a source to a target ontology can be used to transfer knowledge between the two ontologies. When ontologies are represented in OWL (formally corresponding to DL knowledge bases) there are two types of knowledge that can be transferred: terminological knowledge (i.e., mappings can force new concept subsumptions

in the target ontology) and assertional knowledge (i.e., mappings can assert new instances of a certain concept). To motivate and explain the problem, let us consider a scenario in which ontology mappings are used for migrating instances.

*Example 1 (Motivating scenario).* A computer science department of some university employed an ontology  $O^{\text{CS Dept}}$  for accessing data related to organization of the department. Successively, a human resource office of the university decides to make use of ontologies to represent data about the employees of the whole university. For this purpose they create an ontology  $O^{\text{HRO Uni}}$  and further populate it with instance data. To save time, they decide to reuse the work done by the computer science department. However, to do that they need to solve the following two problems. First, the conceptualization in  $O^{\text{HRO Uni}}$  is different from the one in  $O^{\text{CS Dept}}$ , so the classification of instances done in  $O^{\text{CS Dept}}$  must be adapted to the classification schema implemented in  $O^{\text{HRO Uni}}$ . Second, the identifiers used in  $O^{\text{CS Dept}}$  for people do not coincide with the identifiers used in  $O^{\text{HRO Uni}}$ , thus they have to provide a way to transform (regenerate) identifiers of the people they want to insert in their ontology.

The main objective of the paper is to provide a logical characterization of the assertional information enforced by a set of mappings. Based on this characterization, we propose a sound and complete distributed tableaux algorithm that reclassifies instances of a source ontology into a target ontology in accordance to the mappings. The feasibility of this approach is shown by describing the implementation of such an algorithm in the DRAGO DDL Reasoner<sup>1</sup>. The approach described in the paper, relies on the logical framework of distributed description logics (DDL) [4]. In such a framework a *distributed knowledge base* consists of a family of standard DL knowledge bases, corresponding to each given ontology, a set of *bridge rules*, corresponding to mapping between pairs of terminologies (T-boxes), and *individual correspondences*, corresponding to rules for transforming individuals across instance storages (A-boxes). The concrete contribution of the present work includes:

- characterization of the role that bridge rules and individual correspondences play when reasoning with instances in DDL; in particular, we show that they are capable of propagating concept membership assertions across ontologies
- definition of a sound and complete distributed extension to *SHIQ*-A-box tableau for reasoning with instances in DDL; the algorithm implements the backward chaining strategy for computation of propagated assertions by distributed communication with DL tableaux reasoners attached to other ontologies
- extending the DRAGO DDL Reasoner, currently limited to reasoning without instances, with the algorithm proposed in this paper and further showing its application to drive instance migrations.

The paper is organized as follows. In Section 2 we introduce the definition of a DDL distributed knowledge base with bridge rules and individual mappings. In Section 3 we study how bridge rules and individual mappings propagate information across knowledge bases. The implementation in DRAGO and the application to the instance migration scenario is further outlined in Section 4. We end up with an overview of related work and concluding remarks.

<sup>1</sup> <http://sra.itc.it/projects/drago>

## 2 Distributed Knowledge Bases

As introduced by Borgida and Serafini in [4], DDL is a formalism for representing multiple ontologies *pairwise* interconnected by directional semantic mappings. In this section we recall and extend the basic definitions of DDL.

### 2.1 Syntax and Semantics

A distributed knowledge base formalizes a set of ontologies interconnected by semantic mappings. The first component of a distributed knowledge base is a family of knowledge bases  $\mathcal{K} = \{\mathcal{K}_i\}_{i \in I}$ . According to a standard DL definitions, each  $\mathcal{K}_i$  consists of a terminological component  $\mathcal{T}_i$  (T-box) and an assertional component  $\mathcal{A}_i$  (A-box). Since the very same symbol can be used in two knowledge bases with different meaning, to unambiguously refer to elements of  $\mathcal{K}_i$ , they are prefixed with the index  $i$  of the knowledge base. The notations  $i : a$ ,  $i : C$ ,  $i : C \sqsubseteq D$ ,  $i : C(a)$  and  $i : R(a, b)$ , stand for an individual  $a$ , concept  $C$ , subsumption  $C \sqsubseteq D$ , assertions  $C(a)$  and  $R(a, b)$ , respectively in the knowledge base  $\mathcal{K}_i$ .

Mappings from  $\mathcal{K}_i$  to  $\mathcal{K}_j$  knowledge bases with ( $i \neq j$ ) are encoded as bridge rules and individual correspondences, which are expressions of the following forms:

- $i : C \xrightarrow{\sqsubseteq} j : D$  (into-bridge rule)
- $i : C \xrightarrow{\sqsupseteq} j : D$  (onto-bridge rule)
- $i : a \mapsto j : b$  (individual correspondence)

where  $C$  and  $D$  are concept names of  $\mathcal{T}_i$  and  $\mathcal{T}_j$ , and  $a$  and  $b$  are individuals of  $\mathcal{A}_i$  and  $\mathcal{A}_j$  respectively.<sup>2</sup>

Both bridge rules and individual correspondences from  $\mathcal{K}_i$  to  $\mathcal{K}_j$  express relations between  $\mathcal{K}_i$  and  $\mathcal{K}_j$  viewed from the  $j$ -th *subjective* point of view. Intuitively, the into-bridge rule  $i : \text{PhDThesis} \xrightarrow{\sqsubseteq} j : \text{Thesis}$  states that, from the  $j$ -th point of view the concept PhDThesis in  $i$  is less general than its local concept Thesis. Similarly, the onto-bridge rule  $i : \text{InProceedings} \xrightarrow{\sqsupseteq} j : \text{ConferencePaper}$  expresses the more generality relation. The individual correspondence  $i : \text{mario\_phd\_thesis} \mapsto j : \text{mario\_thesis}$  expresses the fact the  $\mathcal{A}_j$ 's individual mario\_thesis is one of the possible translations in the language of  $\mathcal{K}_j$  of the  $\mathcal{A}_i$ 's individual mario\_phd\_thesis. In the general case we admit that an individual can have more than one translation.

A *distributed T-box*  $\mathcal{T}$  consists of T-boxes  $\mathcal{T}_i$  and a collection  $\mathfrak{B}$  of bridge rules between them. A *distributed A-box*  $\mathcal{A}$  consists of A-boxes  $\mathcal{A}_i$  and a collection of individual correspondences  $\mathfrak{C}$ . A *distributed knowledge base*  $\mathfrak{K}$  is then a tuple  $\langle \mathcal{T}, \mathcal{A} \rangle$ .

The semantics of DDL is defined with a fundamental assumption that each knowledge base  $\mathcal{K}_i$  in the family is *locally interpreted* on its *local interpretation domain*. To support directionality, (i.e., mappings from  $i$  to  $j$  only propagate in the  $i$ -to- $j$ -direction), we admit the hole interpretation  $\mathcal{I}_\epsilon$  with empty domain (see more details

<sup>2</sup> In this work we concentrate only on individual correspondences, and don't consider complete correspondences which does not have any additional effect in data migration.

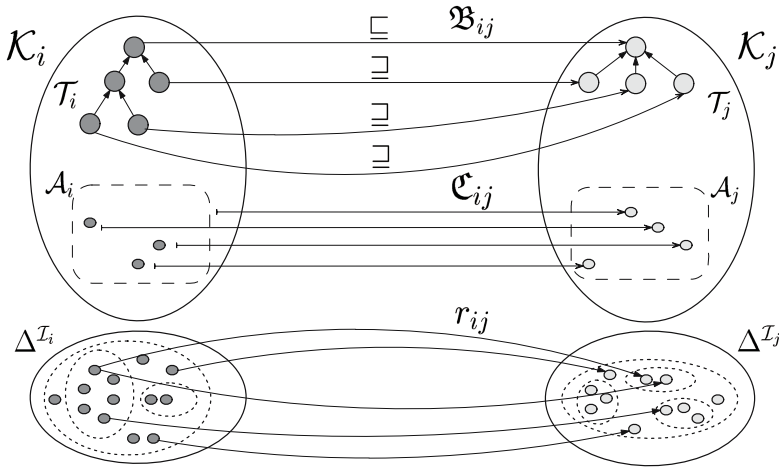


Fig. 1. Visualized semantics of DDL framework

in [15].<sup>3</sup> By definition, we impose that  $\mathcal{I}_\epsilon$  satisfies any knowledge base. To resolve heterogeneity between different domains the DDL defines a binary *domain relation*  $r$  between pairs of these domains. Figure 1 intuitively depicts component elements of DDL semantics.

A *distributed interpretation*  $\mathcal{I}$  of a distributed knowledge base  $\mathfrak{K} = \langle \mathfrak{T}, \mathfrak{A} \rangle$  consists of a family of local interpretations  $\mathcal{I}_i$  on local interpretation domains  $\Delta^{\mathcal{I}_i}$  and a family of domain relations  $r_{ij} \subseteq \Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_j}$  between pairs of local domains.

A distributed interpretation  $\mathcal{I}$  *satisfies* a distributed knowledge base  $\mathfrak{K} = \langle \mathfrak{T}, \mathfrak{A} \rangle$ , is called a *model* of  $\mathfrak{K}$ , if all its' components are satisfied according to the following rules:

- $\mathcal{I}_i$  satisfies  $\mathcal{K}_i$
- $r_{ij}(C^{\mathcal{I}_i}) \supseteq D^{\mathcal{I}_j}$  for all  $i : C \xrightarrow{\exists} j : D$
- $r_{ij}(C^{\mathcal{I}_i}) \subseteq D^{\mathcal{I}_j}$  for all  $i : C \xrightarrow{\sqsubseteq} j : D$
- $b^{\mathcal{I}_j} \subseteq r_{ij}(a^{\mathcal{I}_i})$  for all  $i : a \mapsto j : b$

## 2.2 Inference Services

Although both in DL and Distributed DL the fundamental reasoning services lay in verification of concept satisfiability/subsumption and instance checking/retrieval within a certain ontology, in DDL, besides the ontology itself, the other ontologies and mappings between them should be taken into account. Given a distributed knowledge base  $\mathfrak{K} = \langle \mathfrak{T}, \mathfrak{A} \rangle$ , the *distributed inference services* can be defined as follows:

<sup>3</sup> Classically, DL interpretation maps every individual into an *element* of the domain, while the hole maps everything into the empty *set*. To allow homogeneous treatment of standard DL interpretations and holes, we require that any individual  $x$  is standardly interpreted into a singleton set, rather than into an element of the domain. Hence,  $\mathcal{I}_i \models C(a) \iff a^{\mathcal{I}_i} \subseteq C^{\mathcal{I}_i}$ , rather than  $a^{\mathcal{I}_i} \in C^{\mathcal{I}_i}$ . We thank the anonymous reviewer for pointing at this mismatch.



**Satisfiability.** A concept  $C$  is *satisfiable* in  $i$  with respect to  $\mathfrak{K}$  if there exists a distributed interpretation  $\mathcal{I}$  of  $\mathfrak{K}$  such that  $C^{\mathcal{I}_i} \neq \emptyset$ .

**Subsumption.** A concept  $C$  is *subsumed* by a concept  $D$  in  $i$  with respect to  $\mathfrak{K}$  if for every distributed interpretation  $\mathcal{I}$  of  $\mathfrak{K}$  we have that  $C^{\mathcal{I}_i} \subseteq D^{\mathcal{I}_i}$ . In this case we will write  $\mathfrak{K} \models i : C \sqsubseteq D$ .

**Instantiation.** An individual  $a$  is an *instance* of a concept  $C$  in  $i$  with respect to  $\mathfrak{K}$  if for every distributed interpretation  $\mathcal{I}$  of  $\mathfrak{K}$  we have that  $a^{\mathcal{I}_i} \subseteq C^{\mathcal{I}_i}$ . In this case we will write  $\mathfrak{K} \models i : C(a)$ .

**Retrieval.** Computing the individuals in  $\mathcal{K}_i$  that instantiate a given concept  $C$  in  $i$  with respect to  $\mathfrak{K}$ .

The group of concept satisfiability/subsumption services is typically referred to as terminological reasoning services, while the remaining instantiation/retrieval services are grouped into assertional reasoning services.

The question of providing terminological services for DDL has been already studied in [15]. It has been shown that certain combinations of into- and onto-bridge rules can lead to the propagation of knowledge in form of subsumption axioms across ontologies participating in DDL. Moreover, in case of DDL with *SHIQ* components without instances adding these additional propagation rules to existing DL tableaux algorithms leads to a correct and complete reasoning in DDL. The presented method has been also implemented in the DRAGO DDL reasoner.

In the consequent sections, we investigate the assertional reasoning services.

### 3 Reasoning with Instances in Distributed Knowledge Bases

For the sake of clarity, we start considering the case of DDL with two component knowledge bases and unidirectional sets of bridge rules and individual correspondences. For the general results and proofs we refer the interested reader to the technical report [16].

#### 3.1 Inference Patterns

In the following we characterize the knowledge propagated from a knowledge base  $i$  (the source) to  $j$  (the target) by a set of *propagation rules* of the form:

$$\frac{(1) \text{ facts in } i, \quad (2) \text{ bridge rules from } i \text{ to } j, \quad (3) \text{ individual mappings from } i \text{ to } j}{(4) \text{ fact in } j}$$

which must be read as: if the facts in (1) are true in  $\mathcal{K}_i$ , the bridge rules in (2) are contained in  $\mathfrak{B}_{ij}$ , the individual correspondences in (3) are contained in  $\mathfrak{C}_{ij}$ , then the fact in (4) must be true in  $\mathcal{K}_j$ .

Following the semantics of mappings in DDL outlined in the previous section, it can be observed that the individual correspondences can interact with into-bridge rules with the effect of propagating concept membership assertions:

$$\frac{i : C(a), \quad i : C \xrightarrow{\Xi} j : D, \quad i : a \mapsto j : b}{j : D(b)} \quad (1)$$

Indeed,  $b^{\mathcal{I}_j} \subseteq r_{ij}(a^{\mathcal{I}_i}) \subseteq r_{ij}(C^{\mathcal{I}_i}) \subseteq D^{\mathcal{I}_j}$ .



In practice, this means that if an ontology  $O_1$  defines an instance `mario` of a concept `PhDStudent`, and an ontology  $O_2$  has some individual name `person_123`, then a bridge rule  $1 : \text{PhDStudent} \xrightarrow{\text{E}} 2 : \text{Student}$  and an individual correspondence  $1 : \text{mario} \mapsto 2 : \text{person\_123}$  entail the  $O_2$ 's assertion that `person_123` is an instance of `Student`.

In languages that support disjunction, the above propagation can be generalized to the propagation of instance membership over a disjunction of  $n \geq 0$  concepts:

$$\frac{i : (C_1 \sqcup \dots \sqcup C_n)(a), \quad i : C_k \xrightarrow{\text{E}} j : D_k \ (1 \leq k \leq n), \quad i : a \mapsto j : b}{j : (D_1 \sqcup \dots \sqcup D_n)(b)} \quad (2)$$

Several observations on the stated propagation pattern require a specific attention.

**Generality.** Rule (2) appears to be the *most general* form of assertion propagation in DDL when individual correspondences are restricted to be *functional*. A set of individual correspondences  $\mathfrak{C}_{ij}$  is functional if for every individual  $a$  of  $\mathcal{A}_i$  the set  $\mathfrak{C}_{ij}$  contains at most one individual correspondence  $i : a \mapsto j : b$ . For the sake of presentation, in this paper we restrict ourself to functional individual correspondences, leaving the most general case to the technical report [16].<sup>4</sup>

**Inconsistency propagation.** When  $n = 0$ , the inference pattern in (2) becomes the following inference rule:

$$\frac{i : \perp(a), \quad i : a \mapsto j : b}{j : \perp(b)} \quad (3)$$

which states that to propagate the inconsistency of  $\mathcal{K}_i$  to  $\mathcal{K}_j$  it's enough to have one single individual correspondence. From the representational point of view this inference rule is very fragile. We currently do not see an easy solution to fix this sensitivity to inconsistency propagation. This topic will be subject for further studies.

**Instance migration.** Up to now, we have supposed that individual correspondences are explicitly enumerated in  $\mathfrak{C}_{ij}$ . However in real situation, with thousands of individuals, one cannot expect to pre-compile all the individual mappings. However, the formalism support a more compact approach of declaring individual correspondences via a translation function  $f_{ij}$ , defined on the domain of the source ontology  $i$  and producing individuals in the domain of target ontology  $j$ . For example,  $f_{ij}$  can be the identity function, hence its application yields all instances from  $\mathcal{K}_i$  to be copied to  $\mathcal{K}_j$ . Such an approach is practically applicable to the instance migration scenario described in the introduction.

Once a translation function  $f_{ij}$  is defined, we can revisit propagation pattern (2):

$$\frac{i : (C_1 \sqcup \dots \sqcup C_n)(a), \quad i : C_k \xrightarrow{\text{E}} j : D_k \ (1 \leq k \leq n)}{j : (D_1 \sqcup \dots \sqcup D_n)(f_{ij}(a))} \quad (4)$$

<sup>4</sup> To give an intuition of the effect of non functional individual mappings, consider the case in which there are two into-bridge rules  $i : C_1 \xrightarrow{\text{E}} j : D_1$  and  $i : C_2 \xrightarrow{\text{E}} j : D_2$  and, the non functional set of individual mappings  $\{i : a \mapsto j : b, \ i : a \mapsto j : c\}$ . Then the fact that  $\mathcal{K}_i \models C_1 \sqcup C_2(a)$  entails the disjunctive assertion  $(D_1(b) \wedge D_1(c)) \vee (D_2(b) \wedge D_2(c))$ . This implies that, for the general case we have to introduce the technicalities for disjunctive A-boxes.

This later means that a fresh individual  $f_{ij}(a)$  is injected into  $\mathcal{K}_j$  and asserted as an instance of the disjunction of the  $D_k$ 's.

### 3.2 Soundness and Completeness

To demonstrate the correctness and completeness of the inference pattern presented in Section 3.1, we follow the approach similar to the one taken in [15]. The main idea consists in construction of an operator which essentially applies the generalized inference pattern (2) to extend knowledge bases with new assertions induced by mappings.

Given a set of bridge rules  $\mathfrak{B}_{12}$  and set of individual correspondences  $\mathfrak{C}_{12}$  from  $\mathcal{K}_1$  to  $\mathcal{K}_2$ , the *individual correspondence operator*  $\mathfrak{C}_{12}(\cdot)$ , taking as input a knowledge base  $\mathcal{K}_1$  and producing an A-box of  $\mathcal{K}_2$ , is defined as follows:

$$\mathfrak{C}_{12}(\mathcal{K}_1) = \left\{ (D_1 \sqcup \dots \sqcup D_n)(b) \left| \begin{array}{l} \mathcal{K}_1 \models (C_1 \sqcup \dots \sqcup C_n)(a) \\ 1 : C_k \xrightarrow{\underline{E}} 2 : D_k \in \mathfrak{B}_{12} \ (1 \leq k \leq n) \\ 1 : a \mapsto 2 : b \in \mathfrak{C}_{12} \end{array} \right. \right\}$$

It is remarkable that *onto*-bridge rules do not affect instance propagation. The reason is that onto-bridge rules impose only existence of preimages of objects that already exists in the target ontology. Into-bridge rules, instead, constraint the individual mappings to be defined within a certain range. The individual correspondence operator formalizes the assertional knowledge that is propagated across ontologies.

The characterization of the propagation of the terminological knowledge is characterized by an analogous operator, called *bridge operator*, introduced in [15] and defined as follows:  $\mathfrak{B}_{12}(\cdot)$ , taking as input a knowledge base  $\mathcal{K}_1$  and producing a T-box of  $\mathcal{K}_2$ :

$$\mathfrak{B}_{12}(\mathcal{K}_1) = \left\{ B \sqsubseteq D_1 \sqcup \dots \sqcup D_n \left| \begin{array}{l} \mathcal{T}_1 \models A \sqsubseteq C_1 \sqcup \dots \sqcup C_n \\ 1 : C_k \xrightarrow{\underline{E}} 2 : D_k \in \mathfrak{B}_{12} \ (1 \leq k \leq n) \\ 1 : A \xrightarrow{\underline{\exists}} 2 : B \in \mathfrak{B}_{12} \end{array} \right. \right\}$$

With the remarkable exception of inconsistency propagation—by rule (3)—the individual correspondences do not affect the propagation of terminological knowledge. The inferences formalized by the two operators described above *completely* describe the possible propagations that are forced by a set of bridge rules and individual correspondences. This is formally stated in the following theorem.

**Theorem 1 (Soundness and completeness).** *Let  $\mathfrak{R}_{12}$  be a distributed knowledge base consisting of  $\mathcal{K}_1, \mathcal{K}_2$  SHIQ knowledge bases, and  $\mathfrak{B}_{12}, \mathfrak{C}_{12}$  mappings between them. For any statement  $\phi$  (of the form  $C \sqsubseteq D$  or  $C(a)$ ) in the language of  $\mathcal{K}_2$*

$$\mathfrak{R}_{12} \models 2 : \phi \iff \langle \mathcal{T}_2 \cup \mathfrak{B}_{12}(\mathcal{K}_1), \mathcal{A}_2 \cup \mathfrak{C}_{12}(\mathcal{K}_1) \rangle \models \phi$$

The proof of the generalization of the Theorem 1 is fully described in the technical report. Some remarks are necessary.

**Independence between terminological and assertional propagation.** From the characterization above, one can see that propagation of terminological and assertional

knowledge are orthogonal. The two effects can be computed in parallel and independently. What is more important, however, is that the change of the A-box does not affect the propagation of the terminological knowledge. This means that if the source T-box does not change the terminological propagation is computed once for all.

**Local propagation of assertional knowledge.** Assertional propagation operator ensures, if a change of the source A-box involves only the set of individuals  $\{a_1, \dots, a_n\}$ , then assertional propagation must be computed only for the portion of the target A-box  $\mathcal{A}_2$  concerning the set of individuals  $\{b \mid 1 : a_i \mapsto 2 : b \in \mathcal{C}_{12}\}$ .

**Upper bound and complexity.** If the mapping from 1 to 2 is finite and contains  $m$  into-bridge rules,  $n$  onto-bridge rules, and  $o$  individual correspondences, then the bridge operator  $\mathfrak{B}_{12}$  applied to any knowledge base generates at most  $n * 2^m$  subsumption statements, and the individual correspondence operator  $\mathcal{C}_{12}$  generates at most  $o * 2^m$  instantiation statements. In total, the maximal number of statements that can propagate from  $\mathcal{K}_1$  to  $\mathcal{K}_2$  via mappings is  $(n + o) * 2^m$ . Since the propagation of statements needs checking subsumption and instantiation in the source knowledge base, which is EXPTIME complete, we have that computing subsumption and instantiation in a distributed setting is EXPTIME complete in the dimension of the source knowledge base plus mappings.

**Vanilla implementation.** The above theorem supports a vanilla implementation of *forward chaining* inference engine for DDL. The implementation consists of three steps: computation of propagation operators  $\mathfrak{B}_{12}(\mathcal{K}_1)$  and  $\mathcal{C}_{12}(\mathcal{K}_1)$ , construction of extended version of knowledge base  $\mathcal{K}_2$  as  $\langle \mathcal{T}_2 \cup \mathfrak{B}_{12}(\mathcal{K}_1), \mathcal{A}_2 \cup \mathcal{C}_{12}(\mathcal{K}_1) \rangle$ , and finally applying to this knowledge base one of existing DL reasoners, such as FaCT++ [18], Racer [10], or Pellet [17].

This approach to reasoning has a strong advantage of reuse of existing highly optimized DL reasoners, however it can be very costly for situations when semantic mappings are changed dynamically or when the required number of reasoning questions to be verified is relatively small. In the next section, we propose an alternative, *backward chaining* reasoning approach, which does “lazy”, or on demand, computation of propagated axioms and hence better fits to instable and short-living distributed environments.

### 3.3 Distributed Tableaux Algorithm

In this section we present a distributed tableaux algorithm for reasoning with instances in DDL. The main design idea consists in constructing a network of standard DL tableaux, one for each ontology in DDL, which communicate via mappings in a backward fashion.

Since we restricted the expressivity of ontologies participating in DDL to *SHIQ* DL, we will consider in the following that ontologies  $\mathcal{K}_1$  and  $\mathcal{K}_2$  from a distributed knowledge base  $\mathfrak{K}_{12} = \langle \mathcal{T}_{12}, \mathfrak{A}_{12} \rangle$  are attached with *SHIQ*-tableau reasoning procedures **Tab<sub>1</sub>** and **Tab<sub>2</sub>** [13]. Due to the reduction of reasoning with concepts to reasoning with instances [2], we suppose that each procedure **Tab<sub>i</sub>**( $\alpha$ ) can check the satisfiability of any statement  $\alpha$  of form  $i : C \sqsubseteq D$ ,  $i : C(a)$ .

As described in [13], the *SHIQ*-tableau works on a so called “completion forest”, a collection of trees whose root nodes correspond to instances in A-box. Given a knowledge base, the algorithm initializes a completion forest  $\mathcal{F}$  with a set of root nodes  $\mathbf{x}_0 = \{x_0^k\}$  corresponding to a set of instances  $b_k$  in A-box, labels each  $x_0^k$  with a set  $\mathcal{L}(x_0^k)$  of concepts  $C$  for each concept assertion  $C(b_k)$  in A-box, and finally draws an edge between  $x_0^k$  and  $x_0^m$  for each role assertion  $R(h_k, h_m)$  in A-box. After that, the set of *SHIQ* completion rules expanding the forest  $\mathcal{F}$  is applied. The fully expanded forest then represents a model of the knowledge base. To test entailment of arbitrary assertion  $X(a)$ ,  $\neg X(a)$  is added to A-box and further the tableau is expanded to see whether a model of such knowledge base can be constructed or not.

To accommodate the knowledge propagation from  $\mathcal{K}_1$  to  $\mathcal{K}_2$  in  $\mathfrak{R}_{12}$ , we intervene in the completion process of **Tab**<sub>2</sub> in order to capture new facts induced by bridge rules and individual correspondences. Hence, we get a *distributed tableaux procedure* **DTab**<sub>2</sub> which extends **Tab**<sub>2</sub> with two additional expansion rules:

$\mathfrak{C}_{12}$ -rule:

if 1.  $x \in \mathbf{x}_0$ , such that  $x$  is a node corresponding to individual  $b$  and  $1 : a \mapsto 2 : b$ ,  
 $\mathbf{H} \subseteq \{H_k \mid 1 : B_k \xrightarrow{\Xi} 2 : H_k \in \mathfrak{B}_{12}\}$ ,  
 $\mathbf{B} = \{B_k \mid H_k \in \mathbf{H}, 1 : B_k \xrightarrow{\Xi} 2 : H_k \in \mathfrak{B}_{12}\}$ ,  
 2. **Tab**<sub>1</sub>(( $\bigsqcup \mathbf{B}$ )( $a$ )) = *true* for  $\bigsqcup \mathbf{H} \notin \mathcal{L}(x)$ ,  
 then  $\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{\bigsqcup \mathbf{H}\}$

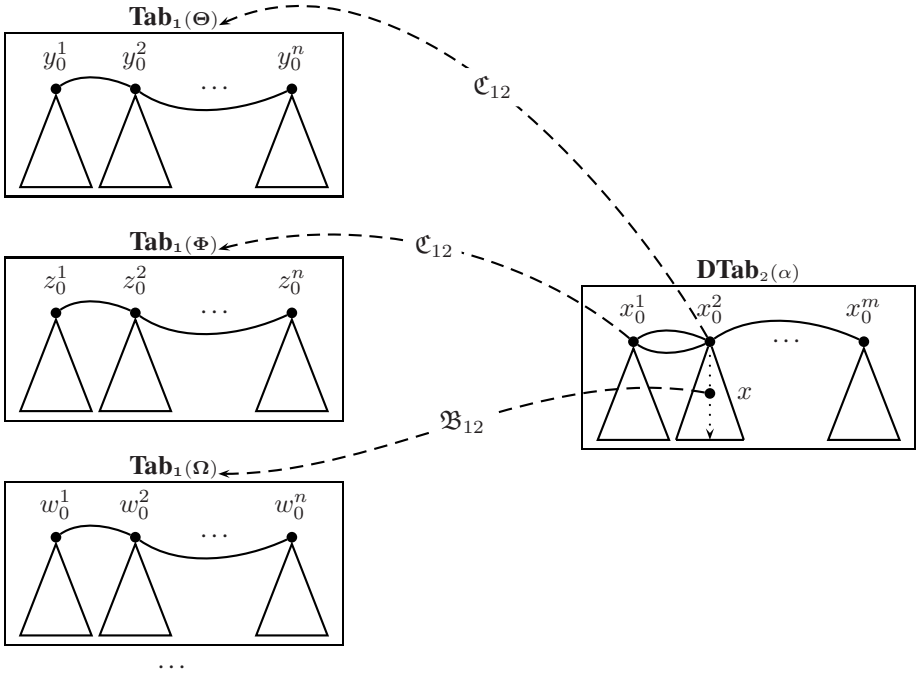
$\mathfrak{B}_{12}$ -rule:

if 1.  $G \in \mathcal{L}(x)$ , such that  $1 : A \xrightarrow{\Xi} 2 : G \in \mathfrak{B}_{12}$ ,  
 $\mathbf{H} \subseteq \{H_k \mid 1 : B_k \xrightarrow{\Xi} 2 : H_k \in \mathfrak{B}_{12}\}$ ,  
 $\mathbf{B} = \{B_k \mid H_k \in \mathbf{H}, 1 : B_k \xrightarrow{\Xi} 2 : H_k \in \mathfrak{B}_{12}\}$ ,  
 2. **Tab**<sub>1</sub>( $A \sqsubseteq \bigsqcup \mathbf{B}$ ) = *true* for  $\bigsqcup \mathbf{H} \notin \mathcal{L}(x)$ ,  
 then  $\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{\bigsqcup \mathbf{H}\}$

The principle idea of these additional expansion rules consists in implementing backward versions of bridge and individual correspondences operators introduced in Section 3.2. According to rule  $\mathfrak{C}_{12}$ , if **DTab**<sub>2</sub> encounters a root node  $x$  connected by an individual correspondence, then a disjunction of concepts  $\bigsqcup \mathbf{H}$  should be added to the label  $\mathcal{L}(x)$  if  $\bigsqcup \mathbf{H}(x)$  is entailed by interaction of individual correspondence with into-rules. To determine this entailment, **DTab**<sub>2</sub> remotely requests foreign **Tab**<sub>1</sub> to check if it is the case that  $\bigsqcup \mathbf{B}(b)$  in  $\mathcal{K}_1$ .

The role of  $\mathfrak{B}_{12}$ -rule is to analyse the nodes of completion forest and import consequences of subsumption propagations. If **DTab**<sub>2</sub> encounters a node  $x$  which contains a label  $G$  connected by an onto-bridge rule, then if  $G \sqsubseteq \bigsqcup \mathbf{H}$  is entailed by the bridge rules, the label  $\bigsqcup \mathbf{H}$  is added to  $x$ . While in order to determine the entailment, **DTab**<sub>2</sub> invokes the procedure **Tab**<sub>1</sub> with a question whether a subsumption  $A \sqsubseteq \bigsqcup \mathbf{B}$  holds in  $\mathcal{K}_1$ .

Graphically, the distributed execution of  $\mathbf{DTab}_2$  can be depicted as follows:



**Theorem 2 (Termination, Soundness, Completeness).** Given  $\mathcal{SHIQ}$  DL knowledge bases  $\mathcal{K}_1$  and  $\mathcal{K}_2$ , let  $\mathfrak{K}_{12} = \langle \langle \mathcal{T}_1, \mathcal{T}_2 \rangle, \mathfrak{B}_{12} \rangle, \langle \{A_1, A_2\}, \mathcal{C}_{12} \rangle \rangle$  be a distributed knowledge base. Then, given a  $\mathcal{SHIQ}$  statement  $\alpha$

1. a distributed procedure  $\mathbf{DTab}_2(\alpha)$  terminates, and
2.  $\alpha$  is satisfiable in  $\mathcal{K}_2$  with respect to  $\mathfrak{K}_{12}$  if and only if  $\mathbf{DTab}_2(\alpha)$  yields a complete and clash-free completion forest.

It can be shown that the proposed algorithm enjoys generalization to arbitrary number of  $\mathcal{SHIQ}$  knowledge bases participating in DDL, and moreover can be extended to distributed knowledge bases containing cyclical paths of bridge rules and individual correspondences. For the sake of clarity, we omit the discussion of these generalizations and refer the reader to the technical report [16] for details.

## 4 Implementation and Application

In this section we first outline the implementation of the distributed tableaux procedure on top of the DRAGO DDL Reasoner; second we describe its application to a problem of instance migration between heterogeneous ontologies.

### 4.1 DRAGO DDL Reasoner

DRAGO DDL Reasoner implements a peer-to-peer architecture for reasoning with a set of ontologies  $\{O_i\}_{i \in I}$  interconnected by semantic mappings  $\{M_{ij}\}_{i \neq j \in I}$ . The principal component of DRAGO is the *DRAGO Reasoning Peer*, or shortly *DRP*. Each

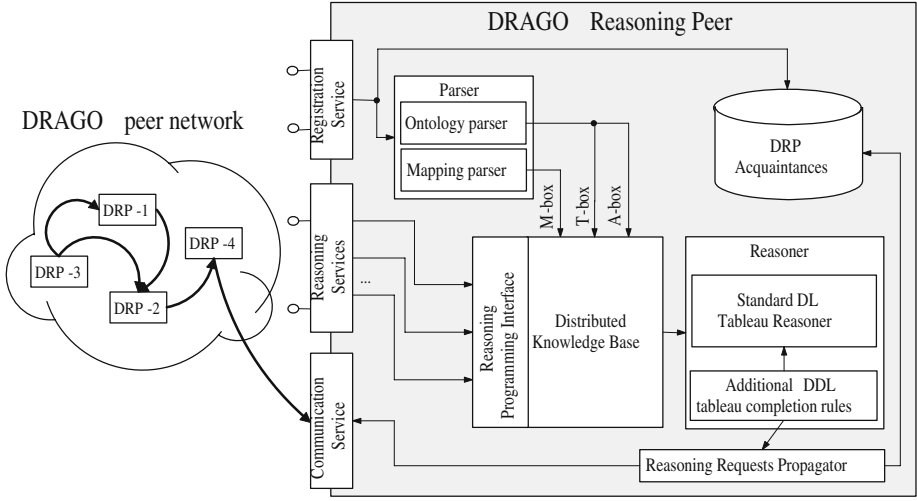


Fig. 2. DRAGO Architecture

DRP  $P$  can host a set  $\mathbf{O}_P = \{O_i\}_{i \in I_P}$  (where  $I_P \subseteq I$ ) of ontologies, and a set of mappings  $\mathbf{M}_P = \{M_{ij}\}_{i \in I, j \in I_P}$  incoming into  $\mathbf{O}_P$ . Mappings can come both from ontologies of the peer  $P$ , or from ontologies of other peers. Each DRP  $P$  supports both *local* and *distributed* reasoning tasks on each ontology in  $\mathbf{O}_P$ . Local reasoning tasks are standard DL reasoning tasks defined on one single ontology. Distributed reasoning tasks are those defined on the distributed knowledge base induced by the ontologies and mappings managed by all the DRP's. When the DRP  $P$  executes a distributed reasoning task on an ontology with mappings coming from another peer  $Q$ ,  $P$  submits the reasoning sub-tasks to  $Q$ . Figure 2 sketches the overall architecture of DRAGO and displays principle components forming the Reasoning Peer.

The *Reasoner* is the central component of DRP. It is implemented by extending a standard tableaux based DL reasoner with additional completion rules that implements the distributed reasoning. DRAGO is based on the Pellet reasoner, an open-source Java implementation of tableau for reasoning with OWL ontologies [17]. In accordance with algorithm presented in Section 3.3 these new completion rules comprise the bridge rule and individual correspondence. Due to the remark to Theorem 1 on independence of terminological and assertional propagation, we reused the bridge completion rule from available purely terminological version of DRAGO, and implemented an individual completion rule, new due to the present paper.

Practically, DRAGO works with ontologies represented in OWL [3] and semantic mappings encoded in C-OWL [5]. Syntactically, C-OWL extends OWL with constructs for specification of semantic mappings, while semantically it is founded on the presented DDL framework.

## 4.2 Execution of Migrations with DRAGO

In the following we describe how to use DRAGO to execute the task of instance migration between heterogeneous ontologies. Given a source ontology  $O_s$  and a target ontology  $O_t$ , the task of instance migration from  $O_s$  to  $O_t$  can be encoded into the following steps:

1. Match concepts of  $O_t$  with concepts of  $O_s$  and then encode discovered semantic relations into a set  $\mathfrak{B}_{st}$  of bridge rules between  $O_t$  and  $O_s$ . This task can be done manually or with the help with some (semi-)automatic ontology matcher.
2. Choose a translation  $f_{st}$  from individuals of  $O_s$  to individuals of  $O_t$ , and generate the set of individual correspondences  $\mathfrak{C}_{st} = \{s : x \mapsto t : f_{st}(x) | x \text{ individual of } O_s\}$ . The simplest case of translation function is the identity function, when exactly the same set of individuals of  $O_s$  are included into the target ontology  $O_t$ . However, this is not always the case. E.g., if  $O_s$  and  $O_t$  follow different naming conventions, then individuals of  $O_s$  are needed to be renamed in accordance with rules  $O_t$ .
3. Instantiate DRAGO Reasoning Peer  $DRP_s$  for  $O_s$  and  $DRP_t$  for  $O_t$  with semantic mapping  $\mathfrak{B}_{st} \cup \mathfrak{C}_{st}$ .
4. Ask  $DRP_t$  to classify translated individuals in accordance with  $O_t$ .

## 4.3 Experimental Run

To see the instance migration in work and get the practical impression from the implemented distributed tableaux, we emulated and executed the motivating scenario described in the introduction. As a source of instances, we used a publicly available ontology populated with data on publications at the Semantic Web Conference<sup>5</sup>. To evidence the correctness of migration, as a target ontology we used a source ontology with all instance data removed from it. As required, the establishment of into-bridge rules between the same concepts of source and target and the application of identity function to translation of source ontology instances yields the exactly the same classification of migrated instances in target ontology.

Besides the correctness, the performed practical run demonstrated the necessity of developing optimization strategies for completion of distributed tableaux for instance migration. This is due to the relatively slow speed of migration (e.g., the reclassification of 50 instances of the selected source ontology through 30 established into-bridge rules takes around 5 minutes). The slow speed is a consequence of the necessity to consider all possible disjunctions of concepts connected by into-bridge rules when completing distributed tableaux. In the next study, we investigate possible optimization strategies for reducing the amount of disjunctions to be considered to still guarantee the complete reasoning result.

## 5 Related Work

The problem of heterogeneity is one of the crucial issues to be resolved on the semantic web. This explains the big research interest to devising frameworks capable of

<sup>5</sup> <http://annotation.semanticweb.org/iswc/iswc.owl>

representing and reasoning with multiple ontologies interrelated by semantic mappings. While DL is already a standard for working with web ontologies, the question of formal representations and reasoning with mappings is still a subject to the standardization. Hence, multiple frameworks co-exist.

In *OntoEngine* [7], the authors address the problem of translating instances from a source ontology into a heterogeneous target. In their approach mappings between ontologies are represented using the same primitives as for encoding knowledge within ontologies themselves, i.e., using “subClassOf”, “subPropertyOf”, etc. axioms. The reasoning with mapping is based on idea of merging ontologies together with the mappings into a single ontology, in which further the standard reasoner can execute instantiation queries over vocabulary of target ontology. The main drawback of this approach is its strong centralization, which is not typically affordable on the web.

In contrast, the *SomeWhere* [9] targets a question of decentralized approach to querying heterogeneous ontologies. Similarly to *OntoEngine*, mappings in *SomeWhere* has a form of a subsumption statements, but the reasoning is based on rewriting techniques for combining reasoning over heterogeneous ontologies. The big advantage of the presented approach is its scalability, while the disadvantage is its limitation to a “propositional” ontologies, containing only disjunction, conjunction and negation.

Another recent example of decentralized infrastructure for querying distributed ontologies is *KAONp2p* [11][12]. The authors adopt the approach of [6] to express mappings as correspondences between conjunctive queries over ontologies. The querying further requires the terminologies and mapping to be merged into a single global ontology, while instance data is then retrieved from distributed instance storages.

The recent study of query answering in *distributed description logics* has been proposed in [1]. The main idea consist in constructing a closure ontology by forward propagating, via DDL mappings, relevant axioms contained in other mapped ontologies (in a vein of vanilla implementation of DDL reasoner discussed in the current study). Doing so, further enables reformulation of distributed query answering problem into local query answering. Although the approach of [1] is sound, the authors point out the incompleteness of their study.

Another important framework is  $\mathcal{E}$ -connections [14]. Original purpose of  $\mathcal{E}$ -connections is to aggregate ontologies that model different (non-overlapping) aspects of the world, rather than integrate those overlapping as in DDL. Nonetheless, it has been shown in [14] that mathematically DDL constructs can be simulated in  $\mathcal{E}$ -connections, however sacrificing the directionality of knowledge propagation. Another difference concerns with reasoning approach. In contrast to distributed coordinating tableaux in DDL, in  $\mathcal{E}$ -connections a global tableau, both theoretically and practically, needs to be constructed.

## 6 Conclusion

In the present study, we investigated a task of correct and complete migration of instances of one ontology into another heterogeneous ontology. We formally grounded our approach on DDL framework, which allowed us to instantiate the problem of migration into the problem of reasoning with instances in DDL distributed knowledge base.



We theoretically formalized this inference and defined the distributed tableau algorithm for reasoning with multiple *SHIQ* DL ontologies. To demonstrate the feasibility, we implemented the preliminary version of the algorithm in DRAGO Reasoner and applied it to a simple migration task.

## References

1. Alkhateeb, F., Zimmermann, A.: Query Answering in Distributed Description Logics. In: Proc. of the 1st Conference on New Technologies, Mobility and Security (NTMS) (2007)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook: Theory, Implementation, and Applications (2003)
3. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Andrea Stein, L.: OWL Web Ontology Language Reference. In: Web, O.W.L. (ed.) W3C Recommendation (February 2004), <http://www.w3.org/TR/owl-ref>
4. Borgida, A., Serafini, L.: Distributed Description Logics: Assimilating Information from Peer Sources. *Journal of Data Semantics* 1, 153–184 (2003)
5. Bouquet, P., Giunchiglia, F., van Harmelen, F., Serafini, L., Stuckenschmidt, H.: Contextualizing Ontologies. *Journal on Web Semantics* 1(4), 325–343 (2004)
6. Calvanese, D., De Giacomo, G., Lenzerini, M.: A Framework for Ontology Integration. In: SWWS-2001. Proc. of the Semantic Web Working Symposium, pp. 303–316 (2001)
7. Dou, D., McDermott, D., Qi, P.: Ontology Translation on the Semantic Web. *The Journal on Data Semantics* 3360, 35–57 (2005)
8. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer, Heidelberg (2007)
9. Goasdoué, F., Rousset, M-C.: Querying Distributed Data through Distributed Ontologies: a Simple but Scalable Approach. *IEEE Intelligent Systems* 18(5), 60–65 (2003)
10. Haarslev, V., Moller, R.: RACER System Description. In: IJCAR-2001. Proceedings of the International Joint Conference on Automated Reasoning, pp. 701–706 (2001)
11. Haase, P., Motik, B.: A Mapping System for the Integration of OWL-DL Ontologies. In: IHIS 2005. Proceedings of the First International Workshop on Interoperability of Heterogeneous Information Systems, pp. 9–16. ACM Press, New York (2005)
12. Haase, P., Wang, Y.: A Decentralized Infrastructure for Query Answering over Distributed Ontologies. In: SAC-2007. Proceedings of the 22nd Annual ACM Symposium on Applied Computing, ACM Press, New York (2007)
13. Horrocks, I., Sattler, U., Tobies, S.: Reasoning with Individuals for the Description Logic SHIQ. In: McAllester, D. (ed.) *Automated Deduction - CADE-17*. LNCS, vol. 1831, pp. 482–496. Springer, Heidelberg (2000)
14. Kutz, O., Lutz, C., Wolter, F., Zakharyashev, M.: E-Connections of Abstract Description Systems. *Artificial Intelligence* 156(1), 1–73 (2004)
15. Serafini, L., Borgida, A., Taminlin, A.: Aspects of Distributed and Modular Ontology Reasoning. In: IJCAI 2005. Proc. of the 19th Joint Conference on Artificial Intelligence (2005)
16. Serafini, L., Taminlin, A.: Reasoning with Instances in Distributed Description Logics. Technical report, Fondazione Bruno Kessler - IRST (2007), <http://sra.itc.it/people/taminlin/publications/2007/iswc/tr.pdf>
17. Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A Practical OWL-DL Reasoner. *Journal of Web Semantics* (2006)
18. Tsarkov, D., Horrocks, I.: FaCT++ Description Logic Reasoner: System Description. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 292–297. Springer, Heidelberg (2006)

# ALLRIGHT: Automatic Ontology Instantiation from Tabular Web Documents

Kostyantyn Shehekotykhin, Dietmar Jannach, Gerhard Friedrich, and Olga Kozeruk

Universität Klagenfurt, Universitätsstrasse 65, 9020 Klagenfurt, Austria, Europe  
{kostya,dietmar,gerhard,olga}@ifit.uni-klu.ac.at

**Abstract.** The process of instantiating an ontology with high-quality and up-to-date instance information manually is both time consuming and prone to error. *Automatic ontology instantiation from Web sources* is one of the possible solutions to this problem and aims at the computer supported population of an ontology through the exploitation of (redundant) information available on the Web.

In this paper we present ALLRIGHT, a comprehensive ontology instantiating system. In particular, the techniques implemented in ALLRIGHT are designed for application scenarios, in which the desired instance information is given in the form of *tables* and for which existing Information Extraction (IE) approaches based on statistical or natural language processing methods are not directly applicable.

Within ALLRIGHT, we have therefore developed new techniques for dealing with tabular instance data and combined these techniques with existing methods. The system supports all necessary steps for ontology instantiation, i.e. web crawling, name extraction, document clustering as well as fact extraction and validation. ALLRIGHT has been successfully evaluated in the popular domains of digital cameras and notebooks leading to a about eighty percent accuracy of the extracted facts given only a very limited amount of seed knowledge.

## 1 Introduction

The information gathering process for specialized *knowledge-based services* [1] can be a time consuming and error prone task. In order to at least partially automate this task, several *Web Mining* and *Information Extraction* techniques have been proposed over the last years as one cannot yet expect that Web documents are already sufficiently semantically annotated. Some of them, for instance, rely on the redundancy of information on the Web and use statistical methods [7,4,3,5], others use natural language processing (NLP) techniques [19,20] to extract the required knowledge from unstructured documents. It has also been shown that domain *ontologies* can be useful to support knowledge extraction, which in turn means that the information extraction problem can be generalized to the problem of finding and inserting information into the system's knowledge base that matches a given ontology [1], a process also referred to as *ontology instantiation* or *ontology population*.

The herein described ALLRIGHT system was designed based on the requirements that came up in the development of such a real-world knowledge-based service. In particular, the problem was to set up and maintain an up-to-date knowledge base of detailed

item descriptions in an online store. These in-depth product descriptions should subsequently serve as a basis for building content-based recommender and product comparison applications as described in [9]. One can for example easily imagine that maintaining data sets for more than 400 digital camera models by hand and updating the knowledge base whenever new products are available is time consuming and costly. Our experiences also showed that the quality and level of detail of commercially available data sets is not sufficient for our purposes.

Knowing that in many domains the required item information is already available on the Web (e.g., on manufacturer homepages or community portals), the goal was thus to develop a Web Mining System (WMS) that is capable of finding and extracting this knowledge automatically. What was soon recognized, however, is that existing Web Mining techniques like the ones mentioned above are not directly applicable, mostly because the instance information is in many cases described in *tabular form*. Note that these tables are not necessarily constructed from underlying databases which means that neither SPARQL nor “Hidden-Web” techniques can be used.

In general, tables are well-suited for the human reader as they are a compact and precise form of representing information. The problem with tables in the context of existing information extraction techniques, however, is that they contain nearly no information that can be exploited by NLP methods. In addition, with respect to clustering methods, the problem is that pages with *the same sets of keywords/tokens* (from the same portal) describe *different items*. In the ALLRIGHT system these particularities have consequently been taken into account and new techniques for dealing with tabular information have been developed.

Nonetheless, ALLRIGHT was designed as a more general, domain independent ontology instantiation system in the sense of [11]: The input to the extraction system is thus a domain ontology that describes the structure of and the relations between the items to be searched for. The extraction process itself is structured in a series of tasks like Web crawling, name extraction, validation, document clustering, and fact extraction.

We see the contributions of ALLRIGHT to the state of the art as follows. First, we show how the approach of [11] to automatic ontology instantiation can also be applied to domains in which data is given in tables, which is a common representation form for all kinds of data on the Web like personal information, geographical data and so forth. In addition, our usage of ontologies is broader than in previous approaches, as we also apply model-based diagnosis techniques [10] to automatically identify conflicts between the domain ontology and the extracted knowledge and are thus able to generate corresponding resolution proposals. Our contributions also include a new crawling method for fast location of tabular descriptions, a visual table identification technique, and a novel way of applying clustering algorithms to deal with tabular descriptions. Finally, given the promising evaluation results with an accuracy of about eighty percent, we show how Semantic Web technology, ontologies, and a combination of new and existing extraction techniques can help us to better deal with real-world, industrial problem settings.

In the subsequent sections of the paper we will - based on a continuing example problem from the domain of digital cameras - sketch the overall process flow, give technical details of the individual extraction steps, and report evaluation results for the individual subtasks.

## 2 Automatic Ontology Instantiation

Fig. 1 sketches the overall ontology instantiation process in one possible example setting, in which detailed product data should be extracted for a content-based recommender system in an online store. First, a domain ontology is developed, which describes the basic structure and characteristics of the items to be searched for. In our example configuration, some parts of the knowledge are imported from the ADVISOR SUITE system [8], in which for instance the list of characteristics for an item to be recommended can be defined. From the domain ontology (more details will be given in subsequent sections), the system generates keywords (1) that are used for crawling the Web for relevant Web pages (2,3) xCRAWL. Next, the downloaded pages are analyzed by the Identification Component (4) in order to determine whether they *really* contain item descriptions. Again, this analysis is done on the basis of the knowledge from the ontology (5). The filtered set of documents is then forwarded (6) to a module which first clusters those pages that describe the same products, then extracts the specific facts for each product and feeds the new knowledge back to the ontology (7). Note that in principle, also an alternative strategy for this step is possible, i.e., first perform fact extraction and then try to remove the duplicates [16]. Still such systems require more detailed seed knowledge and specialized and complex fact recognition techniques. In all phases, feedback and validation loops are contained which are used to refine the knowledge acquired so far. Within steps 8,9, and 10 relationships between the successfully identified ontology instances and other shop items are instantiated. In the digital camera domain, this could for instance be the problem of finding out which accessories are compatible with a certain camera model. However, these steps are only covered superficially in this paper as we want to focus on the extraction of product descriptions.

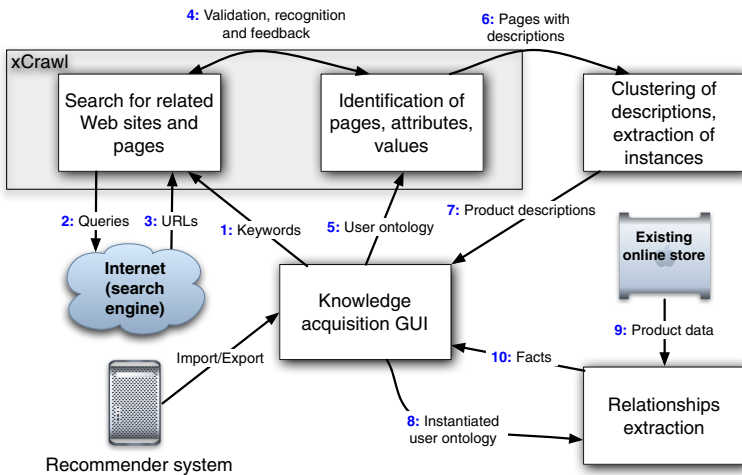


Fig. 1. ALLRIGHT ontology instantiation process in application scenario

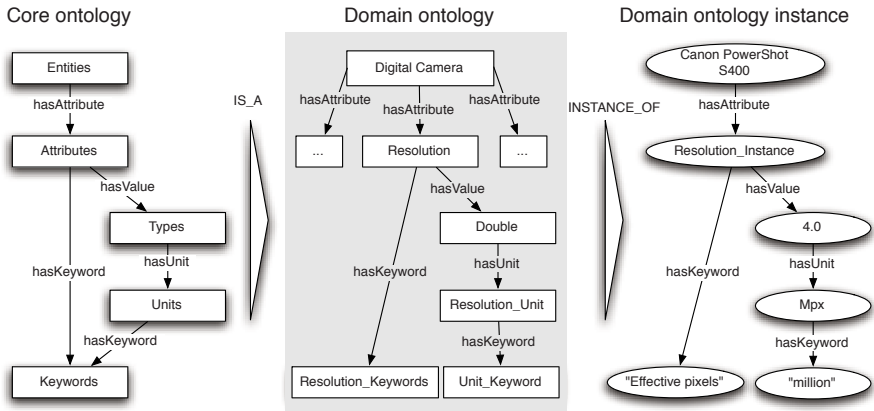


Fig. 2. Fragments of ontologies used in the ALLRIGHT system

## 2.1 Ontologies in ALLRIGHT

In the ALLRIGHT system, three “levels” of ontologies are of interest, see Fig. 2 for example fragments. In the product data extraction scenario, the predefined *core ontology* for instance describes that we search for entities that have attributes of given types and certain units of measurement. Both the attributes and the units can be annotated with string-typed keywords.

When ALLRIGHT is to be configured for a certain domain, i.e., digital cameras in our scenario, a *domain ontology* has to be derived from the core ontology. In the example, we thus state that “resolution” is a property of interest for entities of type “digital camera”. The resolution is given as a real-valued number and is measured in mega pixels. Appropriate keywords could be “effective pixels” and “million”, respectively. Note that the task of defining the domain ontology corresponds to the definition of *seed knowledge*, which is required in every Web Mining System. Technically, the domain ontology is stored as a Web Ontology Language (OWL) document. The (graphical) definition and maintenance of the model can thus be accomplished with the help of any general purpose ontology editing tool. In our application scenario, however, we used the built-in graphical knowledge acquisition tools from ADVISOR SUITE [8].

The right-most part of Fig. 2 shows how one specific ontology *instance* in the domain could look like. In contrast to the definition of the domain ontology, which is done manually, the extraction of all corresponding instances that match the domain ontology is the automated task of the ALLRIGHT system. We will subsequently show how the domain ontology is exploited within the different subtasks and summarize the other advantages of using ontologies in that context.

## 2.2 Ontology-Based Data Collection

Once the domain ontology is defined, the next task of the system is to locate as many Web documents as possible that contain information which matches the ontology (Steps 1 to 5 in Fig. 1). The resulting set of pages, which may contain multiple documents per instance, are then forward to the extraction component (Step 6).

```

function FIND-AND-EXTRACT returns a collection of extracted tables
  inputs: domainOntology, an ontology derived by a user from the core ontology.
           maxIterations, (number) a convergency threshold of the crawler.
  local variables (initially empty):
           Pages, a collection of pages downloaded by the crawler.
           Hubs, a collection of pages that are linked with many authorities.
           Authorities, a collection of pages with tabular descriptions of domain
           ontology instances {
  Authorities := GETSEEDDATA( domainOntology )
  Sites := GETWEBSITES(Authorities);
  for each ( site ∈ Sites ) {
    do {
      Authority := SELECTRANDOM( Authorities, site )
      newPages := PERFORMRANDOWALK( Authority, Pages )
      Pages := Pages ∪ newPages
      Authorities := Authorities ∪ ANALYZEPAGES( domainOntology, newPages )
      Hubs := Hubs ∪ EXTRACT( Pages, Authorities )
    } while (NONNEWHUBSFOUND( maxIterations ))
  }
  for each ( hub ∈ Hubs ) {
    Authorities := Authorities ∪ GETALLAUTHORITIES( domainOntology, hub )
  }
  return STOREDOCUMENTS( Authorities )
}

```

**Fig. 3.** ALLRIGHT data collection algorithm

The ALLRIGHT system implements the subsequently described specific data collection algorithm (Fig. 3), which represents a novel combination of new and existing techniques in the area.

**GETSEEDDATA.** In the first step, a *seed* collection of relevant documents and web sites is created with the help of an *Automatic Query Generation* (AQG) technique (cf. [12]). Note that in general, there exist two commonly used approaches to automated data collection: A Focused Crawler efficiently seeks out documents about a specific topic and is guided by the link structure and the content of the Web [2]. In opposite to crawlers used by search engines that perform breadth-first search, a focused crawler employs depth-first search that is usually guided by some scoring algorithm. The other approach, Automatic Query Generation, retrieves documents by sending sophisticated queries to search engines and analyzes the retrieved documents with the help of domain-specific classifiers. The outcomes of this analysis process are then used to refine the queries in order to improve the results.

Note that we use AQG in the first phase because it helps us to retrieve the set of interesting Web documents and *sites* quickly (see [12] for a comparison of data collection strategies). Later on, however, we will use a focused crawling technique in combination with the “index-page heuristic” to achieve high recall: The basic assumption of the heuristic is that Web site creators typically arrange the documents of a site in a way that they can be easily accessed by users through index pages or hierarchies.

For the query generation task, our crawler produces an ordered list of all attribute keywords from the domain ontology whose order is based on the number of hits returned for search engine queries that consist only of the keyword itself. This ordering helps us to distinguish general keywords from very specific ones. Next, the AQG system generates further queries to a search engine, where each query is a combination of keywords with the maximum length of  $m$  query terms. The generation algorithm starts from the middle of the ordered list and proceeds in both directions simultaneously. Each generated query is sent to a search engine and the first  $n$  URLs are saved. In our experiments we used  $m = 10$ , because most of the search engines accept no more than 10 terms in one query, and  $n = 30$ , because 30 descriptions per instance is a statistically good sample size.

Also within the GETSEEDDATA method, a first *validation* phase (identification step in Fig. 1) takes place, i.e., the goal now is to check whether a downloaded page actually contains a tabular description of an ontology instance. The *validation process* begins with the analysis of individual Web pages with “content spotters” which is similar to systems presented in [13,6]. Content spotters are used to identify and mark up interesting text positions for later processing (extraction). To some extent they also mimic human behavior when reading a product description: A document is first roughly skimmed for areas of interest detecting keywords or familiar text constructs like attribute values (numbers) that are followed by units (keyword strings). Once such “context frame” is detected, the reader will take a closer look at these contents or page fragments.

In the ALLRIGHT system, content spotters are automatically constructed from the core ontology and the domain ontology, each one created to find a special keyword or structural feature. For example one spotter could look for definitions of number ranges (e.g. 10 to 12 cm) while another one seeks string enumerations and a third looks for numbers accompanied by units. Once a spotter finds an interesting region it will continue to examine the region in more detail and may for instance check whether the found number is within the allowed range as defined in the ontology. The content-spotting task itself is subdivided into several phases: First, simple spotters are applied on the document. Then the results are merged and potential conflicts resolved before more advanced content spotters operate on the matches found in the previous phase. The final output of the content spotters is an annotated text document in which interesting positions are marked and classified.

Based on these annotated documents, the ALLRIGHT system then uses a new *visual table recognition* method [15] to check whether the document contains tabular information which can be extracted. The method retrieves the coordinates of text boxes from a browser integrated into the crawling component and tries to draw lines that separate the text boxes (words) of a rendered page by using a system of heuristics of possible text alignments. Then the method analyzes those parts of the documents which were annotated by the content spotters and decides where the target table is placed on the page. The most plausible area is extracted for further processing and the page will be added to the set of *Authorities*, i.e., interesting pages.

**GETWEBSITES.** This small function takes the identified web pages as an input and extracts the basic URLs of the corresponding web sites like e.g. a manufacturer



home page. These URLs serve as a starting point for the subsequent focused crawling phase.

**FOR EACH SITE  $\in$  SITES.** The main goal of this block is to identify *hubs* like index pages on each web site (as it is defined in [14]). This is done in an iterative process that continues until no new hubs are found over a predefined number of iterations. Our experiments showed that using an iteration limit of around 50 was appropriate in our setting. Lower values led to insufficient crawler performance and greater ones did not improved the crawler’s result considerably and required much longer running times.

The crawler selects randomly one of the authorities (SELECTRANDOM) at the beginning of each iteration. This authority is used as a starting point for a *random walk with restart* [17] strategy to explore the Web site graph until the the first restart event happens. Note, that this strategy has the advantage that the search process does not get stuck in non-interesting or well-explored branches of the graph. The newly found pages are added to the local set of all *Pages* found so far in order to prevent the crawler from downloading the same page twice.

Next, ANALYZEPAGE searches for new authorities within the newly identified pages by using the same identification and validation techniques as described in the GETSEEDDATA function, i.e., content spotters and table recognition.

Finally, we analyze all pages found so far (EXTRACT) by applying the HITS algorithm [14] in order to detect new hubs for already known authorities.

**FOR EACH HUB  $\in$  HUBS.** Given all the hubs from the previous step, the algorithm now downloads all the pages that are referenced by each hub. All downloaded pages are then analyzed (GETALLAUTHORITIES) and if new authorities are found (again based on validation and identification), they are stored in the corresponding list.

**STOREDOCUMENTS.** Finally, the full list of extracted Web pages is stored for the subsequent document clustering and fact extraction steps.

### 2.3 Document Clustering & Instance Name Recognition


Most of the latest statistics-based WMS like KnowItAll [7], Web $\rightarrow$ KB [4], C-Pankow [3], SemTag [5], and other systems exploit the redundancy of information in the Web to acquire knowledge. These systems apply different comparison and classification methods to analyze document collections that contain redundant descriptions of a target class and/or their instances: An instance description of a class, like for example a specification of a digital camera model, can be published on different Web sites. Our goal is thus to determine in the next step which pages describe the *same* instance in order to extract as much information as possible from different sources.

The main problem in this context however is that pages describing the same camera can significantly differ from each other in three main dimensions, see Fig. 4:

- *Presentation.* The left table is not structured like the original manufacturer’s one, which contains *sections* like “Image capture device” or “Lens”. Moreover, some distinct attributes of one table can, in fact, be described in on single attribute in the other one. For example, the “Type” attribute in the right table covers the information of the “Sensor type” and “Sensor size” attributes of the left table.



**Canon PowerShot S400**



Image

More information

Discussion

Owners opinions

Support this site by purchasing from our affiliate merchants

Format

Price (street)

Also known as

Release Status

Max resolution

Low resolution

Image ratio w:h

Effective pixels

Sensor photo detectors

Sensor size

Sensor type

**Type of Camera**

Type of Camera: Compact digital still camera with built-in flash and 3.0x Optical / Digital / 11x Combined Zoom

**Image Capture Device**

Type: 1/1.8 inch charge coupled device (CCD)

**Total Pixels**: Approx. 4.1 million

**Effective Pixels**: Approx. 4.0 million

**Lens**

Focal Length: 7.4 - 22.2mm (35mm film equivalent: 36 - 108mm)

Digital Zoom: 3.6x

Focusing Range: Normal AF: 46cm/1.5 ft. - infinity Macro AF: 5 - 46cm (2.0 in. - (W), and 30 - 46cm (1.0 - 1.6 ft.) (T). Manual focus: 10cm/4 in. - infinity (W), 30cm/1 in. - infinity (T)

Autofocus System: 9-point AiAF( Automatic Focus Point Selection)/ 1-point AF (Fix center)

**Viewfinders**

Optical Viewfinder: Real-image optical zoom viewfinder

LCD Monitor: 1.5 inch low-temperature polycrystalline silicon TFT color LCD

LCD Pixels: Approximately 118,000 pixels

LCD Coverage: 100%

**Aperture and Shutter**

Fig. 4. Different representation of information for the same camera

- *Detail coverage.* Review sites, like dpreview.com, often publish attributes of products that are not mentioned by manufacturers, but may be important for customers. Canon.com for instance does not report the capacity of the memory card shipped with a camera, which, however, can be found on dpreview.com in the “Storage included” attribute.
- *Content.* Attributes that are named differently can identify the same property of a camera, like “Total pixels” in the right table corresponds to “Sensor photo detectors” in the left one. In addition, there can be also a mismatch between the values of attributes that describe the same property of a camera. In Fig. 4, the values of the “Effective pixels” attribute are actually different.

Overall, the existing approaches to document clustering are designed to work with general sources written in natural language, i.e., they analyze documents or their parts “as is”. Web→KB [4] for example uses a combination of three Naive Bayes classifiers trained to group the available documents based on full text, headers/titles and hyperlinks (anchor tags). Another technique is implemented in the KnowItAll [7] system, which uses web statistics acquired from search engines in order to determine the plausibility of extracted facts.

If, however, instances are described in tabular form as in our case, the applicability of these clustering methods is limited, because tables that describe *different* instances typically contain very common sets of words (like attribute names) and almost no grammar. On the other hand, it may also be possible that there are two documents that describe the same instance, but use non-overlapping sets of tokens, which means that standard clustering techniques would put them into different groups.

Therefore, in the ALLRIGHT system, we aim to cluster the documents indirectly by means of some previously extracted unambiguous instance identifiers like an *instance name*. These extracted names can consequently also be used to retrieve additional documents from the Web, as to increase redundancy and to improve the results of a

statistics-based Web mining process. Unfortunately, such names or other identifiers are usually not explicitly given and there is no general rule where such identifiers typically appear on a Web page. Therefore, in our approach, we exploit the information contained in the *title* tag of the Web page that describes the instance as well as *hyperlinks* that point to that page. Of course, depending on the domain, also other tags can be considered even if they contain additional and irrelevant information.

Our current approach, which is more general in comparison to a previously reported implementation, see [11], uses the *X-Means* clustering algorithm [18] in order to extract the most significant part (name candidate) of strings contained in the title and hyperlinks. These name candidates are then used to group the instance descriptions so that each group contains only descriptions of one instance. Furthermore, a new name validation method is used to extract the real instance name from the candidate, which finally leads to high precision and recall (F-measure >0.9) for the instance name recognition process.

## 2.4 Fact Extraction and Instance Generation

In this last step of the ontology instantiation process, we finally aim at the extraction of the exact or most plausible information from the many documents retrieved in the previous phases and insert the extracted instances into the ontology.

As a starting point we use the clustered and annotated documents (authorities) in which possible attribute-value pairs were marked up by the content spotters and for which instance names were determined in the previous step. What is thus left to do is to resolve those problematic situations in which we for instance have multiple value candidates for an individual attribute or have values that do not match the given domain ontology.

In a first step we therefore determine for each identified attribute-value pair if something has to be done and differentiate between the following situations.

The found attribute name is valid (there exists an unambiguous match with the class defined in the domain ontology) and the value is consistent with the domain ontology.

(2) The attribute name is valid, but we have multiple value candidates that were found by the content spotter.

(3) The attribute is valid, but the value found is not consistent with the ontology, i.e. it violates a value restriction defined in the domain ontology.

(4) The found attribute name cannot be uniquely matched with the ontology; values however, have been found.

(5) The attribute cannot be uniquely matched, and no consistent values are found.

(6) The attribute name is not known at all in the ontology; possible values are given though.

(7) Neither the attribute name is known, nor have any values been found.

Situation (1) is obviously okay and nothing has to be done. Situation (7) on the other hand cannot be resolved. The only thing what we can do about it is to store this information as a hint to the knowledge engineer that maybe the domain ontology should be revised accordingly, i.e., extended with an additional feature.

For the other cases, we first try to resolve the conflict by simply looking at the attribute-value pairs of all documents in the same cluster, i.e., that describe the same instance. For case (2) with multiple value candidates, this would mean that we look if the content spotters detected unique values in the other documents. If for the majority

of the other documents such a unique value was found, we accept it as the value for this attribute. The same technique is applied for situations (3) to (6).

When the conflicts cannot be resolved that way, because for instance none of the documents in one cluster contains a valid attribute-value pair as defined in the domain ontology, we apply a *fact extraction technique* similar to those described in [7] and [3]. In particular, we again use search engine queries to find out the “most plausible” values for the attributes and in parallel try to learn (and constantly improve) a suitable threshold value for each attribute that determines whether a value should be accepted or not.

The basic procedure can be summarized as follows. First, we take the attribute-value pairs of Situation (1), i.e., those for which we are confident that the values are correct, and construct two search engine queries. Query 1 contains the instance name and the name of the attribute. Query 2, *the discriminator query*, in addition contains the value. From the number of hits returned for each query, we calculate the PMI (Pointwise Mutual Information) measure [21]:  $PMI(q_1, q_2) = Hits(q_1 \cup q_2) / Hits(q_2)$ .

Roughly speaking, we compare the number of all documents that most probably describe the instance to the number of documents that describe this instance but also mention the same value. The basic idea now is to use PMI values, which we calculate for all completely recognized attribute-value pairs from Situation (1), in order to *learn a threshold* (acceptance level) for the more problematic cases. Based on this initial PMI value we are thus able to train a Naive Bayes classifier for each attribute. The classifier, which is used to determine whether a value should be accepted or not, is hence based on the PMI value and the information gained from the analysis of Situation (1) cases.

We then start to try to incrementally resolve the open problems starting with the easier cases from Situation (2) and proceed stepwise to Situation (6). Throughout, we exploit the information we gained from resolving previous cases and also continuously update the threshold values for each attribute.

We will here only sketch the basic idea of the approach using Situation (3) as an example. Let us assume that we have identified the attribute “memory type” but the value “CompactFlash” has been found, which is not consistent with the domain ontology (which may for instance specify that “SecureDigital” and “MemoryStick” are the allowed values). Let us also assume that the problem could not be resolved by looking at the other documents in the cluster. Therefore, we will send the two queries to the search engine as described above. Note that we will use the (inconsistent) value “CompactFlash” in the second query, which for instance leads to a PMI value of 0.002. In the background, however, we know from training the classifier with the assumedly correct attribute-value pairs for “memory type” that the acceptance threshold for this attribute is 0.001. As the newly determined PMI value is higher than the threshold, the system will accept “CompactFlash” as a value for the attribute. In addition, the classifier will be updated and will further on take this new PMI value into account. We apply the same technique - with slight differences - for all Situations (2) to (6).

Of course, adding for instance the value “CompactFlash” to the ontology will introduce an inconsistency in the knowledge base. Note however, that the user (or knowledge engineer) can in such situations use the model-based debugging facility of the ALLRIGHT system [10], which is not only capable of detecting these inconsistencies but also of generating corresponding repair proposals.

### 3 Evaluation

We evaluated our approach in the two popular domains of digital cameras and notebooks, for which we imported domain ontologies from two commercial knowledge-based advisory applications built with ADVISOR SUITE, see <http://bitsuperstore.de>. Both ontologies included definitions of around 40 attributes; an attribute definition comprises the attribute type, keywords, value restrictions and so forth. The ALLRIGHT system took those ontologies as the only input and performed the instantiation process fully automatically. In the data collection step, 3135 observations of digital camera instances from 18 Web sites and 2930 observations of the notebook domain have been extracted as described in Section 2.2. Note that after this step it is not guaranteed that there are no “false positives” in the observations like pages that describe photo printers which share similar attribute names with digital cameras. The digital camera observations contained 85 false positives, the notebook observations 34, which are to be removed in the clustering step. If only one page with an instance description was found it was also considered as a “false positive” since we cannot check the reliability of this data using redundancy.

To evaluate the accuracy of *instance name recognition* (Section 2.3), we analyzed inputs and outputs by hand and compared these manually defined groups with automatically created ones. This also gave us the number existing clusters, i.e., the real number of cameras in the observations. An automatically created cluster was considered correct if it did not contain false positives and there was no other cluster that contained a description of the same instance. For the domain of digital cameras the system created 498 groups, i.e., 498 camera models have been located for which more than one description existed. 234 groups were generated for the domain of notebooks. Note that during the creation of clusters, many “false positives”, for which only one description existed, are filtered out. The clustering and name recognition results (Table 1) are presented in terms of standard information retrieval (IR) metrics: Precision, Recall and F-Measure (F1).

As described in Section 2.4, the clusters of the descriptions allow us to resolve conflicts in the content spotter mark-up. The size of attribute-value sets for the recognition Situations (1 - completely recognized) to (7 - no match) are presented on Fig. 5. It can be seen that the size of the set that corresponds to Situation 7 is rather big. This is mainly because a lot of attributes were not specified in the ontology. In the domain of digital cameras, for instance, some review sites contain over 60 attribute descriptions for their cameras, although only 38 of them were defined in the Web store’s ontology. The same tendency could be observed for notebooks. A big number of observations in the 7th category is a good indication for a user to review the domain ontology and probably add some attribute definitions. Averages of IR metrics were calculated by comparing the results generated by automatic fact extraction with handcrafted ones. If a value was falsely accepted, had a wrong value or was not assigned at all, it was considered as incorrect. The resulting averages of IR measures for fact extraction are presented in Table 2.

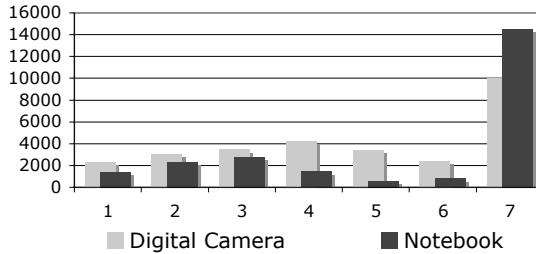
The fact extraction system performed best with the single-valued features like the “Effective pixels” attribute from the Digital Camera domain. The learned threshold value was very effective leading to an average F1-value of 0.813. Even better performance was achieved in the Notebook domain. For the “Processor” attribute, values were extracted with an average F1-value of 0.859, “RAM” with an F1-value of 0.923. The average F1-values for multi-valued attributes like the “ISO Rating” were typically lower

**Table 1.** Clustering and name extraction results

Domain	Precision	Recall	F-Measure
Digital Camera	0.964	0.965	0.964
Notebook	0.943	0.918	0.93

**Table 2.** Fact Extraction: Average Precision, Recall and F-Measure for two evaluation domains

Domain	Precision	Recall	F-Measure
Digital Camera	0,738	0,91	0,815
Notebook	0,878	0,9275	0,902

**Fig. 5.** Recognition situations: number of attribute-value pairs after cluster-based resolution

(average F1: 0.766), mostly because the number of found values for such attributes varies from instance to instance. So, if for instance the average number of values in training data was smaller than the actual average in the whole data set, the threshold will be very restrictive.

The overall ALLRIGHT instantiation accuracy was assessed using *analysis of variance* (ANOVA) in order to reduce the amount of manual validation: Instead of reviewing nearly 19,000 attribute values extracted for the digital camera domain we reviewed only 946, which is a sufficient sample size according to the Cochran criteria with the level of acceptable error at 1%. For each of the attributes we thus created a random sample of extracted attribute-value pairs.

In order to calculate Precision, we used these samples to estimate the number of correct values and divided it by the number of all values that were found by the ALLRIGHT system. Recall was calculated as a ratio of the estimated number of correct values to the number of all ontology instances found by ALLRIGHT. This was basically done under assumption that the general goal is to instantiate all specified attributes. Also note that the final results are also influenced by “empty” attributes. Since we imported the domain ontology from an existing application, there were some out-of-date attribute definitions, which were never found on the Web pages. These empty values were considered as faulty in our evaluation strategy thus decreasing the Recall.

The obtained IR measures are shown in Table 3. Overall, over 77% of all instances of the digital camera ontology and over 85% of the notebook ontology were correctly instantiated. We view these results to be highly promising, in particular as we started

**Table 3.** ALLRIGHT attribute-value extraction results

Domain	Precision	Recall	F-Measure
Digital Camera	0,819	0,74	0,777
Notebook	0,834	0,883	0,858

with a very small amount of seed knowledge and only relied on domain-independent heuristics and metrics, i.e., for instance no manually defined text patterns for content spotting and so forth have been used.

## 4 Summary and Outlook

In this paper we have shown how the process of populating an ontology for *knowledge-based systems* with instance data can be automatized by retrieving and extracting the corresponding information from Web sources. The presented ALLRIGHT system relies on a novel combination of new and existing techniques from different areas and is in particular optimized for dealing with information which is contained in tabular form. The evaluation of the approach in two popular domains showed that highly precise instance information can be retrieved even if only a small amount of seed knowledge is available. We thus see our work as a contribution toward the broader application of ontology- or semantics-based applications whose adoption is in many cases hampered by the fact that larger amounts of ontology instance data have to be manually entered to the system.

We currently aim to extend and further improve the ALLRIGHT system in several dimensions. On the technical level, we evaluate additional crawling techniques and heuristics that go beyond the “index-page” approach and which can also better handle dynamic HTML pages. Also, the generated content spotters will be extended such that they are able to *learn* and thus improve their accuracy over time.

Beside these technical improvements, we are currently evaluating the results of the *relationship extraction* module, with which we for instance aim at detecting which kinds of accessories (like memory cards) are compatible with a certain camera model. The first evaluations, which are based on an ontology alignment technique and on similar fact extraction methods like the ones described herein, are already very promising but go beyond the scope of this paper.

## Acknowledgments

We would like to thank the reviewers for their valuable comments. The research project is funded partly by the grant from the Austrian Research Promotion Agency, Programm Line FIT-IT Semantic Systems (Project AllRight, Contract 809261) and the Austrian Science Fund FWF (Project V-Know, Contract 19996).

## References

1. Alani, H., Kim, S., Milard, D.E., Weal, M.J., Hall, W., Lewis, P.H., Shadbolt, N.R.: Automatic ontology-based knowledge extraction from web documents. *IEEE Intelligent Systems* 18, 14–21 (2003)
2. Chakrabarti, S., van den Berg, M., Dom, B.: Focused crawling: a new approach to topic-specific web resource discovery. *Comput. Networks* 31(11-16), 1623–1640 (1999)
3. Cimiano, P., Ladwig, G., Staab, S.: Gimme' the context: context-driven automatic semantic annotation with c-pankow. In: *Proceedings of the 14th international conference on World Wide Web*, pp. 332–341 (2005)

4. Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., Slattery, S.: Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence* 118(1), 69–113 (2000)
5. Dill, S., Tomlin, J., Zien, J., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., Rajagopalan, S., et al.: SemTag and seeker: bootstrapping the semantic web via automated semantic annotation. In: *Proceedings of the Twelfth International Conference on World Wide Web*, pp. 178–186 (2003)
6. Embley, D.W., Campbell, D.M., Jiang, Y.S., Liddle, S.W., Ng, Y.-K., Quass, D., Smith, R.D.: Conceptual-model-based data extraction from multiple-record web pages. *Data Knowledge Engineering* 31(3), 227–251 (1999)
7. Etzioni, O., Cafarella, M., Downey, D., Popescu, A., Shaked, T., Soderland, S., Weld, D., Yates, A.: Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence* 165(1), 91–134 (2005)
8. Felfernig, A., Friedrich, G., Jannach, D., Zanker, M.: An integrated environment for the development of knowledge-based recommender applications. *International Journal of Electronic Commerce* 11(2), 2006–2007 (2007)
9. Felfernig, A., Friedrich, G., Schmidt-Tieme, L.: Recommender systems. *IEEE Intelligent Systems - Special Issue on Recommender Systems* 22(3) (May 2007)
10. Friedrich, G., Shchekotykhin, K.: A General Diagnosis Method for Ontologies. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005. LNCS*, vol. 3729, pp. 232–246. Springer, Heidelberg (2005)
11. Friedrich, G., Shchekotykhin, K.: NameIt: Extraction of product names. In: Perner, P. (ed.) *ICDM 2006. LNCS (LNAD)*, vol. 4065, pp. 29–33. Springer, Heidelberg (2006)
12. Ipeirotis, P.G., Agichtein, E., Jain, P., Gravano, L.: To search or to crawl?: towards a query optimizer for text-centric tasks. In: *SIGMOD 2006. Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pp. 265–276. ACM Press, New York (2006)
13. Jayram, T., Krishnamurthy, R., Raghavan, S., Vaithyanathan, S., Zhu, H.: Avatar information extraction system. *IEEE Data Engineering Bulletin* 29(1), 40–48 (2006)
14. Kleinberg, J.: Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)* 46(5), 604–632 (1999)
15. Krüpl, B., Herzog, M., Gatterbauer, W.: Visually guided bottom-up table detection and segmentation in web documents. In: *WWW 2006. The 15th International World Wide Web Conference* (2006)
16. Naumann, F., Bilke, A., Bleiholder, J., Weis, M.: Data fusion in three steps: Resolving schema, tuple, and value inconsistencies. *IEEE Data Eng. Bull* 29(2), 21–31 (2006)
17. Pan, J., Yang, H., Faloutsos, C., Duygulu, P.: Automatic multimedia cross-modal correlation discovery. In: *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 653–658. ACM Press, New York (2004)
18. Pelleg, D., Moore, A.W.: X-means: Extending k-means with efficient estimation of the number of clusters. In: *Seventeenth International Conference on Machine Learning*, pp. 727–734. Morgan Kaufmann, San Francisco (2000)
19. Petasis, G., Karkaletsis, V., Spyropoulos, C.: Cross-lingual information extraction from web pages: the use of a general-purpose text engineering platform. In: *4th International Conference on Recent Advances in Natural Language Processing (RANLP)*, Borovets, Bulgaria (2003)
20. Popescu, A., Etzioni, O.: Extracting product features and opinions from reviews. In: *Proceedings of EMNLP 2005* (2005)
21. Turney, P., et al.: Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 417–424 (2002)



# A Caching Mechanism for Semantic Web Service Discovery

Michael Stollberg, Martin Hepp, and Jörg Hoffmann

Digital Enterprise Research Institute (DERI),  
University of Innsbruck, Austria  
firstname.lastname@deri.org

**Abstract.** The discovery of suitable Web services for a given task is one of the central operations in Service-oriented Architectures (SOA), and research on Semantic Web services (SWS) aims at automating this step. For the large amount of available Web services that can be expected in real-world settings, the computational costs of automated discovery based on semantic matchmaking become important. To make a discovery engine a reliable software component, we must thus aim at minimizing both the mean and the variance of the duration of the discovery task. For this, we present an extension for discovery engines in SWS environments that exploits structural knowledge and previous discovery results for reducing the search space of consequent discovery operations. Our prototype implementation shows significant improvements when applied to the Stanford SWS Challenge scenario and dataset.

## 1 Introduction

Web service discovery is the process of finding suitable Web services for a given task, denoting one of the central operations in Service-oriented Architectures (SOA). There is already substantial work in the field Semantic Web services (SWS) on automating Web service discovery by semantic matchmaking, mostly focussing on the retrieval performance (e.g. [14, 11, 13, 9]).

However, the computational performance of semantically enabled discovery and the practical consequences for SWS environments have not received a lot of attention so far. Considering the increasing amount of available Web services, this becomes in particular relevant for employing a discovery engine as a heavily used component in systems for dynamic Web service composition or semantically enabled business process management (e.g. [19, 8]). For this, we consider the following characteristics for judging the computational reliability of a discovery engine: *efficiency* as the time required for finding a suitable Web service, *scalability* as the ability to deal with a large search space of available Web services, and *stability* as a low variance of the execution time of several invocations.

This paper presents a technique that addresses this challenge by adapting the concept of caching to Web service discovery. It captures knowledge on discovery results for generic descriptions of objectives to be achieved, and exploits this for optimizing Web service discovery for concrete requests at runtime.



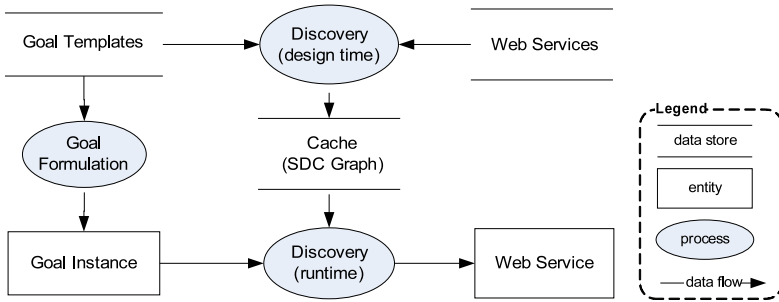


Fig. 1. Overview of Web Service Discovery Framework

We refer to this as *Semantic Discovery Caching* (SDC). Figure 1 shows the basic idea as a dataflow diagram. There are three core conceptual entities: *Web services* that have a formal description, *goal templates* as generic objective descriptions that are stored in the system, and *goal instances* that describe concrete requests by instantiating a goal template with concrete inputs. At design time, Web services for goal templates are discovered. The result is captured in a special knowledge structure called the *SDC graph*. At runtime, a concrete client request is formulated as a goal instance for which suitable Web services need to be discovered. As the expectably most frequent operation in SOA applications, we optimize this by exploiting the SDC graph in order to reduce the search space and minimize the number of necessary matchmaking operations.

The paper is structured as follows. Section 2 briefly describes our approach for semantically enabled Web service discovery as presented in earlier works [18,17]. Then, Section 3 specifies the Semantic Discovery Caching technique. Section 4 presents the evaluation, and discusses the relevance and specifics of our approach. Section 5 positions it within related work, and Section 6 concludes the paper. We use the shipment scenario from the Stanford SWS Challenge as a running example, a widely recognized initiative for demonstration and comparison of semantically enabled discovery techniques (<http://www.sws-challenge.org>). A detailed technical report on this work is provided in [16].

## 2 Foundations of the Discovery Framework

As the basis for the subsequent sections, the following summarizes our approach for semantically enabled Web service discovery as presented in earlier works.

We follow the goal-driven approach for Semantic Web services as promoted by the WSMO framework [5]. In contrast to an invocation request for a Web service, a goal describes a client objective with respect to the problem that shall be solved while abstracting from technical details on Web service invocation. The aim is to facilitate problem-oriented Web service usage: the client merely specifies the objective to be achieved as a goal, and the system automatically discovers, composes, and executes suitable Web services for solving this [18].

The distinction of goal templates and goal instances allows to ease the goal formulation by clients, and it facilitates the two-phase Web service discovery as outlined above. For this, we have defined a formal model that considers a state-based model of the world that Web services operate in, and provides precise definitions of goals, Web services, and the necessary matchmaking techniques. We here recall the central aspects; the full model is defined in [17].

## 2.1 Web Services, Goals, and Functional Descriptions

We consider functional aspects as the primary aspect for discovery: if a Web service does not provide the functionality for solving a goal, then it is not usable and other, non-functional aspects are irrelevant. The relevant parts of goal and Web service descriptions for discovery by semantic matchmaking are the formally described requested and the provided functionalities.

In our state-based model, a particular execution of a Web service  $W$  denotes a sequence of states  $\tau = (s_0, \dots, s_m)$ , i.e. a change of the world from a start state  $s_0$  to an end state  $s_m$ . In consequence, the overall functionality provided by  $W$  is the set of all possible executions of  $W$ , denoted by  $\{\tau\}_W$ . Analogously, we understand a particular solution of a goal as a sequence of states from the initial state into a state of the world wherein the objective is solved. A functional description  $\mathcal{D}$  formally describes the possible executions of a Web service – respectively the possible solutions for a goal – with respect to the start- and end states.

We define  $\mathcal{D}$  over a signature  $\Sigma$ , and use ontologies  $\Omega$  as the background knowledge.  $\mathcal{D}$  consists of a set of input variables  $IN = \{i_1, \dots, i_n\}$ , a precondition  $\phi^{pre}$  that constrains the possible start states, and an effect  $\phi^{eff}$  that constrains the possible end states. To properly describe the dependency of the start- and end states,  $IN$  occur as free variables in both  $\phi^{pre}$  and  $\phi^{eff}$ ; the predicate *out* denotes the outputs. The formal meaning of  $\mathcal{D}$  is defined as an *implication semantics* between the precondition and the effect. We say that a Web service  $W$  provides the functionality described by  $\mathcal{D}$ , denoted by  $W \models \mathcal{D}$ , if and only if for all  $\tau \in \{\tau\}_W$  holds that if  $s_0 \models \phi^{pre}$  then  $s_m \models \phi^{eff}$ . In order to deal with functional descriptions in terms of model-theoretic semantics, we present this as a FOL formula  $\phi^{\mathcal{D}}$  of the form  $\phi^{pre} \Rightarrow \phi^{eff}$ . Then,  $W \models \mathcal{D}$  is given if and only if every  $\tau \in \{\tau\}_W$  is represented by a  $\Sigma$ -interpretation that is a model of  $\phi^{\mathcal{D}}$ .

Analogously, the functional description  $\mathcal{D}_G$  of a goal template  $G$  formally describes the set  $\{\tau\}_G$  as the state sequences that are possible solutions for  $G$ . Goal templates are generic objective descriptions that are kept in the system. At runtime, a concrete client request is formulated as a goal instance that instantiates a goal template with concrete inputs. We define a goal instance as a pair  $GI(G) = (G, \beta)$  with  $G$  as the corresponding goal template, and an input binding  $\beta : \{i_1, \dots, i_n\} \rightarrow \mathcal{U}$  as a total function that assigns objects of  $\mathcal{U}$  to each  $IN$ -variable of  $\mathcal{D}_G$ . This  $\beta$  is subsequently used to invoke a Web service in order to solve  $GI(G)$ . We say that  $GI(G)$  is a consistent instantiation of its corresponding goal template, denoted by  $GI(G) \models G$ , if  $\phi^{\mathcal{D}_G}$  is satisfiable under the input binding  $\beta$ . A usable Web service for  $GI(G)$  can only be found if this is given. Moreover, it holds that if  $GI(G) \models G$  then  $\{\tau\}_{GI(G)} \subseteq \{\tau\}_G$ .

**Table 1.** Examples for Functional Descriptions of Goals and Web Services

Goal Template $G$ “ship a package of any weight in Europe”	Web Service $W$ “shipment in Germany, max 50 kg”
$\Omega$ : location & shipment ontology $IN$ : $\{?s, ?r, ?p, ?w\}$ $\phi^{pre}$ : $person(?s) \wedge in(?s, europe)$ $\wedge person(?r) \wedge in(?r, europe)$ $\wedge package(?p) \wedge weight(?p, ?w)$ $\wedge maxWeight(?w, heavy)$ . $\phi^{eff}$ : $\forall ?o, ?price. out(?o) \Leftrightarrow ($ $shipmentOrder(?o, ?p)$ $\wedge sender(?p, ?s) \wedge receiver(?p, ?r)$ $\wedge costs(?o, ?price) )$ .	$\Omega$ : location & shipment ontology $IN$ : $\{?s, ?r, ?p, ?w\}$ $\phi^{pre}$ : $person(?s) \wedge in(?s, germany)$ $\wedge person(?r) \wedge in(?r, germany)$ $\wedge package(?p) \wedge weight(?p, ?w)$ $\wedge maxWeight(?w, 50)$ . $\phi^{eff}$ : $\forall ?o, ?price. out(?o) \Leftrightarrow ($ $shipmentOrder(?o, ?p)$ $\wedge sender(?p, ?s) \wedge receiver(?p, ?r)$ $\wedge costs(?o, ?price) )$ .

Table 1 shows examples for functional descriptions in our running example. Using classical first-order logic as the specification language, the preconditions define conditions on the required inputs, and the effects state that the output is a shipment order with respect to the input values.<sup>1</sup>

## 2.2 Semantic Matchmaking

The matchmaking techniques for Web service discovery are defined on the basis of the functional descriptions explained above. We consider a Web service  $W$  to be usable for a goal template  $G$  if there exists at least one execution of  $W$  that is also a solution for  $G$ , i.e. if  $\exists \tau. \tau \in \{\tau\}_G \cap \{\tau\}_W$ . We express the usability of  $W$  for  $G$  in terms of matching degrees as defined in Table 2. Four degrees – *exact*, *plugin*, *subsume*, *intersect* – denote different situations wherein  $W$  is usable for solving  $G$ ; the *disjoint* degree denotes that this is not given. We always use the highest possible degree as it holds that (1) *plugin*  $\wedge$  *subsume*  $\Leftrightarrow$  *exact*, (2) *plugin*  $\Rightarrow$  *intersect*, (3) *subsume*  $\Rightarrow$  *intersect*, and (4)  $\neg$  *intersect*  $\Leftrightarrow$  *disjoint*.

Analogously, we consider a Web service  $W$  to be usable for a goal instance  $GI(G)$  if  $W$  can provide a solution for  $GI(G)$  when it is invoked with the input binding  $\beta$  defined in  $GI(G)$ . Formally, this is given if union of the formulae  $\Omega \cup \{[\phi^D_G]_\beta, [\phi^D_W]_\beta\}$  is satisfiable. This means that under consideration of the domain knowledge  $\Omega$  and under the input binding  $\beta$  defined in  $GI(G)$  there must be a  $\Sigma$ -interpretation that represents a solution for the corresponding goal template  $G$  as well as a possible execution of the Web service  $W$ . However, we

<sup>1</sup> (a) We consider Web services to provide *deterministic functionalities*, i.e. that the end-state of an execution is completely dependent on the start-state and the provided inputs; this is a pre-requisite for precise discovery by semantic matchmaking. (b) We consider all functional descriptions  $\mathcal{D}$  to be *consistent*, i.e. that  $\phi^D$  is satisfiable under an input binding  $\beta$ . Otherwise, a Web service  $W \models \mathcal{D}$  would not be realizable, and there would not be any solution for a goal. The full model further considers dynamic symbols that are changed during executions.

**Table 2.** Definition of Matching Degrees for  $\mathcal{D}_G$ ,  $\mathcal{D}_W$ 

Denotation	Definition	Meaning
$\mathbf{exact}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \forall \beta. \phi^{\mathcal{D}_G} \Leftrightarrow \phi^{\mathcal{D}_W}$	$\tau \in \{\tau\}_G$ if and only if $\tau \in \{\tau\}_W$
$\mathbf{plugin}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \forall \beta. \phi^{\mathcal{D}_G} \Rightarrow \phi^{\mathcal{D}_W}$	if $\tau \in \{\tau\}_G$ then $\tau \in \{\tau\}_W$
$\mathbf{subsume}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \forall \beta. \phi^{\mathcal{D}_G} \Leftarrow \phi^{\mathcal{D}_W}$	if $\tau \in \{\tau\}_W$ then $\tau \in \{\tau\}_G$
$\mathbf{intersect}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \exists \beta. \phi^{\mathcal{D}_G} \wedge \phi^{\mathcal{D}_W}$	there is a $\tau$ such that $\tau \in \{\tau\}_G$ and $\tau \in \{\tau\}_W$
$\mathbf{disjoint}(\mathcal{D}_G, \mathcal{D}_W)$	$\Omega \models \neg \exists \beta. \phi^{\mathcal{D}_G} \wedge \phi^{\mathcal{D}_W}$	there is no $\tau$ such that $\tau \in \{\tau\}_G$ and $\tau \in \{\tau\}_W$

can simplify the determination of the usability of  $W$  for  $GI(G)$  on the basis of the usability degree of  $W$  for the corresponding goal template  $G$  as follows.

**Definition 1.** Let  $GI(G) = (G, \beta)$  be a goal instance with  $GI(G) \models G$ . Let  $W$  be a Web service, and let  $\mathcal{D}_W$  be a functional description such that  $W \models \mathcal{D}_W$ .

$W$  is usable for solving  $GI(G)$  if and only if:

- (i)  $\mathbf{exact}(\mathcal{D}_G, \mathcal{D}_W)$  or
- (ii)  $\mathbf{plugin}(\mathcal{D}_G, \mathcal{D}_W)$  or
- (iii)  $\mathbf{subsume}(\mathcal{D}_G, \mathcal{D}_W)$  and  $\Omega \wedge [\phi^{\mathcal{D}_W}]_\beta$  is satisfiable, or
- (iv)  $\mathbf{intersect}(\mathcal{D}_G, \mathcal{D}_W)$  and  $\Omega \wedge [\phi^{\mathcal{D}_G}]_\beta \wedge [\phi^{\mathcal{D}_W}]_\beta$  is satisfiable.

This states that only those Web services that are usable for the corresponding goal template  $G$  are potentially usable for the goal instance  $GI(G)$ . If a Web service  $W$  is usable for  $G$  under the *exact* or *plugin* degree, then it is also usable for any goal instance of  $G$  because  $\{\tau\}_{GI(G)} \subseteq \{\tau\}_G \subseteq \{\tau\}_W$ . Under the *subsume* degree, all executions of  $W$  are solutions of  $G$  but not vice versa. Table 2 above is an example for this. Consider a goal instance that defines  $\beta = \{?s|paris, ?r|vienna, ?p|aPackage, ?w|3.1\}$ : although this properly instantiates the goal template,  $\beta$  does not allow to invoke the Web service (which is restricted to Germany); thus, it is not usable here. Under the *intersect* degree, the complete matching condition explained above must be checked at runtime.

### 3 Semantic Discovery Caching

We now turn towards the caching mechanism for Web service discovery. Working on the formal model explained above, the aim is to improve the computational reliability of Web service discovery for goal instances that is performed at runtime. We commence with the design principles, then provide the formal definition, and finally explain the optimization for the runtime discovery process.

#### 3.1 Overview

The idea is to reduce the search space and minimize the necessary matchmaking operations for Web service discovery by exploiting the formal relationships

between goal templates, goal instances, and Web services. The central element for this is the SDC graph that organizes goal templates in a subsumption hierarchy with respect to their semantic similarity, and captures the minimal knowledge on the functional usability of the available Web services for the goal templates.

Two goal templates  $G_i$  and  $G_j$  are considered to be similar if they have at least one common solution. Then, mostly the same Web services are usable for them. We express this in terms of *similarity degrees*  $d(G_i, G_j)$  that denote the matching degree between the functional descriptions  $\mathcal{D}_{G_i}$  and  $\mathcal{D}_{G_j}$ . Formally, these degrees are defined analog to Table 2 (cf. Section 2.2). In order to enable efficient search, we define the SDC graph such that the only occurring similarity degree is *subsume*( $G_i, G_j$ ). If this is given, then (1) the solutions for the child  $G_j$  are a subset of those for the parent  $G_i$ , and thus (2) the Web services that are usable for  $G_j$  are a subset of those usable for  $G_i$ .

In consequence, the SDC graph is a directed acyclic graph that consists of two layers. The upper one is the *goal graph* that defines the subsumption hierarchy of goal templates by directed arcs. The lower layer is the *usability cache* that explicates the usability of each available Web service  $W$  for every goal template  $G$  by directed arcs that are annotated with the usability degree  $d(G, W)$ . The discovery operations use this knowledge structure by inference rules of the form  $d(G_i, G_j) \wedge d(G_i, W) \Rightarrow d(G_j, W)$  that result from the formal definitions.

Figure 2 illustrates the SDC graph for our running example along with the most relevant inference rules. There are three goal templates:  $G_1$  for package shipment in Europe,  $G_2$  for Switzerland, and  $G_3$  for Germany. Their similarity degrees are *subsume*( $G_1, G_2$ ) and *subsume*( $G_1, G_3$ ), which is explicated in the goal graph. Consider some Web services, e.g.  $W_1$  for package shipment in Europe,  $W_2$  in the whole world,  $W_3$  in the European Union, and  $W_4$  in the Commonwealth. Their usability degree for each goal template is explicated in the usability cache, whereby redundant arcs are omitted. We shall explain the creation of the SDC graph as well as its usage for optimizing the discovery process below.

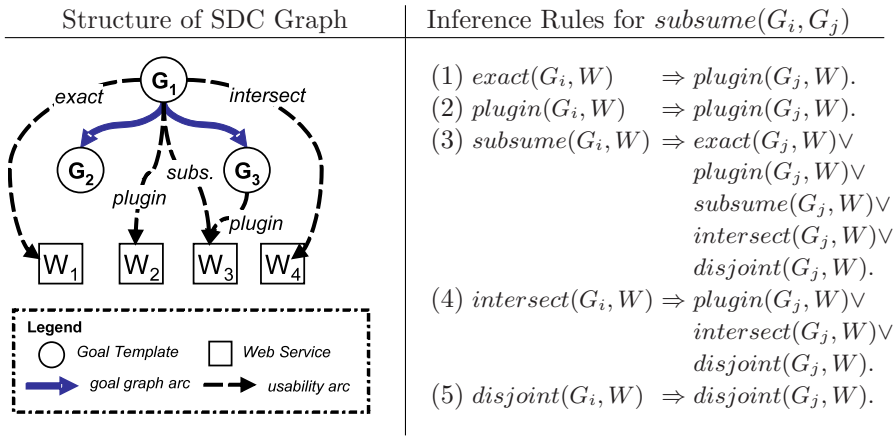


Fig. 2. Example of a SDC Graph and Inference Rules

### 3.2 Definition

The following provides the formal definition of the SDC graph and explains the algorithms for ensuring that the properties are maintained at all times.

**Definition 2.** Let  $d(G_i, G_j)$  denote the similarity degree of goal templates  $G_i$  and  $G_j$ , and let  $d(G, W)$  denote the usability degree of a Web service  $W$  for a goal template  $G$ . Given a set  $\mathcal{G}$  of goal templates and a set  $\mathcal{W}$  of Web services, the SDC graph is a directed acyclic graph  $(V_{\mathcal{G}} \cup V_{\mathcal{W}}, E_{sim} \cup E_{use})$  such that:

- (i)  $V_{\mathcal{G}} := \mathcal{G} \cup \mathcal{G}^I$  is the set of inner vertices where:
  - $\mathcal{G} = \{G_1, \dots, G_n\}$  are the goal templates; and
  - $\mathcal{G}^I := \{G^I \mid G_i, G_j \in \mathcal{G}, d(G_i, G_j) = \textit{intersect}, G^I = G_i \cap G_j\}$  is the set of intersected goal templates from  $\mathcal{G}$
- (ii)  $V_{\mathcal{W}} := \{W_1, \dots, W_m\}$  is the set of leaf vertices representing Web services
- (iii)  $E_{sim} := \{(G_i, G_j) \mid G_i, G_j \in V_{\mathcal{G}}\}$  is the set of directed arcs where:
  - $d(G_i, G_j) = \textit{subsume}$ ; and
  - not exists  $G \in V_{\mathcal{G}}$  s.t.  $d(G_i, G) = \textit{subsume}$ ,  $d(G, G_j) = \textit{subsume}$ .
- (iv)  $E_{use} := \{(G, W) \mid G \in V_{\mathcal{G}}, W \in V_{\mathcal{W}}\}$  is set of directed arcs where:
  - $d(G, W) \in \{\textit{exact}, \textit{plugin}, \textit{subsume}, \textit{intersect}\}$ ; and
  - not exists  $G_i \in V_{\mathcal{G}}$  s.t.  $d(G_i, G) = \textit{subsume}$ ,  $d(G_i, W) \in \{\textit{exact}, \textit{plugin}\}$ .

This defines the structure of a SDC graph as outlined above. Two refinements are necessary to obtain this from an initial set of goal templates and Web services.

The first one ensures that the only similarity degree that occurs in the SDC graph is  $\textit{subsume}(G_i, G_j)$ , cf. clause (iii). This denotes that  $G_j$  is a functional specialization of  $G_i$  such that  $\{\tau\}_{G_j} \subset \{\tau\}_{G_i}$ . In consequence, only those Web services that are usable for  $G_i$  can be usable for  $G_j$  because if  $\{\tau\}_{G_i} \cap \{\tau\}_W = \emptyset$  then also  $\{\tau\}_{G_j} \cap \{\tau\}_W = \emptyset$ , cf. rule (5) in Figure 2. With this as its constituting element, the SDC graph provides an index structure for efficient search of goal templates and Web services as explained above. The other possible similarity degrees are handled as follows: if  $\textit{exact}(G_i, G_j)$ , only one goal template is kept while the other one is redundant; if  $\textit{plugin}(G_i, G_j)$  then we store the opponent arc  $(G_j, G_i)$ . If  $\textit{disjoint}(G_i, G_j)$ , then both are kept as disconnected nodes in the SDC graph. Effectively, each of its connected subgraphs covers a problem domain, e.g. one for the shipment scenario and another one for flight ticketing.

The only critical similarity degree is  $\textit{intersect}(G_i, G_j)$ , denoting that  $G_i$  and  $G_j$  have a common solution but there are also exclusive solutions for each. This can cause cycles in the SDC graph which hamper its search properties. To avoid this, we create an *intersection goal template*  $G^I(G_i, G_j)$  whose solutions are exactly those that are common to  $G_i$  and  $G_j$ , cf. clause (i). Formally,  $G^I$  is defined as the conjunction of the functional descriptions of the original goal templates, i.e.  $\phi^{D_{G^I(G_i, G_j)}} = \phi^{D_{G_i}} \wedge \phi^{D_{G_j}}$  so that  $\{\tau\}_{G^I(G_i, G_j)} = \{\tau\}_{G_i} \cap \{\tau\}_{G_j}$ . Because of this, it holds that  $\textit{subsume}(G_i, G^I(G_i, G_j))$  and  $\textit{subsume}(G_j, G^I(G_i, G_j))$ . Thus,  $G^I$  becomes a child node of both  $G_i$  and  $G_j$  in the goal graph. This is applied for every occurring *intersect* similarity degree so that eventually all similar goal templates are organized in a subsumption hierarchy and no cycles occur in

the SDC graph. Intersection goal templates are only used as logical constructs; their functional descriptions do not have to be materialized.

The second refinement ensures the minimality of the usability cache, *cf.* clause (iv). For optimizing the discovery operations, we must know the usability degree of every Web service for each goal template. However, in order to avoid redundancy, we omit arcs for which the precise usability degree can be inferred from the SDC graph. It holds that if  $\text{subsume}(G_i, G_j)$ , then the usability degree of a Web service  $W$  for the child  $G_j$  is always *plugin* if  $W$  is usable for the parent  $G_i$  under the degrees *exact* or *plugin* because  $\{\tau\}_W \supseteq \{\tau\}_{G_i} \supset \{\tau\}_{G_j}$ . Thus, the arc  $(G_j, W)$  is not explicated in the SDC graph. In the above example, the Web services  $W_1$  and  $W_2$  are usable under the *plugin* degree for both  $G_2$  and  $G_3$ ; this can be inferred from the usability cache arcs of  $G_1$  (*cf.* rules (1) and (2) in Figure 2). Therewith,  $E_{use}$  is the minimal set of arcs that are necessary to explicate the usability degrees of the available Web services for each goal template.

In our implementation, the creation of a SDC graph is realized by the subsequent addition of goal templates. Applying the refinements explained above, a new goal template is first inserted at the right position in the goal graph and then the usability cache is created for it. The removal or modification of goal templates are manual maintenance operations; respective algorithms ensure that the properties of the SDC graph are maintained. Analogous algorithms are provided for the addition, removal, and modification of Web services. These are automatically triggered by changes in the Web service repository.<sup>2</sup>

### 3.3 Runtime Discovery Optimization

We now explain the usage of the SDC graph for optimizing the runtime discovery process, i.e. for finding a Web service that is usable for solving a goal instance. We consider this as the most frequent operation in real-world SOA applications, while changes on goal templates and Web services are significantly less frequent maintenance operations. The optimization is achieved by (1) *reducing the search space* as only the Web services that are usable the corresponding goal template need to be inspected, and (2) *minimizing the number of necessary matchmaking operations* by first inspecting Web services for which no matchmaking is required at runtime. Listing 1 illustrates the algorithm for this.

---

```

input: GI(G);
if ( ! consistentInstantiation (GI(G)) ) then fail ;
if ( lookup(G) ) then return W;
while ( subsume(G,G') and consistentInstantiation (GI(G')) ) do {
  replace (G,G');
  if ( lookup(G') ) then return W; }
if ( otherWS(G) ) then return W;
else fail ;

```

---

**Listing 1.** Algorithm for SDC-enabled Runtime Discovery

<sup>2</sup> The SDC prototype is open source software available from the SDC homepage at [members.deri.at/~michaels/software/sdc/](http://members.deri.at/~michaels/software/sdc/). It is realized as a discovery component in the WSMX system (the WSMO reference implementation, [www.wsmx.org](http://www.wsmx.org)). We use VAMPIRE for matchmaking, a FOL automated theorem prover.



The input is a goal instance  $GI(G) = (G, \beta)$  for which a usable Web service shall be found. At first, we need to ensure that this is a consistent instantiation of its corresponding goal template  $G$ ; if this is not given, a usable Web service can not be found. Then, we try to find a usable Web service by *lookup*. This searches in the SDC graph for a Web service  $W$  that is usable for  $G$  under the *exact* or the *plugin* degree; this  $W$  is usable for solving  $GI(G)$  without the need of matchmaking at runtime (*cf.* Definition II, Section 2.2). If this is successful,  $W$  is returned and the discovery is completed successfully.

Otherwise, we continue with refining the goal instance in order to reduce the search space. For this, we successively replace the corresponding goal template  $G$  by the child node  $G'$  for which the goal instance still is a consistent instantiation. In the example from Figure 2, let  $GI(G_1)$  be a goal instance for shipping a package from Munich to Berlin that instantiates  $G_1$  for package shipment within Europe. This is also a proper instantiation of  $G_3$  for shipment within Germany; hence, we refine the goal instance to  $GI(G_3)$ . In the SDC Graph, all children of  $G$  are disjoint – those for which there is no intersection goal template – so that there can only be one  $G'$  with  $subsume(G, G')$  and  $GI(G) \models G'$ . If there is an intersection goal template  $G^I$  and  $GI(G) \models G^I$ , this is found by following the path via either of its parents. We thus can search downwards in the goal graph until finding the lowest possible  $G'$ : for this, the number of usable Web services is minimal. In each refinement step we invoke the *lookup* procedure because the probability of success is the higher the lower  $G'$  is allocated in the goal graph.

As the last option for finding a usable Web service, we inspect those ones that are usable for the (possibly refined) corresponding goal template under the degrees *subsume* and *intersect*; this requires matchmaking at runtime (*cf.* Definition II). As soon as a usable Web service is detected, it is returned as the discovery result. Otherwise, a Web service for solving  $GI(G)$  does not exist.

This algorithm finds one Web service that is usable for a goal instance under functional aspects. Of course other aspects are also relevant for discovery. Most prominent in literature are *selection* and *ranking* techniques: the former reduces the set of suitable Web services with respect to quality-of-service criteria (e.g. [20]); the latter provides a preference order for the Web services (e.g. [12]). Under the reasonable premise of performing selection and ranking *after* functional discovery, the SDC technique provides a sufficient optimization technique for integrated discovery engines: the bottleneck is the number of available Web services, which is only relevant for the first processing step. The above algorithm can easily be modified to return the set of all functionally usable Web services.

## 4 Evaluation

In order to evaluate the performance gain achievable with the SDC technique, we have compared the our prototype implementation with a not-optimized discovery engine for goal instances that applies the same matchmaking techniques. The following explains the test set-up and methodology, summarizes the results, and discusses the impact of our observations.



## 4.1 Methodology

The aim of this evaluation is to quantify the effect of the SDC technique on the duration of realistic discovery tasks over larger sets of available Web services that can be expected in real-world settings. We therefore compare the SDC-enabled runtime discovery with a naive discovery engine for goal instances. We will discuss the relationship to other optimization techniques in Section 5.

For the comparison, we use the original data set from the Stanford SWS challenge shipment scenario that already served as the running example above. Based on real-world services, this challenge defines five Web services for package shipment from the USA to different destination countries, and several examples of client requests. We map this to our framework such that goal templates are generic objective descriptions for package shipment, and the individual requests are described as goal instances. The formal functional descriptions of goals and Web services are analog to Table 1, cf. Section 2.1. The root goal template of the resulting SDC graph describes the objective of shipping a package of any weight from and to anywhere in the world. The more detailed levels of the goal graph are concerned with shipment between continents, the next levels between countries, and the lowest levels differentiate the weight classes. At the top of the goal graph, the most common usability degree for the Web services is *subsume*; this changes to *plugin* at the lower levels.

On this basis, we define ten goal instances for which a usable Web service is to be found. These are modeled such that each part of the SDC-enabled runtime discovery algorithm is covered (cf. Section 3.3). The comparison engine is a naive runtime discoverer that does not apply any optimization techniques. It retrieves the available Web services in a random order, and performs the basic matchmaking to determine their usability for a goal instance as defined in Section 2.2. It uses the same matchmaking techniques and infrastructure as the SDC-enabled engine. For comparing the behavior of the engines, we perform Web service discovery for each goal instance with different numbers of available Web services. Among these are always the five Web services defined in the scenario that are potentially usable for the tested goal instance; all others are not.

## 4.2 Results

For the analysis, each comparison test has been run 50 times and the results are prepared in the following statistical standard notations: the *arithmetic mean*  $\mu$  as the average value, the *median*  $\bar{x}$  that denotes the value in the middle of the 50 test runs, and the standard deviation  $\sigma$  as a measurement for the value spread among the test runs. Table 3 shows a fragment of the augmented data of all test runs for all ten goal instances; the original data is provided in [16].<sup>3</sup>

From this we can observe the following differences between the compared engines with respect to the three quality criteria for reliability: the SDC-enabled

<sup>3</sup> The comparison has been performed as a JUnit test for Java 5.0 in Eclipse 3.2; the test machine was a standard laptop with a 2 GHz Intel processor and 1 GB of RAM. For this use case, the average time for a single matchmaking operation has been 115 msec, and 15 msec for the discovery-by-lookup procedure.

**Table 3.** Comparison Test Statistics (all values in seconds)

no. of WS	Engine	Mean $\mu$	Median $\bar{x}$	Standard Deviation $\sigma$
10	SDC	0.28	0.27	0.03 (11.74 %)
	naive	0.41	0.39	0.21 (51.71 %)
100	SDC	0.29	0.28	0.03 (11.53 %)
	naive	3.96	3.68	2.55 (64.48 %)
1000	SDC	0.29	0.29	0.04 (14.79 %)
	naive	37.69	33.22	26.28 (69.70 %)
2000	SDC	0.31	0.29	0.05 (18.03 %)
	naive	72.96	65.55	52.13 (71.45 %)

discovery is in average faster than the naive engine, even for smaller numbers of Web services (*efficiency*); the time required by the SDC-engine is independent of the number of available Web services while it grows proportionally for the naive engine (*scalability*); over several invocations, the SDC-engine varies a lot less than the naive engine (*stability*). The high variation of the naive engine results from the randomized order under which the available Web services are examined. In this particular use case, the SDC optimization is mainly achieved by the pre-filtering via goal templates; the refinement step in the discovery algorithm reveals its potential when there are more usable Web services.

This indicates that the SDC technique helps to satisfy the requirements for using a Web service discovery engine as a reliable component in large SOA systems. Future-oriented works for Web service composition or business process management envision that the actual Web services for execution are dynamically discovered at runtime in order to provide better flexibility and allow compensation (e.g. [19,8]). Considering that compositions or processes can be complex and may consist of several Web services, the absolute overhead and the predictability of the discovery engine becomes a pre-requisite for realizing such technologies.

### 4.3 Discussion

For the applicability of the SDC technique in real-world applications not only the performance but also the appropriateness of the conceptual model is relevant.

To verify the our approach in this respect, we have examined its applicability for the SOA system maintained by the US-based telecommunication provider *Verizon*. This contains nearly 4000 Web services that provide basic functionalities for managing customer, product and order information; they differ in the specific usage conditions (e.g. in- and outputs, data formats, etc.). These are used by more than 600 applications that integrate the Web services into their software. The main challenges reported by the system providers is the precision of the discovery technique and the management of changes in the system. The former can be increased by discovery techniques with semantic matchmaking. The latter results from the currently hard-wired invocation of the Web services by the client applications. To overcome this, the usage tasks could be formulated in terms of

goals for which the actual Web services are discovered at runtime. Moreover, the tasks are very similar because most of the client applications work in the telecommunication domain. Thus, the relevant goal templates can be organized in a subsumption hierarchy so that the SDC technique can reveal its potential.

Another central aspect is the creation of goal templates as the central element of the SDC graph. While an initial goal template of a problem domain must be defined manually, it is possible to generate further ones on basis of the used domain ontologies. In our running example, the ontologies define several continents, countries, and weight classes. Given a goal template for package shipment in Europe, we can generate goal templates for shipment in each European country, for other continents, and so on. Eventually, we can generate all goal templates that can be expressed by the domain ontologies. These naturally are semantically similar, and thus will constitute a very fine-grained subsumption hierarchy in the resulting SDC graph. This enhances the achievable performance increase for Web service discovery: the more semantically similar goal templates are in the SDC graph, the higher is its effectiveness for reducing the search space and minimizing the number of necessary matchmaking operations.

## 5 Related Work

Although there is a wealth of work on Semantic Web services and semantically enabled Web service discovery, we are not aware of any work that addresses the performance challenge for discovery in a similar way. The following outlines the foundations of our approach and positions it within related works.

The concept of goals as an abstraction layer for facilitating problem-oriented client-system interaction has initially been developed in AI technologies like BDI agents and cognitive architectures. Inspired by the works on UPML [6], our approach has been developed in the spirit of the WSMO framework that promotes a goal-driven approach for Semantic Web services [5], and the IRS system that provides a goal-based broker for Web service usage [2]. We have integrated these approaches, and extended them with a sufficiently rich formalization and the caching mechanism for Web service discovery.

**Discovery and Semantic Matchmaking.** This has been subject to many research works that provide valuable insights on several aspects, e.g. on the architectural allocation in SWS systems [15], the quality criteria of discovery [13,9], and semantic matchmaking for different logical languages [11,14,10]. Our contribution towards this end is the two-phase Web service discovery with precise formal semantics and adequate matchmaking techniques ([17], cf. Section 2).

**Web Service Clustering.** Other, not goal-based approaches aim at reducing the search space for discovery by indexing Web service repositories. Keyword-based categorization as already supported by UDDI is imprecise in comparison to the SDC graph: it can not be ensured that the classification scheme properly reflects the functionalities provided by Web services. More sophisticated solutions perform clustering on the basis of formal descriptions. E.g. [4] creates a

search tree on based so-called interval constraints that describe Web services. These are significantly less expressive than our functional descriptions. Besides, although a logarithmic search time may be achieved (if the tree is balanced), still matchmaking is required for each new incoming request. The SDC technique can detect usable Web services without invoking a matchmaker.

**Caching.** Caching techniques are a well-established means for performance optimization applied in several areas of computing, among others also for increasing the efficiency of reasoning techniques (e.g. [13]). Respective studies show that caching can achieve the highest efficiency increase if there are many similar requests [7]. This complies with the design of our approach: the SDC graph provides the cache structure for Web service discovery, and the more similar goals and Web services exists, the higher is the achievable optimization.

## 6 Conclusions

This paper has presented a novel approach for enhancing the computational performance of Web service discovery by applying the concept of caching. We capture the minimal knowledge on the functional usability of available Web services for goal templates as generic, formal objective descriptions. At runtime, a concrete client request is formulated as a goal instance that instantiates a goal template with concrete inputs. The captured knowledge is used for optimizing the detection of usable Web services. The approach is based on a profound formal model for semantically enabled discovery. An evaluation with real-world data shows that our technique can help in the realization of scalable and reliable automated discovery engines, which becomes important for their employment as a heavily used component in larger, semantically enabled SOA systems. For the future, we plan to adopt the model to other specification languages and further integrate the caching mechanism into Semantic Web services environments.

*Acknowledgements.* The work presented in this paper has been supported by the European Commission under the projects SUPER (FP6-026850), SWING (FP6-26514), and MUSING (FP6-027097), and by the Austrian BMVIT/FFG under the FIT-IT project myOntology (Grant no. 812515/9284). The authors thank Holger Lausen, Stijn Heymans, and Dieter Fensel for fruitful discussions, and Michael Brodie (Verizon Chief Scientist) for the provision of information.

## References

1. Astrachan, O.L., Stickel, M.E.: Caching and Lemmaizing in Model Elimination Theorem Provers. In: CADE-11. Proc. of the 11th International Conference on Automated Deduction (1992)
2. Cabral, L., Domingue, J., Galizia, S., Gugliotta, A., Norton, B., Tanasescu, V., Pedrinaci, C.: IRS-III – A Broker for Semantic Web Services based Applications. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)

3. Clayton, R., Cleary, J.G., Pfahringer, B., Utting, M.: Tabling Structures for Bottom-Up Logic Programming. In: Proc. of 12th International Workshop on Logic Based Program Synthesis and Transformation, Madrid, Spain (2002)
4. Constantinescu, I., Binder, W., Faltings, B.: Flexible and Efficient Matchmaking and Ranking in Service Directories. In: ICWS 2005. Proc. of the 3rd International Conference on Web Services, Florida, USA (2005)
5. Fensel, D., Lausen, H., Polleres, A., de Bruijn, J., Stollberg, M., Roman, D., Domingue, J.: Enabling Semantic Web Services. The Web Service Modeling Ontology. Springer, Heidelberg (2006)
6. Fensel, D., et al.: The Unified Problem Solving Method Development Language UPML. Knowledge and Information Systems Journal (KAIS) 5(1) (2003)
7. Godfrey, P., Gryz, J.: Semantic Query Caching for Heterogeneous Databases. In: VLDB 1997. Proc. of 4th Knowledge Representation Meets Databases Workshop (KRDB), Athens, Greece (1997)
8. Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel, D.: Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management. In: Proc. of the IEEE ICEBE 2005, Beijing, China (2005)
9. Keller, U., Lara, R., Lausen, H., Fensel, D.: Semantic Web Service Discovery in the WSMO Framework. In: Cardoso, J. (ed.) Semantic Web: Theory, Tools and Applications, Idea Publishing, USA (2006)
10. Kifer, M., Lara, R., Polleres, A., Zhao, C., Keller, U., Lausen, H., Fensel, D.: A Logical Framework for Web Service Discovery. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, Springer, Heidelberg (2004)
11. Li, L., Horrocks, I.: A Software Framework for Matchmaking based on Semantic Web Technology. In: Proceedings of the 12th International Conference on the World Wide Web, Budapest, Hungary (2003)
12. Lu, H.: Semantic Web Services Discovery and Ranking. In: WI 2005. Proc. of the ACM International Conference on Web Intelligence, Compiegne, France (2005)
13. Di Noia, T., Di Sciascio, E., Donini, F., Mongiello, M.: A System for Principled Matchmaking in an Electronic Marketplace. In: WWW 2003. Proc. of the 12th International Conference on the World Wide Web, Budapest, Hungary (2003)
14. Paolucci, M., Kawamura, T., Payne, T., Sycara, K.: Semantic Matching of Web Services Capabilities. In: Proc. of the 1st International Semantic Web Conference, Sardinia, Italy (2002)
15. Preist, C.: A Conceptual Architecture for Semantic Web Services. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, Springer, Heidelberg (2004)
16. Stollberg, M.: Martin Hepp. Semantic Discovery Caching: Prototype & Use Case Evaluation. Technical Report DERI-2007-03-27, DERI (2007)
17. Stollberg, M., Keller, U., Lausen, H., Heymans, S.: Two-phase Web Service Discovery based on Rich Functional Descriptions. In: ESWC 2007. Proc. 4th European Semantic Web Conference, Innsbruck, Austria (2007)
18. Stollberg, M., Norton, B.: A Refined Goal Model for Semantic Web Services. In: ICIW 2007. Proc. of the 2nd International Conference on Internet and Web Applications and Services, Mauritius (2007)
19. Traverso, P., Pistore, M.: Automatic Composition of Semantic Web Services into Executable Processes. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, Springer, Heidelberg (2004)
20. Vu, L.-H., Hauswirth, M., Aberer, K.: QoS-Based Service Selection and Ranking with Trust and Reputation Management. In: Proc. of the OTM Confederated International Conferences CoopIS, DOA, and ODBASE 2005, Cyprus (2005)

# A Method for Recommending Ontology Alignment Strategies

He Tan and Patrick Lambrix

Department of Computer and Information Science  
Linköpings universitet, Sweden

**Abstract.** In different areas ontologies have been developed and many of these ontologies contain overlapping information. Often we would therefore want to be able to use multiple ontologies. To obtain good results, we need to find the relationships between terms in the different ontologies, i.e. we need to align them. Currently, there already exist a number of different alignment strategies. However, it is usually difficult for a user that needs to align two ontologies to decide which of the different available strategies are the most suitable. In this paper we propose a method that provides recommendations on alignment strategies for a given alignment problem. The method is based on the evaluation of the different available alignment strategies on several small selected pieces from the ontologies, and uses the evaluation results to provide recommendations. In the paper we give the basic steps of the method, and then illustrate and discuss the method in the setting of an alignment problem with two well-known biomedical ontologies. We also experiment with different implementations of the steps in the method.

## 1 Introduction

In recent years many ontologies have been developed. The benefits of using ontologies include reuse, sharing and portability of knowledge across platforms, and improved documentation, maintenance, and reliability (e.g. [13]). Ontologies lead to a better understanding of a field and to more effective and efficient handling of information in that field. Many of the currently developed ontologies contain overlapping information. For instance, Open Biomedical Ontologies (OBO, <http://obo.sourceforge.net/>) lists 18 different anatomy ontologies (January 2007), some of which are deprecated (e.g. Arabidopsis anatomy and Cereal anatomy) and have been replaced by a larger ontology (e.g. Plant anatomy) when the large amount of overlap was realized.

Often we would want to be able to use multiple ontologies. For instance, companies may want to use community standard ontologies and use them together with company-specific ontologies. Applications may need to use ontologies from different areas or from different views on one area. Ontology builders may want to use already existing ontologies as the basis for the creation of new ontologies by extending the existing ontologies or by combining knowledge from different smaller ontologies. In each of these cases it is important to know the relationships between the terms in the different ontologies. Further, the data in different

data sources in the same domain may have been annotated with different but similar ontologies. Knowledge of the inter-ontology relationships would in this case lead to improvements in search, integration and analysis of data. It has been realized that this is a major issue and some organizations have started to deal with it. For instance, in the area of anatomy the SOFG (<http://www.sofg.org/>) has developed the SOFG Anatomy Entry List and an NCBO anatomy workshop was organized to start the development of the Common Anatomy Reference Ontology ([http://www.bioontology.org/wiki/index.php/CARO:Main\\_Page](http://www.bioontology.org/wiki/index.php/CARO:Main_Page)).

In the remainder of this paper we say that we align two ontologies when we define the relationships between terms in the different ontologies. Currently, there exist a number of ontology alignment systems that support the user to find inter-ontology relationships. For overviews we refer to, e.g., [4,11,6] and the Ontology Matching website (<http://www.ontologymatching.org/>). These systems use different techniques, but it is not clear how well these techniques perform for different types of ontologies. Relatively few comparative evaluations on ontology alignment systems and algorithms have been performed. It is therefore difficult for a user to decide, among the different alignment strategies, which strategy or combination of strategies is best to use for aligning given ontologies.

In this paper we tackle this problem by proposing a method that provides recommendations on alignment strategies for a given alignment problem. As not much information is available on which strategies work best in which situations, we use information inherent in the actual ontologies to align. We base our method on the evaluation of the different available alignment strategies on several small selected pieces from the ontologies. These evaluation results are then used to provide recommendations. The method defines different steps: segment pair selection, segment pair alignment generation, evaluation and recommendation. The method and different steps are presented in section 3 and illustrated in the setting of an alignment problem with two well-known biomedical ontologies in section 4. Each step in the method can be instantiated by different algorithms. In section 4 we also discuss experiments with different algorithms for different steps of the method and discuss their influence on the recommendations. We conclude the paper with a conclusion and discussion of future work. In the next section we give some background and related work.

## 2 Background

**Ontology alignment.** Many of the current systems are based on the computation of similarity values between terms in the source ontologies, and can be seen as instantiations of the framework defined in [6]. This framework is shown in figure 1. It consists of two parts. The first part (*I* in figure 1) computes alignment suggestions. The second part (*II*) interacts with the user to decide on the final alignments. Some systems may not have the second part. An alignment algorithm receives as input two source ontologies. The algorithm can include several matchers. The matchers can implement strategies based on linguistic matching,



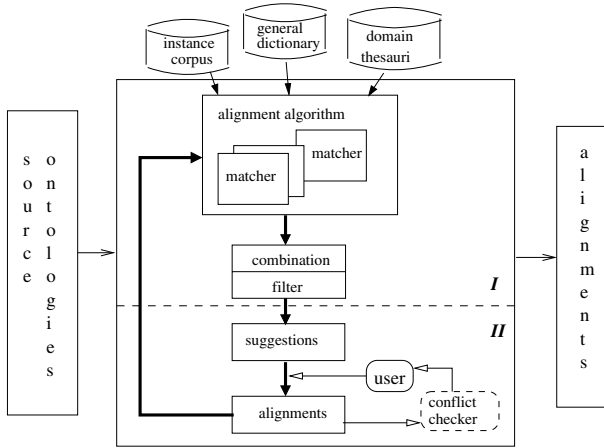


Fig. 1. Alignment framework [6]

structure-based strategies, constraint-based approaches, instance-based strategies, strategies that use auxiliary information or a combination of these. Each matcher utilizes knowledge from one or multiple sources. The matchers calculate similarities between the terms from the different source ontologies. Alignment suggestions are then determined by combining and filtering the results generated by one or more matchers. By using different matchers and combining and filtering the results in different ways we obtain different alignment strategies. The suggestions are then presented to the user who accepts or rejects them. The acceptance and rejection of a suggestion may influence further suggestions.

Further, a conflict checker is used to avoid conflicts introduced by the alignment relationships. The output of the alignment algorithm is a set of alignment relationships between terms from the source ontologies.

**Evaluation of alignment strategies.** Currently, we do not have much knowledge about how well the different alignment strategies perform for different kinds of ontologies. Comparative evaluations of ontology alignment systems have been performed by some groups. The EU OntoWeb project [10] evaluated the systems PROMPT based on Protégé (with extension Anchor-PROMPT), Chimaera (described, not evaluated), FCA-Merge and ODEMerge. This evaluation focused on such things as functionality, interoperability and visualization, but did not include tests on the quality of the alignment. In [5,6] PROMPT, Chimaera, FOAM and SAMBO were evaluated in terms of the quality of the alignment as well as the time it takes to align ontologies with these tools. Different alignment algorithms and their combinations were evaluated with different threshold values for filtering in [6]. Further, there are evaluations connected to the Ontology Alignment Evaluation Initiative (OAEI, <http://oaei.ontologymatching.org/>). The 2006 campaign consisted of 4 tracks: a comparison track, an expressive ontologies track, a directories and thesauri track, and a consensus workshop. Each track has a different



evaluation purpose. The comparison track used precision and recall as evaluation measures.

It is realized that the study of the properties, and the evaluation and comparison of the alignment strategies and their combinations, give us valuable insight in how the strategies could be used in the best way. Recently, some tools have been developed for evaluating and comparing the non-interactive part of alignment algorithms. The OAEI describes an API [3] that could be used by systems participating in the initiative. Evaluators are implemented. They compute the precision, recall, fallout and f-measure of an alignment result and a weighted symmetric difference between two alignments. KitAMO [7] provides an integrated system for comparative evaluation and analysis of alignment strategies and their combinations. KitAMO reports on similarity values, allows evaluations based on precision and recall for different combinations of different algorithms, combination weights and thresholds, as well as computes the performance of the strategies. Further, an environment is provided for analyzing the available data.

**Selecting the best alignment strategies.** The problem of selecting the best alignment strategy is tackled by [9] and [2]. In [9] it is argued that finding appropriate alignment strategies should be based on knowledge about the strategies and their previous use. As a first step a number of factors (related to input, output, approach, usage, cost and documentation) were identified that are relevant when selecting an alignment strategy. The relevant data is collected by questionnaires. The Analytic Hierarchy Process is used to detect suitable alignment approaches. In [2], APFEL, a machine learning approach to optimize alignment strategies is proposed. In APFEL a set of feature parameters are declared for the source ontologies, the similarity assessment, and the different matchers, combination and filter algorithms. To generate training data, an existing parametrization is used and alignment suggestions are generated. These suggestions need to be validated by the user. A machine learning approach is then used to learn an optimal parametrization. In the next section we propose another technique. As not much knowledge is available yet about the suitability of alignment strategies for different alignment problems (as is required by the method in [9]), we have chosen to use information about the actual ontologies to be aligned. This information is in the form of alignments between small pieces of the ontologies, which can be used to compute how well the available alignment strategies perform for these small pieces. In contrast to [2], it gives us complete information on the alignments for smaller areas in the ontologies, while [2] can only assume full knowledge about their initially generated alignment suggestions.

The problem of finding the best approach for aligning ontologies can also be seen as a variant of the general tuning problem for schema matching systems as defined in [8]. However, the scenario that is discussed in [8] is a different instance of the general tuning problem than what is tackled here. They consider the problem of finding the best approaches for matching a given (relational) schema with all other future schemas, while we tackle the problem of finding the best approaches for aligning two given ontologies.

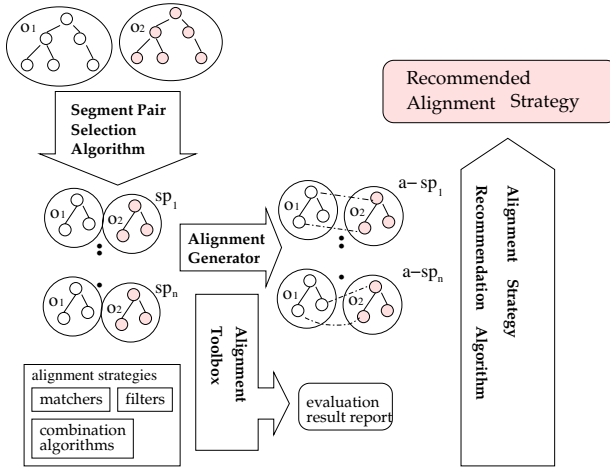


Fig. 2. Recommendation method

### 3 Recommendation Method

As a first step in our method (figure 2), the segment pair selection algorithm selects pairs of small pieces of the ontologies, called segments. For these segment pairs expected alignments need to be generated. In the alignment toolbox the available alignment strategies align the segment pairs, and reports on the alignment results are generated. Based on these reports, the recommendation algorithm gives recommendations on the strategies for aligning the two given ontologies. In the rest of this section we present each step in more detail.

**Segment Pair Selection Algorithm.** A segment of an ontology is a portion of the ontology. It represents a piece of the knowledge that the ontology represents and can be viewed as an ontology itself, usually with similar characteristics as the original ontology. The selection algorithm may use already available segment pairs (with or without their alignments). Further, the segment pairs could be manually selected by domain experts or ontology experts. Also (semi-)automatic means may be used.

Several factors regarding the selection of the segment pairs may have an influence on the final recommendation, such as, for instance, the overlap between the segments (e.g. the number of terms occurring in more than one segment), the number of segment pairs, and the number of elements included in a segment. Some recommendation algorithms may also require a certain (minimum) number of segment pairs. For those alignment strategies utilizing the structure of the ontologies and the constraint knowledge, the distribution of the segment pairs over the ontologies and the granularity within the segments may also influence the evaluation results.

**Segment Pair Alignment Generator.** For the segment pairs, expected alignments need to be generated. In some cases, alignments may already be available. The expected alignments can also be specified manually by domain experts. As the segments are only small parts of the original ontologies, the effort and complexity related to this task is much smaller than for the whole ontologies. In both these cases, there is an assumption that we have full information about the alignments, although, in general, this is not the case. Domain experts may not always agree with each other. In the case where no alignments or domain experts are readily available, it may still be possible to obtain alignments by using established bodies of domain knowledge (e.g. in the form of other ontologies and alignments) as oracles.

**Alignment Toolbox.** In the alignment toolbox the alignment strategies including the different matchers, filtering and combination algorithms are applied to align the segment pairs. A report on the alignment results is generated and given to the recommendation algorithm. The report may contain information about, for instance, the similarity values between the terms generated by the different matchers and their combinations, the alignment suggestions that are filtered out by the filtering algorithms, and the execution time for the strategies.

**Alignment Strategy Recommendation Algorithm.** The main purpose of the recommendation algorithm is to recommend one or more alignment strategies. The algorithm can return the best strategies according to a certain performance measure, or the top  $n$  strategies, or the best  $m\%$  of the strategies. For a particular alignment problem, several alignment strategies could be suitable and even strategies with a slightly lower performance may work well for the whole ontologies. The performance measure may be based on such things as the quality of the alignment suggestions (e.g. in terms of precision, recall, f-measure) or the execution performance. The different components in the performance measure may have different degrees of importance. For instance, the quality of the suggestions may be more important than the execution time.

## 4 Experiments and Illustration

In this section we illustrate our method for recommending alignment strategies using two biomedical ontologies and 6 available matchers with 5 different thresholds each. We describe implementations for the different steps and discuss the results. Further, we experimented with 2 segment pair selection algorithms, different numbers of segment pairs, and 3 recommendation measures, and discuss their influence on the recommendation results.

### 4.1 Experiment Case

**Ontologies** In the experiments we use two well-known biomedical ontologies. The NCI thesaurus ([http://ncicb.nci.nih.gov/infastructure/cacore\\_overview/-vocabulary](http://ncicb.nci.nih.gov/infastructure/cacore_overview/-vocabulary)) is a reference terminology produced by the National Cancer

Institute's Center for Bioinformatics (NCICB). The thesaurus includes broad coverage of the cancer domain, including cancer-related diseases, drugs and chemicals, genes and gene products, and anatomy. Around 34,000 terms are hierarchically organized and partitioned into 20 kinds. In our experiment we use the anatomy kind which contains 3495 terms. Within the NCICB Core Infrastructure, the NCI thesaurus together with the NCI Metathesaurus provides the semantic base for different projects. Medical Subject Headings (MeSH, <http://www.nlm.nih.gov/mesh/>) is a controlled vocabulary published by the American National Library of Medicine (NLM). It consists of sets of terms naming descriptors in a hierarchical structure. These descriptors are organized in 16 categories. The category A for anatomy terms used in the experiment includes 1391 terms. MeSH is used for indexing, cataloging, and searching for biomedical and health-related literature in MEDLINE/PubMed.

We used the Unified Medical Language System (UMLS, [http://www.nlm.nih.gov/research/umls/about\\_umls.html](http://www.nlm.nih.gov/research/umls/about_umls.html)) as an oracle during the generation of expected alignments. The Metathesaurus in UMLS contains more than 100 biomedical and health-related vocabularies, among which the NCI thesaurus and MeSH. It is organized using concepts. The concepts may have synonyms which are the terms in the different vocabularies in the Metathesaurus that have the same intended meaning. This means that the knowledge represented in UMLS can be used as an approximation of domain expert knowledge and UMLS can be used as an oracle in its domain. As the NCI thesaurus and MeSH are included in the Metathesaurus of UMLS, alignments are available. According to UMLS there are 919 expected alignments for the two ontologies.

**Alignment Strategies.** We experiment with four linguistic matchers, a weighted sum combination algorithm and a threshold filter. The n-gram (NG) and edit-distance (ED) matchers use approximate string matching algorithms. An n-gram is a set of n consecutive characters extracted from a string. Similar strings will have a high proportion of n-grams in common. Edit distance is defined as the number of deletions, insertions, or substitutions required to transform one string into the other. The greater the edit distance, the more different the strings are. The two other matchers (WL and WN) compute the similarity between two terms by comparing the lists of words of which the terms are composed. Similar terms have a high proportion of words in common. Both matchers use a Porter stemming algorithm. The more advanced (WN) of the two also uses WordNet (<http://wordnet.princeton.edu/>), which has a good coverage of anatomy [1], during the computation of the similarity values, by using the hypernym relationships in WordNet. All matchers compute similarity values in  $[0..1]$ , where the higher the value the more similar two terms. The combination algorithm is a weighted sum,  $Sim(t_1, t_2) = \sum_{k=1}^n w_k \cdot sim_k(t_1, t_2)$ , where  $n$  is the number of the combined matchers and  $sim_k$  and  $w_k$  represent the similarity values and weights, respectively, for the different matchers. We also require that  $w_k \in [0, 1]$  and  $\sum_{k=1}^n w_k = 1$ . In the experiment we evaluate two combinations.

The first combination (C1) includes n-gram, edit-distance and the first word list matcher. The second combination (C2) includes n-gram, edit-distance and the word list matcher with WordNet. In both combinations all weights are set to  $\frac{1}{3}$ . The threshold filter allows only term pairs with similarity values higher than or equal to the threshold value as alignment suggestions. We use thresholds 0.4, 0.5, 0.6, 0.7 and 0.8. This means that we have 30 available strategies in total (6 matchers with 5 thresholds each).

## 4.2 Algorithms in the Recommendation Process

**Segment Pair Selection Algorithms.** For this experiment we developed two algorithms that select segment pairs. The first algorithm (SubG) collects the pairs of terms in the two ontologies which have the same name (case-insensitive string matching). The pairs of sub-graphs of the ontologies rooted at these terms with respect to the *is-a* and *part-of* hierarchies are candidate segment pairs. The segment pairs are randomly chosen from the candidate segment pairs, with the restriction that the segments are pairwise disjoint. We also required that the number of terms in a segment is strictly between 1 and 60. This avoids leaves in the ontologies as well as too large segments.

In the second algorithm (Clust) the terms in the ontologies are first partitioned into clusters. We use a variant of the algorithm proposed in [12], where a dependency and the strength of the dependency between two terms is defined based on the *is-a* and *part-of* hierarchies of the ontology. The clusters satisfy the intuition that the dependency between any two terms in a cluster is stronger than the dependency between a term in the cluster and a term that is not in the cluster. Further, we require that the number of terms in a cluster is at least 5. The candidate segment pairs are the pairs of clusters from the two ontologies including at least one pair of terms with the same name (case-insensitive string matching). The segment pairs are then randomly chosen from the candidate segment pairs.

In the experiment we generate 5 segment pairs per trial. We experimented with 3 different generated segment pair sets per segment pair selection algorithm, but also used their combinations. This means that for each segment pair selection algorithm, we have 3 sets with 5 segment pairs, 3 sets with 10 segment pairs and 1 set with 15 segment pairs.

**Segment Pair Alignment Generator.** We use UMLS as alignment generator. We query the ontology terms in the UMLS Metathesaurus using their names. If the queries for two terms in different ontologies return the same UMLS concept, we consider the pair of terms as an expected alignment.

**Alignment Toolbox.** We use the current implementation of KitAMO [7] as the alignment toolbox. We input the segments, the results from the alignment generator and the matchers, combination algorithm and filter method into KitAMO. KitAMO runs the alignment strategies and produces reports on the

similarity values generated by the different matchers, their execution time, and the number of correct, wrong and redundant suggestions for different thresholds.

**Alignment Strategy Recommendation Algorithm.** In this experiment we calculate a recommendation score for the alignment strategies as a weighted sum of property measures,  $\sum_{i=1}^m w_i \cdot p_i$ , where  $w_i$  is the weight of the property, and  $p_i$  is the score of the property. The higher the recommendation score, the more preferred the alignment strategy is. Our algorithm returns a ranking of the available strategies based on the recommendation score. This ranking can then be used to return the best, top  $n$ , or best  $m\%$  of the strategies.

We used two properties. The first property is the quality of the alignment suggestions. It is measured as  $\frac{1}{m} \cdot \sum_{s=1}^m f_s$ , where  $m$  is the number of segment pairs, and  $f_s$  the f-measure value. The f-measure integrates precision and recall. It is calculated as  $\frac{P \cdot R}{(1-\alpha) \cdot P + \alpha \cdot R}$ , in which the  $P$  is the precision (the number of correct suggestions divided by the number of suggestions),  $R$  the recall (the number of correct suggestions divided by the number of expected alignments) and  $\alpha$  a number between 0 and 1. The higher the value for  $\alpha$ , the more important precision is with respect to recall. The value of  $\alpha$  is 0.5 in the experiment, i.e. we use the harmonic mean of precision and recall. The second property is the execution performance of an alignment strategy. It is measured as  $-\frac{1}{m} \cdot \sum_{s=1}^m \frac{t_s}{n_s}$ , where  $m$  is the number of the segment pairs,  $t_s$  is the execution time the alignment strategy needs to calculate similarity values for a segment pair, and  $n_s$  is the number of term pairs in the segment pair. The execution time in this experiment is calculated using the KitAMO system.

We experimented with three different ways to compute a recommendation score using the two properties. In the first case, we only consider the quality of the alignment suggestions (F). In the second case we give equal weight to the quality of the suggestions and the execution performance (F+E). In the last case, the quality is weighted ten times higher than the execution performance (10F+E).

**Expected Recommendations.** Table 1 represents the top 3 (or top 10%) alignment strategies per measure for the whole ontologies. A perfect recommendation for the experiment would thus be the simple word list matcher with threshold 0.8 (WL,0.8) for measures F (f-measure) and 10F+E (10 times f-measure and execution performance), and edit distance with threshold 0.8 (ED,0.8) for measure F+E (f-measure and execution performance).

**Table 1.** Top 3 alignment strategies per measure

F	F+E	10F+E
1. (WL,0.8)	1. (ED,0.8)	1. (WL,0.8)
2. (C1,0.8)	2. (WL,0.8)	2. (C1,0.8)
3. (C2,0.8)	3. (NG,0.7)	3. (WL,0.7)

### 4.3 Results

Tables 2 and 3 present the top 3 recommendations given in the experiment for measure F and segment pair selection algorithms SubG and Clust, respectively. The 'SPS's represent the different generated segment pair sets. SPS A(1+2) is the segment pair set including both SPS A1 and SPS A2, and similarly for the other combination segment pair sets. For instance, table 2 shows that using A2, our algorithm recommends the simple word list matcher with thresholds 0.8 (WL,0.8) and 0.7 (WL,0.7) as best, respectively second best strategy, while the matcher with WordNet and threshold 0.7 (WN,0.7) is the third choice. Figure 3 shows the recommendation scores for the different matchers for the measure F and the SPSs containing 5 elements.

**Table 2.** Top 3 recommendations per segment pair set for segment pair selection algorithm SubG and measure F

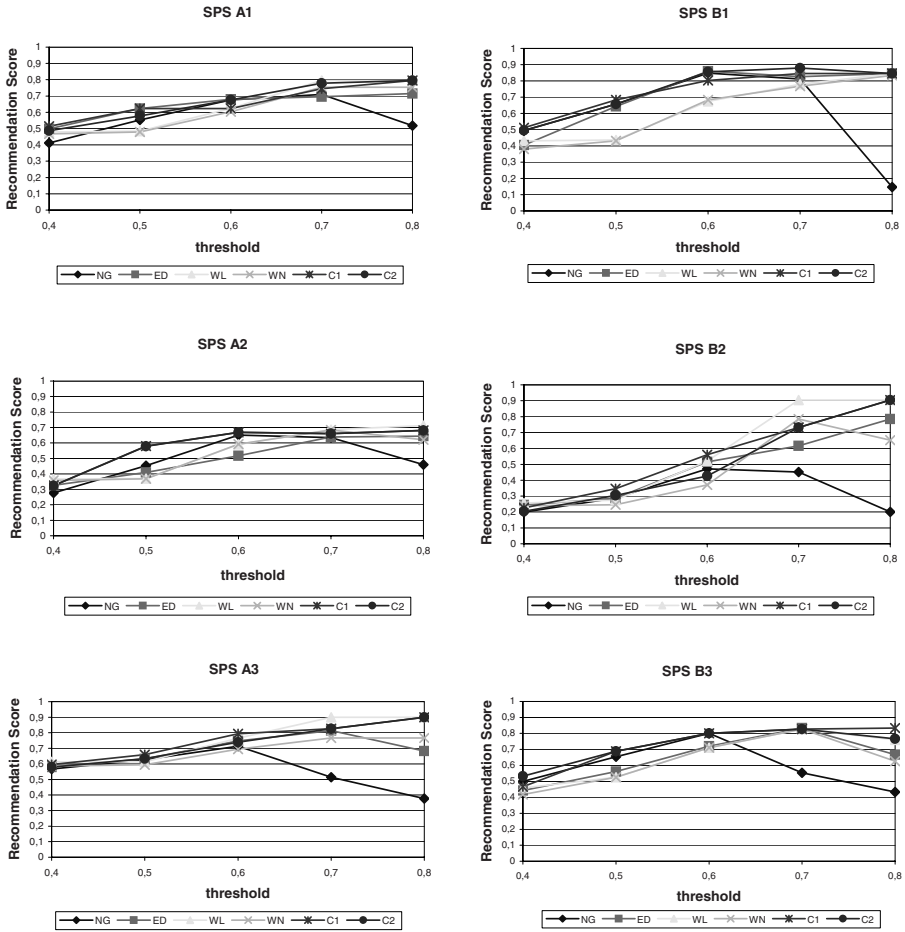
SPS A1	SPS A2	SPS A3	SPS A(1+2)	SPS A(1+3)	SPS A(2+3)	SPS A(1+2+3)
1. (WL,0.8)	1. (WL,0.8)	1. (WL,0.8)	1.(WL,0.8)	1.(WL,0.8)	1.(WL,0.8)	1. (WL,0.8)
1. (WL,0.7)	2. (WL,0.7)	1. (WL,0.7)	2.(WL,0.7)	1.(WL,0.7)	2.(WL,0.7)	2. (WL,0.7)
1. (C1,0.8)	3. (WN,0.7)	1. (C1,0.8)	3.(C2,0.8)	1.(C2,0.8)	3.(C2,0.8)	3. (C2,0.8)
1. (C2,0.8)		1. (C2,0.8)	3.(C1,0.8)	1.(C1,0.8)	3.(C1,0.8)	3. (C1,0.8)

**Table 3.** Top 3 recommendations per segment pair set for segment pair selection algorithm Clust and measure F

SPS B1	SPS B2	SPS B3	SPS B(1+2)	SPS B(1+3)	SPS B(2+3)	SPS B(1+2+3)
1. (C2,0.7)	1. (WL,0.8)	1. (C1,0.8)	1. (WL,0.8)	1. (C2,0.7)	1. (C1,0.8)	1. (C1,0.8)
2. (ED,0.6)	1. (WL,0.7)	1. (ED,0.7)	1. (C1,0.8)	2. (C1,0.8)	2. (WL,0.7)	2. (C2,0.8)
3. (C2,0.6)	1. (C1,0.8)	3. (C1,0.7)	1. (C2,0.8)	3. (C1,0.7)	3. (C2,0.8)	3. (WL,0.8)
	1. (C2,0.8)	3. (C2,0.7)				3. (WL,0.7)
		3. (WL,0.7)				
		3. (WN,0.7)				

There is no overlap between the segment pair sets in the experiments for SubG (A1, A2 and A3), i.e. no term occurs in more than one segment. Similarly, there is no overlap between the segment pair sets in the experiments for Clust (B1, B2 and B3). In the experiments for SubG, the number of terms in a segment ranges from 2 to 34, and the levels in the *is-a* and *part-of* hierarchies in the segments range from 2 to 6. The number of expected alignments between segments ranges from 1 to 4 for A1, 1 to 23 for A2, and 1 to 5 for A3. In the experiments for Clust, the number of terms in a segment ranges from 5 to 14, and the levels in the *is-a* and *part-of* hierarchies in the segments range from 2 to 3. The number of expected alignments between segments ranges from 1 to 4 for B1, 1 to 3 for B2, and 1 to 3 for B3.

Table 4 shows the top 3 recommendations for measures F+E and 10F+E for the segment pair selection algorithms SubG and Clust. We only show the results for the segment pair sets with 5 elements.



**Fig. 3.** Recommendation scores for the different matchers per segment pair set for the measure F. (A1, A2 and A3 for algorithm SubG. B1, B2 and B3 for algorithm Clust.)

**Table 4.** Top 3 recommendations per segment pair set for the recommendation measures F+E and 10F+E. (A1, A2 and A3 for algorithm SubG. B1, B2 and B3 for algorithm Clust.)

	SPS A1	SPS A2	SPS A3	SPS B1	SPS B2	SPS B3
F+E	1. (WL,0.8)	1. (WL,0.8)	1. (WL,0.8)	1. (C2,0.7)	1.(WL,0.8)	1.(ED,0.7)
	1. (WL,0.7)	2. (WL,0.7)	1. (WL,0.7)	2. (ED,0.6)	1.(WL,0.7)	2.(C1,0.8)
	3. (C1,0.8)	3. (C1,0.8)	3. (C1,0.8)	3. (C2,0.6)	3.(C1,0.8)	3.(WL,0.7)
10F+E	1. (WL,0.8)	1. (WL,0.8)	1. (WL,0.8)	1. (C2,0.7)	1.(WL,0.8)	1.(ED,0.7)
	1. (WL,0.7)	2. (WL,0.7)	1. (WL,0.7)	2. (ED,0.6)	1.(WL,0.7)	2.(C1,0.8)
	3. (C1,0.8)	3. (WN,0.7)	3. (C1,0.8)	3. (C2,0.6)	3.(C1,0.8)	3.(WL,0.7)



## 4.4 Discussion

Table 2 shows that the quality of the recommendations for measure F provided by the SPSs which are generated by the algorithm SubG is excellent. The best strategy for this alignment problem ((WL,0.8), see table 1) has always rank 1. The other top 3 strategies ((C1,0.8) and (C2,0.8)) are also often included in the highly recommended strategies. They have both rank 1 for A1 and A3 and (not shown in table) both rank 4 for A2. Regarding the other strategies that are recommended, (WL,0.7) also performs well for the whole ontologies (rank 4), while (WN,0.7) has rank 16.

Table 3 shows that the quality of the recommendations given by the SPSs which are generated by Clust is also good. For B2 we obtain all top 3 strategies. For B3 one of the top 3 strategies (C1,0.8) is recommended, while the others have rank 10 and 11. For B1 the top 3 have all recommendation rank 5. Regarding the other strategies that are recommended, (WL,0.7), (C1,0.7) and (C2,0.7) also perform well for the whole ontologies (ranks 4, 5 and 6), while (ED,0.6), (ED,0.7), (C2,0.6) and (WN,0.7) have ranks between 9 and 16.

The complete results for SubG and Clust with SPS containing 5 elements and measure F are shown in figure 3. For several matchers we can observe a certain trend in the behavior. Although the scores are different, often they behave in a similar way across the sets. This may be useful information to prune some of the computations.

From tables 2 and 3, we also observe that the combination of the results from different segment pair sets often provides higher quality recommendations. For instance, for SubG, the best strategy (WL,0.8) is often ranked alone as number 1 in the combinations (A(1+2),A(2+3),A(1+2+3)), while the other top 3 strategies are also top 3 recommendations. Also, the second recommendation (ED,0.6) for B1, which is ranked 17th for the whole ontologies, has dropped to rank 13 for B(1+2+3). This is in line with the intuition that more information may give better alignment results.

As shown in tables 2 and 3, the recommendations based on the SPSs generated by SubG are usually better than the ones from Clust.

In total the number of expected alignments in SPSs is larger for SubG than for Clust. However, this does not seem to influence the recommendation results. For example, SPS A2 gives the worst recommendations for SubG, although it contains the highest number of expected alignments in the three SPSs from SubG. SPS B2 is the only SPS for Clust that recommends the top 3 strategies, although it has the smallest number of alignments.

The results from the experiments with 10F+E are similar to the results of the experiments with F. This is not surprising as the f-measure is by large the most important component in 10F+E. The differences are that the matchers using WordNet are not as highly ranked anymore. When F+E is the measurement, the quality of the recommendations given by the SPSs from both SubG and Clust is not as good. The best strategy (ED,0.8) (see table 1) is ranked between 5 (for B1 and B2) and 11 (for A3). Also the third best strategy (NG,0.7) has relatively low ranks (between 7 and 21). The second best strategy (WL,0.8)

has rank 1 for all the SubG SPS, and ranks 7, 1, and 9 for B1, B2 and B3, respectively. Regarding the other recommended strategies, (ED,0.7), (C1,0.8), and (WL,0.7) also perform well for the whole ontologies (ranks 4, 5 and 6), while (ED,0.6), (C2,0.6) and (C2,0.7) have ranks between 11 and 20. The method for calculating the property measures may influence the results. For instance, in the experiment the execution time tends to increase from NG to ED to WL to WN. However, in practice there is an influence from the run-time environment which, for instance, gave very different execution times of 0.001 and 0.289 for WL with similar segment pairs. The value for F is between 0 and 1, and thus F+E is sensitive to the variation. Taking an average over several runs, may alleviate the problem in this case.

## 5 Conclusion

In this paper we have tackled the problem of deciding on which strategy to use for a particular alignment problem. We have done this by proposing a method for recommending alignment strategies. As we do not have much knowledge yet regarding the suitability of the current algorithms, we have proposed a method that uses evaluation results from the strategies on small pieces of the ontologies to be aligned. This requires a small effort of the user and as the original ontologies are used, we implicitly use knowledge about these ontologies. We have illustrated the method for an alignment problem using two anatomy ontologies and different alignment strategies. We also experimented with different segment pair selection algorithms, different numbers of segment pairs, and different recommendation measures, and discussed their influence on the recommendations. We also showed that for this alignment problem good results were obtained even with reasonably simple strategies.

There are several issues for future work. Even though we have shown the feasibility of our method and have obtained good results for the alignment problem in the experiment, it is necessary to perform more experiments with different kinds of ontologies and alignment strategies. We also want to further investigate the influence of the different choices in the different steps of the method. This includes investigating other segment pair selection strategies, recommendation measures and recommendation algorithms. We intend to develop a tool that supports these investigations by extending the KitAMO system. It will also be interesting to look at how the ideas from [9] and [2] can be used to augment our approach. For instance, when more knowledge is obtained regarding the different strategies and their previous use (as in [9]), this knowledge could be used as a first step to filter the available strategies and it can be used by the recommendation strategy. Also the optimization approach in [2] may be useful for finding better combinations as well as within the recommendation step. Finally, we intend to extend the SAMBO ontology alignment tool [6] with a recommendation component.

**Acknowledgements.** We thank Bo Servenius for recommending us to use UMLS as an oracle. We acknowledge the financial support of the Swedish Research Council, the Center for Industrial Information Technology, the Swedish national graduate school in computer science, and the EU NoE REWERSE (FP6 project 506779).

## References

1. Bodenreider, O., Burgun, A.: Characterizing the Definitions of Anatomical Concepts in WordNet and Specialized Sources. In: Proc. of the First Global WordNet Conference, pp. 223–230 (2002)
2. Ehrig, M., Staab, S., Sure, Y.: Bootstrapping Ontology Alignment Methods with APFEL. In: Proc. of the Int. Semantic Web Conference, pp. 186–200 (2005)
3. Euzenat, J.: An API for ontology alignment, version 2.1, Ontology Alignment Evaluation Initiative (2006)
4. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. *The Knowledge Engineering Review* 18(1), 1–31 (2003)
5. Lambrix, P., Edberg, A.: Evaluation of ontology merging tools in bioinformatics. *Proc. of the Pacific Symposium on Biocomputing* 8, 589–600 (2003)
6. Lambrix, P., Tan, H.: SAMBO - A System for Aligning and Merging Biomedical Ontologies. *Journal of Web Semantics* 4(3), 196–206 (2006)
7. Lambrix, P., Tan, H.: A Tool for Evaluating Ontology Alignment Strategies. *Journal on Data Semantics VIII*, 182–202 (2007)
8. Lee, Y., Sayyadian, M., Doan, A., Rosenthal, A.: eTuner: tuning schema matching software using synthetic scenarios. *VLDB Journal* 16(1), 97–122 (2007)
9. Mochol, M., Jentzsch, A., Euzenat, J.: Applying an Analytic Method for Matching Approach Selection. *Proc. of the Int. Workshop on Ontology Matching* (2006)
10. OntoWeb Consortium, A survey on ontology tools, Deliverable 1.3 (2002)
11. Shvaiko, P., Euzenat, J.: A Survey of Schema-Based Matching Approaches. *Journal on Data Semantics IV*, 146–171 (2005)
12. Stuckenschmidt, H., Klein, M.: Structure-based partitioning of large concept hierarchies. *Proc. of the 3rd Int. Semantic Web Conference*, 289–303 (2004)
13. Uschold, M., Jasper, R.: A Framework for Understanding and Classifying Ontology Applications. *Proc. of the IJCAI Workshop on Ontologies and Problem-Solving Methods: Lessons Learned and Future Trends* (1999)

# Lifecycle-Support in Architectures for Ontology-Based Information Systems

Thanh Tran<sup>1</sup>, Peter Haase<sup>1</sup>, Holger Lewen<sup>1</sup>, Óscar Muñoz-García<sup>2</sup>,  
Asunción Gómez-Pérez<sup>2</sup>, and Rudi Studer<sup>1</sup>

<sup>1</sup> Institute AIFB, Universität Karlsruhe, Germany  
{dtr,pha,hle,rst}@aifb.uni-karlsruhe.de

<sup>2</sup> Universidad Politécnica de Madrid, Spain  
{omunoz,asun}@fi.upm.es

**Abstract.** Ontology-based applications play an increasingly important role in the public and corporate Semantic Web. While today there exist a range of tools and technologies to support specific ontology engineering and management activities, architectural design guidelines for building ontology-based applications are missing. In this paper, we present an architecture for ontology-based applications—covering the complete ontology-lifecycle—that is intended to support software engineers in designing and developing ontology-based applications. We illustrate the use of the architecture in a concrete case study using the NeOn toolkit as one implementation of the architecture.

## 1 Introduction

Ontology-based applications play an increasingly important role in the public and corporate Semantic Web. Major companies like Oracle<sup>[1]</sup> and IBM<sup>[2]</sup> have invested in semantic technologies. These efforts and work from the research community have led to a number of concrete implementations to support specific ontology engineering and management activities. Yet, there are not many ontology-based information systems (OIS) available that can exploit these technologies to deliver added value for the end user.

Partly, this is due to the lack of guidance for software engineers to develop OIS. Methodologies for the development of knowledge-based applications (e.g. CommonKADS<sup>[3]</sup>) can be applied to OIS, but normally focus purely on knowledge engineering. Architectures for semantic web services involve ontologies, but naturally focus on services. For instance, WSMO<sup>[2]</sup> or ODE-SWS<sup>[3]</sup> provide ontology-based mechanisms to formally describe services. While ontologies are a main component of these frameworks, ontology management features are not supported. Also guidance as to how ontologies can be used and managed at runtime by the service platform are not provided. Even results of the W3C group on best practices and deployment<sup>[4]</sup> cover only usage scenarios and guideline for ontology developments. The Semantic Web Framework (SMF) proposal<sup>[4]</sup> focusses on identifying and describing components including their dependencies. This work is complementary to our work in the sense that while SMF so far

<sup>1</sup> [http://www.oracle.com/technology/tech/semantic\\_technologies/index.html](http://www.oracle.com/technology/tech/semantic_technologies/index.html)

<sup>2</sup> <http://www.alphaworks.ibm.com/tech/semanticstk>

<sup>3</sup> <http://www.w3.org/2001/sw/BestPractices/>

identifies and classifies components required for managing ontology, we focus on OIS architectures with lifecycle-support.

In particular, we discuss activities that must be supported in OIS on the basis of the notion of ontology lifecycle. Applying best practices and architecture paradigms such as SOA [5] and J2EE [6] from the software engineering community, we develop a generic architecture of integrated OIS that can even support scenarios where usage and engineering activities are intertwined at runtime. This architecture aims to provide a guideline for software engineers to design OIS. As an implementation of this architecture, we also discuss the NeOn toolkit<sup>4</sup>, which provides a concrete framework containing reusable components that can be leveraged for the implementation of OIS. We demonstrate the application of both the architecture and the NeOn toolkit on the basis of a case study in the fishery domain.

The paper is structured as follows: In Section 2 we start with the discussion of the ontology lifecycle in ontology-based information systems. In Section 3 we present the generic architecture for OIS and illustrate how this architecture supports ontology lifecycle management. In Section 4 we then provide an instantiation of the generic architecture using the NeOn toolkit within a case study in the fishery domain. We conclude the paper with an outlook in Section 5.

## 2 Lifecycle Management of Ontologies

In this section, we briefly present existing work on the *ontology lifecycle*. The concept has mainly been used in methodologies for ontology engineering [7]. In the following, we give a compiled overview of these methodologies to present a simple lifecycle model (see Fig. 1). This model considers not only the engineering, but also the usage of ontologies at runtime as well as the interplay between usage and engineering activities.

### 2.1 Ontology Engineering

While the individual methodologies for ontology engineering vary, they agree on the main lifecycle activities, namely *requirement analysis*, *development*, *evaluation*, and *maintenance*, plus orthogonal activities such as project management. In the following, we focus on the first three engineering-related activities as described in the literature and then discuss maintenance in the context of usage-related activities.

**Requirement Analysis:** In this step, domain experts and ontology engineers analyze scenarios, use cases, and, in particular, intended retrieval and reasoning tasks performed on the ontology.

**Development:** This is the step in which the methodologies vary most. We therefore present an aggregated view on the different proposals for ontology development.

The initial step is the identification of already available reusable ontologies and other sources such as taxonomies or database schemas. Once reusable ontologies are found, they have to be adapted to the specific requirements of the application. This may include both backward (understanding, restructuring, modifying) and forward (modifying, extending) engineering of these reusable ontologies w.r.t. some design patterns.

---

<sup>4</sup> <http://www.neon-toolkit.org/>

Then, the ontologies are translated to the target representation language. Because of the expressivity-scalability tradeoff involved in reasoning, it may be desirable to tweak the degree of axiomatization, e.g. for performance. An important aspect in development is collaboration. Existing proposals for reaching consensus knowledge involve the assignment of roles and the definition of interaction protocols for knowledge engineers.

**Integration:** Inspired by the componentization of software, recent approaches advocate the modularization of ontologies [8]. Accordingly, the result of the development step shall be a set of modularized ontologies rather than one monolithic ontology. These modules have to be integrated, e.g. via the definition of import declarations and alignment rules. This integration concerns not only the modules that have been developed for the given use case. For interoperability with external applications, they may be embedded in a larger context, e.g. integrated with ontologies employed by other OIS.

**Evaluation:** Similar to bugs in software, inconsistencies in ontologies impede their proper use. So the initial evaluation step is to check for inconsistencies, both at the level of modules and in an integrated setting. Furthermore, ontologies also have to be assessed w.r.t. specific requirements derived from the use cases. Note that any deficiencies detected in this phase have to be addressed, i.e. led back to development.

## 2.2 Ontology Usage

Usage encompasses all activities performed with an ontology after it has been engineered. So far, the lifecycle as described in the literature is more of a static nature, just like the software lifecycle. Namely, if all requirements are met, the ontology will be deployed and the lifecycle continues with ontology evolution—also referred to as maintenance in literature. In this phase, new requirements may arise which are fed back into the loop, e.g. incorporated into the next release, which is then redeployed. Current lifecycle models however do not incorporate activities involved in the actual usage of ontologies. We will elaborate on these activities and based on them, show that the lifecycle can be dynamic.

**Search and Retrieval and Reasoning:** Once the ontologies have been created, they can be used to realize information access in the application, for example via search and retrieval. Typically an OIS involves a reasoner to infer implicit knowledge. The schema can be combined with instance data to support advanced retrieval, e.g. schema

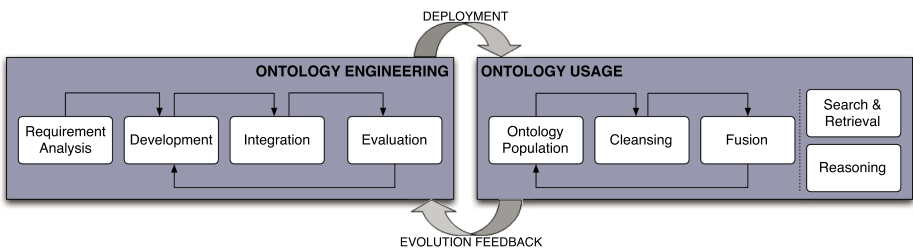


Fig. 1. Lifecycle Model

knowledge exploited for query enhancement (refinement, expansion), and A-Box reasoning to retrieve also inferred knowledge.

Note these are two generic exemplary tasks that shall illustrate the use of ontologies. In the actual application, search and retrieval may be only two of the many ontology-related operations that are embedded in more complex (business) logic implementing a concrete use case. These usages of ontologies may require support by the following application-independent lifecycle activities that are also performed at runtime:

**Ontology Population:** To populate the knowledge base (KB), instances may be collected from the user, e.g. via forms. A substantial overhead may be imposed to the user when all instance data has to be created manually. This burden can be alleviated by a (semi)-automatic population of the KB. Part of this knowledge creation step are also the manipulation and deletion of instances.

**Cleansing and Fusion:** Automatically extracted knowledge cannot be assumed to have the desired quality. Enhancing instance data may include identification and merging of conceptually identical instances that are only differently labeled (object identification) as well as fusion at the level of statements, e.g. merging redundant statements.

Both the population and the fusion steps may lead to inconsistencies which have to be resolved. Consider a user requesting data that yet has to be crawled from external sources. Then, inconsistencies that may arise in the process have to be resolved at runtime for the user to be able to continue his work. Found inconsistencies are fed back to debugging and the development-phase of the ontology lifecycle. That is, ontology evolution—the loop from usage back to engineering activities—is not only due to changing requirements but is also necessary for the runtime usage of ontologies.

### 3 A Generic OIS Architecture with Lifecycle Support

In this section, we present a generic architecture that aims to serve as a guideline for the development of any IS that involves ontologies. Hence, generic use cases that have to be considered may involve mere ontology engineering, mere ontology usage or a combination of both. Therefore, lifecycle activities discussed in the last section will be incorporated as functional requirements. Due to the possible dynamic nature of the lifecycle, it has to be supported in an integrated architecture that allows for a dynamic interaction of engineering and usage activities.

We will start with an overview and continue with a detailed elaboration on the components for lifecycle support. Then, we show how this generic architecture can be adopted for the development of OIS with concrete functional requirements. While the presented architecture abstracts from specific application settings, we also discuss how concrete architecture paradigms can be applied to meet technological requirements.

#### 3.1 Overview of the Architecture

The proposed architecture as shown in Fig. 2 is organized in layers according to the control- and data flow (the control flow is indicated by the arrows) as well as the degree of abstraction of the constituent components. The former means that components at a

higher layer invoke and request data from components at the lower layers. The latter means that components at the same abstraction level can be found on the same architecture layer. A single operation of components at a higher level of abstraction can trigger several low level operations. For example, a functionality provided by an ontology-based application front-end may invoke some ontology usage services, each of them, in turn, making use of several core ontology services. These services rely on requests to specific data sources, which are accessed via connectors of the data abstraction layer.

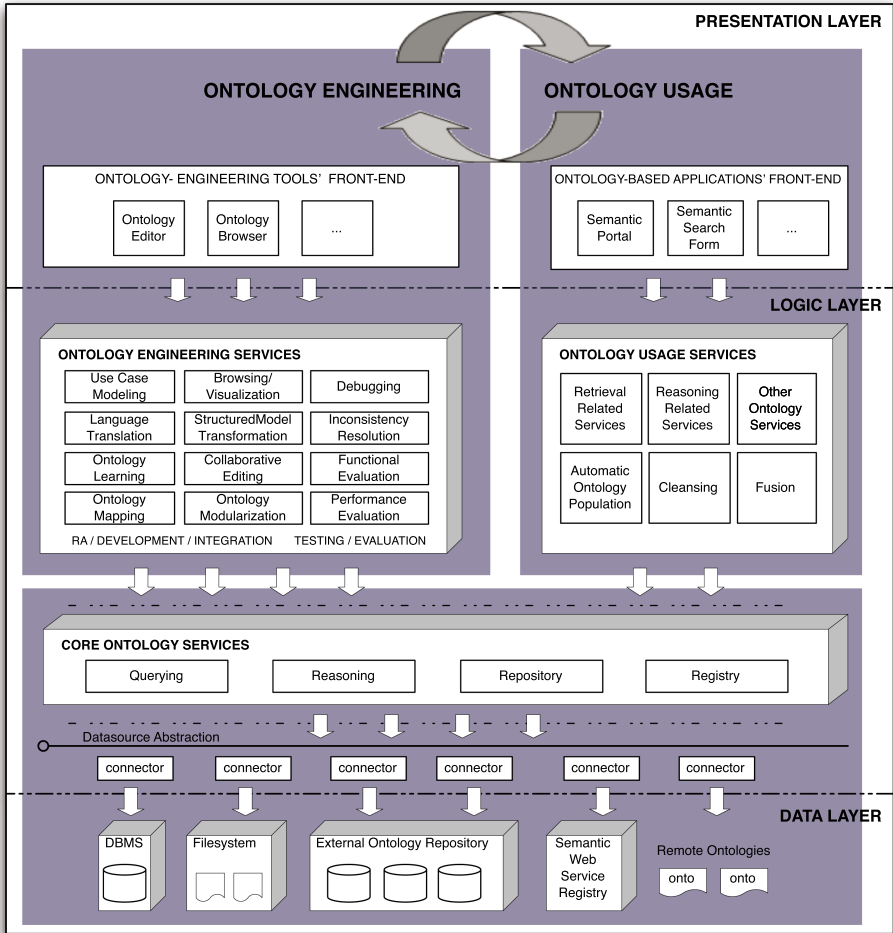


Fig. 2. The Generic Architecture

Note that many of the concepts employed for this architecture proposal, i.e. the presentation components, platform services, data source abstraction and connectors follow J2EE and SOA best practices. Also, the organization in (three different) layers is inspired from the n-tier architecture—a well-known organization principle in software



engineering. We now briefly discuss these concepts and the components at the different layers (see [6,5] for more information on J2EE and SOA best practices).

**The Data Layer:** This layer hosts any kind of *datasources*, including *databases* and *file systems*. This may also encompass ontological sources such as *external ontologies* hosted in repositories, *semantic web services* hosted in registries and any ontology on the web that can be retrieved via its URI. Note that services external to the system can be regarded as a component of the data layer because their processing is transparent to the internal components. The processing can be considered a black-box that simply provides data for internal components (see *connectors* in [9]).

**The Logic Layer:** At this layer, there are application-specific services that are implemented for a particular use case and operate on specific object models. The former encapsulate the processing logic and the latter capture the data. These services invoke *ontology lifecycle services* to manage and retrieve semantic data. Accordingly, object models may encapsulate data coming from conventional datasources like databases (data) or from ontological sources (semantic data), or both. In any case, the actual data comes from a persistent storage, most often a database. The data source abstraction can be used to hide specific datasource implementations by providing a uniform API and specific connectors. While not shown in Fig. 2, services at the logic layer run on a specific platform, which provides orthogonal functionalities for the management, configuration, and identification (registry) of services as well as access control and security.

**The Presentation Layer:** This layer hosts presentation components that the user interacts with. These components could be simply *pages* or *interactive forms* of a web-based system or more *sophisticated UIs* of a desktop application that contains a variety of widgets. The engineering and usage operations performed by the user on these components translate to calls to services situated at the logic layer. The data returned by these services is then presented by the components together with the static content.

We will now continue with a more detailed elaboration on ontology-related services.

### 3.2 Ontology-Related Services

Ontology-related services are organized in one layer for core services and one layer for the higher level ontology lifecycle services. While the control and data flow of lifecycle and core services are top-down as shown in Fig. 2, the interaction between the different lifecycle activities typically corresponds to the structure of the corresponding lifecycle activities, e.g. they follow a sequential flow. However, the actual interaction depends on the needs of a particular use case. That is, ontology lifecycle services can be invoked and controlled by application-specific services as needed.

**Core Ontology Services:** Functionalities offered by services at this layers are used by lifecycle services. An *ontology registry service* is used to find and publish ontologies. An *ontology repository service* provides access, manipulation and storage (persistence is supported by the lower level datastore) at the level of ontologies and at the level of ontology elements. That is, repository functionalities are also available for axioms, concepts, properties, individuals etc. The repository service also includes logging and versioning to ensure reversibility. Besides the common repository retrieval methods, a

*query service* offers a generic mechanism for retrieval. Finally, a *reasoning service* is available for standard reasoning tasks such as consistency checking, classification etc.

**Ontology Engineering Services:** The architecture contains services for the requirement analysis that has functionalities similar to the ones supported in an IDE for software development, e.g. for requirements elicitation, *modeling of use cases* and specification of reasoning and retrieval tasks involved in the use cases.

In the actual development, services are provided for ontology browsing, visualization, editing and integration. In particular, *browsing* and *visualization* supporting ontologies as well as non-ontological artifacts such as interface signatures, data base schema, and UML models to help in identifying reusable artifacts. To enable reuse, there are services for the *translation* of existing ontologies to the target representation formalism. Services for (semi)-automatic *transformation* of non-ontological sources to ontologies are also incorporated into the architecture [10] to facilitate reuse. This transformation is possible in both directions to ensure the interoperability of ontology data w.r.t. these data sources. Services for *ontology learning* are also provided to accelerate the development process by the generation of a base version that can be further refined. Implementations of specific interaction protocols enable a *collaborative editing* process. The *mapping service* includes support for the identification and specification of ontology modules as well as their relations and dependencies. Also, it includes the specification of concept mappings required for the alignment of ontologies.

After the base ontologies have been further developed, adapted to requirements and integrated, they have to be tested and evaluated. For these tasks, there are services for *debugging* (identification of axioms that are responsible for or affected by the inconsistency) and for the *inconsistency resolution* of the conflicts [11]. Also, there are services that evaluate the coverage of the ontology w.r.t. the representative set of retrieval and reasoning tasks envisaged for the use cases (*functional evaluation*). Finally, *performance evaluation* services are essential to meet the requirements and are incorporated into the architecture. In order to meet performance targets for particular scenarios, different configurations for ontology axiomatization may be considered.

**Ontology Usage Services:** In Fig. 2 some application-specific services are shown to illustrate that ontologies may be used as a technology to implement use cases of a particular OIS. This can involve *reasoning*, *retrieval*, but also *other tasks* enabled by ontologies. In order to support these ontology-based services, the architecture contains the following lifecycle usage services that are rather independent from specific use cases.

Services that can automatically *populate* the KB reduce the effort needed for the manual creation of instance data. These services are performed by agents that request external ontology data as well knowledge extractors that crawl external non-ontological sources. They implement learning algorithms to extract instances from text and multimedia contents. Some of these population services (and ontology learning services) may incorporate procedures for natural language processing [12] as subcomponents.

Finally, the quality of the acquired instance data has to be ensured. *Cleansing* services are available to adapt the format and labels to the application requirements. The same instances extracted from different sources may have different labels. Knowledge *fusion* services identify and resolve such occurrences. Similarly, knowledge acquired

from different sources may be redundant and often contradictory. This is also addressed by the fusion services. These services may implement a semi-automatic process, which involves the user and engineering services such as debugging and editing. The arrows in Fig. 2 illustrate this interaction between usage and engineering services. It is provided by the evolution support, a feature part of aforementioned usage services, which possibly require interaction with engineering services.

### 3.3 Designing OIS with the Generic Architecture

We now discuss how this architecture can act as a reference that can be adapted to match functional and technological requirements of a particular OIS.

**Matching Functional Requirements:** The presented architecture is very generic and targets the management of the entire ontology lifecycle. Implementing the whole architecture would result in a fully-fledged integrated system that supports both the engineering and the application of ontologies. However, a particular application often requires only a subset of the envisaged services.

Applications may feature only engineering, or only usage of ontologies that already have been engineered using another system. Then only engineering and usage services, respectively, have to be incorporated into the concrete architecture of the final application. In general, the functional requirements of the system have to be analyzed. Then these requirements have to be mapped to services of the architecture. Finally, for each of the identified services, more fine-grained functionalities have to be derived w.r.t. the use cases to be supported by the application.

For instance, an application that only uses RDF(S) ontologies may not need any lifecycle services at all. Imagine a web application, which simply presents FOAF profiles manually imported from external sources. Then only core ontology services are needed to import, store and retrieve information from the profiles. A more sophisticated version may employ agents to crawl profiles from the web. Even then, only population and basic cleansing is needed, because due to the use of RDF(S), no inconsistencies can arise that would require engineering services. Now, imagine an application using OWL ontologies to manage resources of a digital library. Resources are annotated with ontology concepts that can be defined by the user. Most annotations are extracted automatically and even new concept descriptions are suggested by the system to capture the knowledge contained in new library resources. Clearly, this application would need a wide range of usage and engineering services and hence, an integrated application with lifecycle support.

**Matching Technological Requirements:** The presented architecture is of abstract nature and free of assumptions about specific technological settings. For the development of a specific application, it can be used as a reference to identify the components (as discussed previously) and to organize them with the suggested abstraction layers and control-flow. Then, given specific technological constraints, a concrete architecture paradigm can be chosen and applied to the abstract architecture. These paradigms capture best practices in different application settings and can also give additional guidance

for OIS engineering. We will now outline standard paradigms in software engineering and discuss for which exemplary settings they are most appropriate.

Architecture paradigms can be distinguished along three dimensions, namely the degree of distribution, coupling and granularity of components. Distribution can range from non-distributed rich client, over client-server, three-tier [13], multi-tier to fully-distributed P2P architectures. The last two dimensions make up the differences of two more concrete architecture paradigms with specific platform assumptions, namely the component-oriented multi-tier J2EE architecture [6] and the Service-oriented Architecture (SOA) [5]. While J2EE comprises of tightly-coupled and relatively fine-grained components, SOA advocate the use of loosely-coupled and coarse-grained services.

The main idea behind multi-tier architectures is the encapsulation of each tier, meaning any tier can be upgraded or replaced without affecting the other tiers. While this organization principle has been adopted (where layer stands for tier), the proposed architecture does not make any assumptions about how components may be distributed. In fact, the layered organization can be seen as an orthogonal principle that can be combined with any of the mentioned paradigms.

For instance, elements of the architecture can be implemented as components of a desktop application, e.g. the backend maps to a file system, services and control-flow map to Plain Old Java Objects (POJOs) and their call hierarchy and GUI components map to Swing widgets. In another use case, more flexible access may be required, the application logic may call for more processing capabilities, and the amount of data cannot be managed efficiently by a file system. Then, a database can be employed as backend, data access can be provided by Data Access Objects (DAO) and lifecycle services are realized as Enterprise Java Beans (EJB) of a J2EE platform, and front-ends are implemented as Java Server Pages (JSP) to deliver contents over the web. In some cases lifecycle components could be tightly integrated with other internal systems via J2EE connectors [9]. In other cases external parties may want to choose from different offerings and therefore demand a more flexible way to discover ontology services at runtime and to interact with them on the basis of a standardized protocol. Here, SOA may be the choice: The fine-grained functionalities of some lifecycle components are encapsulated in form of coarse-grained services exposed to consumers via WSDL and SOAP. Instead of using a completely new SOA platform, one may go a more evolutionary way advocated by major J2EE vendors, i.e. switch to a Service Component Architecture (SCA)<sup>5</sup> that implements SOA. SCA provide guidelines for decoupling service implementation and service assembly from the details of underlying infrastructure capabilities. Components can then offer their functionalities as services that can also be consumed externally. However for internal consumption, they do not necessarily have to be loosely coupled—since tight coupling can avoid the overhead of creating, parsing and transporting messages over the network.

In all, the generic architecture gives guidelines for the identification and organization of components. The examples above illustrate that there are many other aspects that have to be considered given concrete requirements. After the choice for a concrete platform and the paradigm to be applied on the architecture, guidance can then be found in the respective reference architectures, e.g. see [9] for J2EE, [5] for SOA and SCA.

---

<sup>5</sup> <http://www.osoa.org/display/Main/Service+Component+Architecture+Home>

## 4 Case Study – An Instantiation of the Generic Architecture

In this section, we discuss the application of the generic architecture w.r.t. a concrete case study of the NeOn project<sup>6</sup> at FAO (United Nations Food and Agriculture Organization). Within this case study, we are developing an ontology-based information system to facilitate the assessment of fisheries stock depletion by integrating the variety of information sources available. The FAO Fisheries department manages statistical data on fishing, GIS data, information on aquaculture, geographic entities, description of fish stocks, etc. Although much of the data are ‘structured’, they are not necessarily interoperable. In addition, there are information resources that are not available through databases but as parts of websites, or as documents, etc. These data sources could be better exploited by bringing together related and relevant information with the use of ontologies, to provide inference-based services, enabling policy makers and national governments to make informed decisions. In this context, the goal of the case study is to implement an ontology-based Fishery Stock Depletion Assessment System (FSDAS) as well as an application to manage the fishery ontologies and their lifecycle. In the following, we follow the methodological guidelines of the previous section: We illustrate how to match the functional requirements by analyzing the use cases w.r.t. the individual phases of the ontology lifecycle, and finally show how we match technical requirements in the realization of these use cases as two particular configurations of the NeOn toolkit.

### 4.1 Uses Cases Within the Lifecycle of the Fishery Ontology

In the ontology lifecycle of the case study we find a clear separation between the ontology engineering and ontology usage phases. In fact, we find two different sets of users that are involved in ontology engineering and ontology usage. We will now discuss selected use cases in the lifecycle that provide functional requirements that need to be covered in the architecture of the system.

**Ontology Engineering:** The ontology engineering environment needs to put mechanisms in place to allow all actors involved in the process to create and maintain distributed networked ontologies and ontology mappings in the fishery knowledge community. These mechanisms require many of the generic ontology engineering services discussed in the previous chapter.

There are several actors involved in the engineering phase of the fishery ontology lifecycle [14], including experts in ontology modeling, ontology editors, and subject matter experts. Each of the actors needs to be supported in different use cases of the ontology development. Ontology development follows a well defined collaborative workflow, which needs to be supported in the engineering environment. Further, contextualized visualization and editing modes—depending on the actor and the task to be performed—are important for the usability and effectiveness of the engineering environment.

Due to the scale and heterogeneity of the various information sources, there must be an easy way to create mappings between ontologies in a manual and semi-automatic way. Modularization of ontologies—i.e. creation of modules manually and semi-automatically as well as merging modules—must also be taken into account. The

---

<sup>6</sup> <http://www.neon-project.org/web-content/>

generation of ontologies from textual sources is another key issue. Given a textual corpora the system must provide a list of candidate elements of the ontology (classes, instances and relations between concepts), showing the documents and excerpts supporting the extracted terminology.

Before publishing a new version of an ontology, debugging and evaluation must be performed. This involves checking for logical consistency, making comparisons with other ontologies, and evaluating structural and functional properties of the ontology.

**Ontology Usage:** The fishery ontologies are used within the FSDAS system to assist the users—i.e. fishery experts in the FAO Fishery department—in gathering, analyzing and producing information on the status and trends of fish stock. For example, a fishery expert may want to research why the stock of tuna is depleting in the Mediterranean Sea. For this purpose, the ontology-based FSDAS allows authorized users to browse and query a knowledge base of fisheries digital resources.

The major use case within FSDAS is to perform ad-hoc queries, using both free-text and ontology elements such as concepts and relations, against the fishery data sources. In this context, matching of keywords and phrases to ontology elements, assistance on query formulation and query refinement are supported by the system. The second major use case related to information access is browsing and navigating the fishery data sources, e.g. using the ontology to find related data instances for a given concept, i.e. analyzing stock depletion. In interacting with the FSDAS, users are able to maintain a profile and store favorite queries.

At runtime, the FSDAS directly integrates dynamic resources to populate the fishery ontology with data instances, e.g. by crawling remote system websites, etc. Additionally, users have the ability to annotate data instances with the FSDAS system.

Finally, the users of the FSDAS system have the ability to comment the use of the application and in particular propose modifications to the ontology. This information is fed back into the ontology development phase to support the evolution of the fishery ontology. Once a new version of the fishery ontology has been approved, it can be deployed in the runtime FSDAS application, closing the loop of the ontology lifecycle.

## 4.2 Instantiation of the Generic Architecture

We now discuss how we address technical requirements in the realization of the case study using the NeOn toolkit [15] as an implementation of our generic architecture. The NeOn toolkit provides an infrastructure and software components for both the engineering and the runtime usage of ontologies. For the case study, two applications are developed, one for the engineering of the fishery ontologies and one for the FSDAS runtime system. They are used by different sets of end users—i.e. ontology engineers and fishery experts—with different technological requirements. They are realized as two particular configurations of the NeOn toolkit, i.e. realized using a subset of components provided by the toolkit. In particular, both applications are built on a shared infrastructure, shared data sources and shared core ontology services. However, whereas one is configured with a bundle of ontology engineering services, the other relies completely on usage services—as shown in Figure 3.

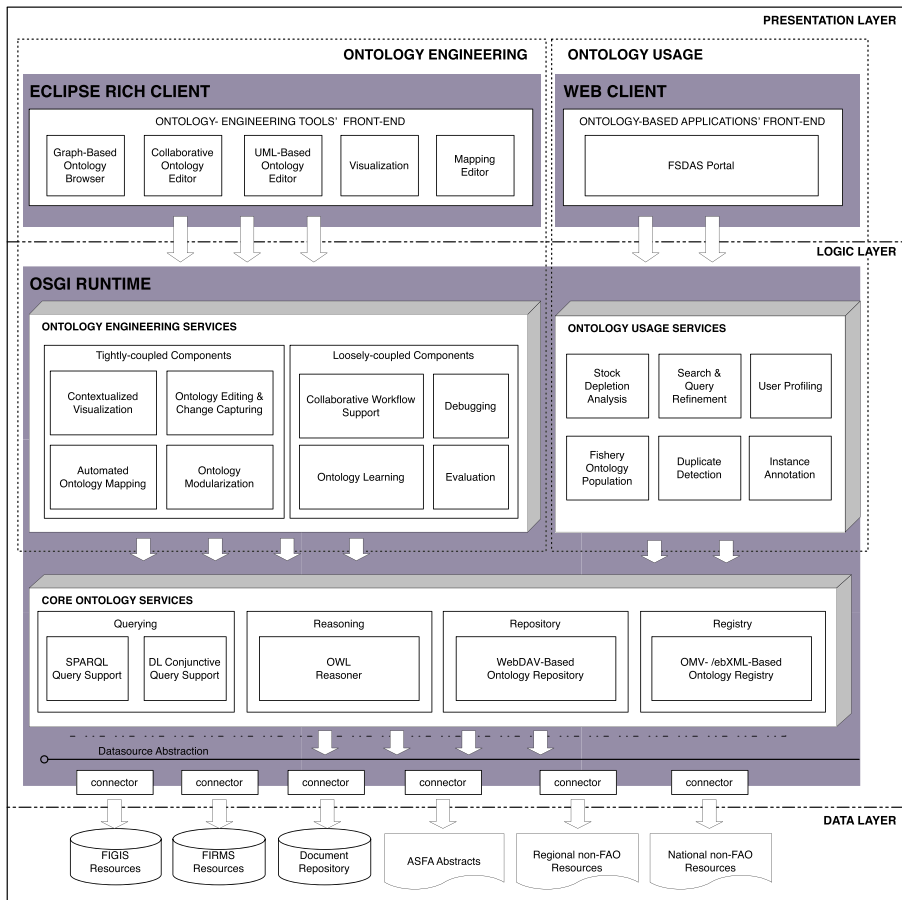


Fig. 3. Case Study Architecture based on the NeOn Platform

The *Core Ontology Services* shared across the engineering and runtime environments are based on the NeOn ontology model API—an API that supports the management of rule extended OWL ontologies and ontology mappings [16]—and include services for reasoning and querying, an ontology repository and registry etc. Based on the core ontology services, there are higher-level services that cover the ontology usage and engineering use cases discussed above.

The shared infrastructure of core services is based on OSGi, an open Java-based platform<sup>7</sup> that is the foundation of a service oriented architecture. Besides many other standard platform services, OSGi defines a lifecycle model and a service registry, that allows for the dynamic interaction of services. The standard implementation used within

<sup>7</sup> <http://www.osgi.org/>



the NeOn toolkit is Eclipse Equinox<sup>8</sup>. The OSGi platform is designed to support both *distributed client-server configurations* and *desktop configurations* via Eclipse/OSGi. In fact, for the realization of the case study, we implement the ontology engineering environment using a desktop configuration, i.e. an Eclipse rich-client application, while the FSDAS is realized using a distributed configuration where user interfaces of the application are web-based. The reason for this lies in the technical requirements of the two different user groups, where the ontology engineers require a rich tool set on their local desktops, whereas the fishery experts as non-technical users want to work with light-weight applications in web environments.

In the ontology engineering environment, we distinguish between tightly and loosely coupled components: Loosely coupled components are non-interactive, large grain, potentially remotely used components. They are integrated as Web Services. In contrast, tightly coupled components are highly interactive, fine grained, locally used components. Every engineering use case discussed previously is addressed by a component consisting of one or more Eclipse plugins. The modularity via the plugin concept of Eclipse follows the philosophy that “everything is a plugin”. Using this plugin concept, components for ontology editing, visualization and other functionalities for the different actors in the engineering process through the integration of various components provided by the NeOn toolkit. As an example, the collaborative ontology engineering is supported by a number of services that are provided by loosely and tightly coupled components: The local ontology editing is tightly integrated with the change capturing, while certain services to support the collaborative workflow such as conflict resolution as well as validation and evaluation services are loosely coupled as web services.

The FSDAS application is realized as a distributed configuration, which combines OSGi with server-based technology. In our configuration, this is achieved by embedding web-server technology in the OSGi runtime platform, making the ontology usage services accessible in the web-based FSDAS portal.

## 5 Conclusion

In order to provide guidance for the development of ontology-based information systems, we have developed an integrated architecture that takes the complete ontology lifecycle into account. As we have illustrated, such an architecture is required to address use cases where ontology usage and engineering are intertwined at runtime, resulting in a dynamic feedback loop. This loop and the lifecycle activities act as functional requirements, which are addressed in our proposal for a generic architecture for ontology-based information systems.

We have discussed how to adapt this architecture to functional requirements of specific use cases, from simple ontology applications to systems for integrated ontology engineering and management. To demonstrate the value of our architecture, we have shown its application in a concrete case study, using the NeOn toolkit. This toolkit is a concrete implementation of our generic architecture, integrating reusable lifecycle components from and for the community that can facilitate the adoption of semantic technologies. In the future, we will further add components to the toolkit as well as promote

---

<sup>8</sup> <http://eclipse.org/equinox/>



the integration of tools externally developed by the community, either as tightly-coupled components or loosely-coupled services. We expect the toolkit to evolve to a more complete set of reusable components which—combined with the generic architecture—can serve as guidelines for the design and can be leveraged for the implementation of many other OIS with lifecycle support.

## Acknowledgements

This work was partially supported by the European Commission in the IST projects NeOn (IST-2006-027595) and X-Media (IST-FP6-026978).

## References

1. Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W., Wielinga, B.: *Knowledge Engineering and Management: The CommonKADS Methodology*. MIT Press, Redmond, Washington (1999)
2. Haller, A., Cimpian, E., Mocan, A., Oren, E., Bussler, C.: WSMX - A Semantic Service-Oriented Architecture. In: ICWS 2005. Proceedings of the IEEE International Conference on Web Services, pp. 321–328. IEEE Computer Society Press, Los Alamitos (2005)
3. Gómez-Pérez, A., González-Cabero, R., Lama, M.: ODE SWS: A Framework for Designing and Composing Semantic Web Services. *IEEE Intelligent Systems* 19, 24–31 (2004)
4. García-Castro, R., Suárez-Figueroa, M.C., Gómez-Pérez, A., Maynard, D., Costache, S., Palma, R., Euzenat, J., Lécué, F., Léger, A., Vitvar, T., Zaremba, M., Zyskowski, D., Kaczmarek, M., Dzbor, M., Hartmann, J., Dasiopoulou, S.: *Architecture of the Semantic Web Framework*. Technical Report D1.2.4, Knowledge Web Consortium (2007)
5. MacKenzie, C.M., Laskey, K., McCabe, F., Brown, P.F., Metz, R.: *OASIS Reference Model for Service Oriented Architecture v1.0*, OASIS Official Committee Specification, approved August 2006 (2006)
6. Singh, I., Stearns, B., Johnson, M.: *Enterprise Team: Designing Enterprise Applications with the J2EE (tm) Platform*. In: *The Java Series*, Addison-Wesley Longman Publishing Co., Inc, Boston, MA, USA (2002)
7. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: *Ontological Engineering*. In: *Advanced Information and Knowledge Processing*, Springer, Heidelberg (2003)
8. Wang, Y., Bao, J., Haase, P., Qi, G.: Evaluating formalisms for modular ontologies in distributed information systems. In: RR 2007. Proc. of The First International Conference on Web Reasoning and Rule Systems, pp. 178–182. Springer, Heidelberg (2007)
9. Sharma, R., Stearns, B., Ng, T.: *J2EE (tm) Connector Architecture and Enterprise Application Integration*. Wesley Longman Publishing, Boston, MA, USA (2002)
10. Wu, Z., Chen, H., Wang, H., Wang, Y., Mao, Y., Tang, J., Zhou, C.: Dartgrid: a semantic web toolkit for integrating heterogeneous relational databases. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006*. LNCS, vol. 4273, Springer, Heidelberg (2006)
11. Haase, P., van Harmelen, F., Huang, Z., Stuckenschmidt, H., Sure, Y.: A framework for handling inconsistency in changing ontologies. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005*. LNCS, vol. 3729, pp. 353–367. Springer, Heidelberg (2005)
12. Valarakos, A., Paliouras, G., Karkaletsis, V., Vouros, G.: Enhancing Ontological Knowledge through Ontology Population and Enrichment. In: Motta, E., Shadbolt, N.R., Stutt, A., Gibbins, N. (eds.) *EKAW 2004*. LNCS (LNAI), vol. 3257, pp. 144–156. Springer, Heidelberg (2004)

13. Eckerson, W.W.: Three Tier Client/Server Architecture: Achieving Scalability, Performance, and Efficiency in Client Server Applications. *Open Information Systems* 10 (1995)
14. Iglesias, M., Caracciolo, C., Jaques, Y.: NeOn Deliverable D7.1.1 Specification of user requirements in the fishery case study. Technical Report D7.1.1, NeOn Consortium (2007)
15. Waterfeld, W., Weiten, M., Haase, P.: NeOn Deliverable D6.2.1 Specification of NeOn reference architecture and NeOn APIs. Technical Report D6.2.1, NeOn Consortium (2007)
16. Haase, P., Rudolph, S., Wang, Y., Brockmans, S., Palma, R., Euzenat, J.: d' Aquin, M.: Networked Ontology Model. Technical Report D1.1.1, NeOn Consortium (2006)

# Ontology-Based Interpretation of Keywords for Semantic Search

Thanh Tran, Philipp Cimiano, Sebastian Rudolph, and Rudi Studer

Institute AIFB, Universität Karlsruhe, Germany  
{dtr,pci,sru,rst}@aifb.uni-karlsruhe.de

**Abstract.** Current information retrieval (IR) approaches do not formally capture the explicit meaning of a keyword query but provide a comfortable way for the user to specify information needs on the basis of keywords. Ontology-based approaches allow for sophisticated semantic search but impose a query syntax more difficult to handle. In this paper, we present an approach for translating keyword queries to DL conjunctive queries using background knowledge available in ontologies. We present an implementation which shows that this interpretation of keywords can then be used for both exploration of asserted knowledge and for a semantics-based declarative query answering process. We also present an evaluation of our system and a discussion of the limitations of the approach with respect to our underlying assumptions which directly points to issues for future work.

## 1 Introduction

Part of the Semantic Web vision is to provide web-scale access to semantically described content. In particular, this implies understanding users' information needs accurately enough to allow for retrieving a precise answer using semantic technologies. Currently, most web search engines are however based on purely statistical techniques. While they are not able to figure out the meaning of a query, they can provide answers by returning the statistically most appropriate answer to a user's query—based on some measures for computing similarity in vector space (cf. [1]). Information Retrieval (IR) techniques applied to the Web have gained a reasonable degree of maturity which is clearly corroborated by the success of search engines such as Google, Yahoo and the like. These search engines are in fact providing a baseline quite difficult to outperform. Due to the nature and the maturity of the underlying statistical techniques, they are more robust and scale to the size of the Web, as opposed to semantic technologies.

For restricted domains which can be formalized using ontologies, there is nevertheless hope that semantic technologies can be put into work to allow for more semantics-based search. One of the crucial steps within such an endeavor is to precisely capture the user's information need (see also [2]). But how does the user express his information need? If we look at the wide-spread usage of web search engines, we can conclude that users are definitely used to express their information need via simple queries based on keywords. However, while there is substantial recent work on interpreting full natural language questions semantically w.r.t. an ontology (cf. [3], [4]) or database schema [5], not as much work has been carried out with respect to the formal interpretation of keyword queries. A notable exception is the approach described in [6], which we discuss further in the related work section.

In this paper, we present an approach for interpreting keyword queries using background knowledge available in ontologies. Based on a few assumptions about how people describe their information needs, we present an approach which translates a keyword query into a DL conjunctive query which can be evaluated with respect to an underlying knowledge base (KB). The evaluation of our approach has been carried out on the KB of the semantic portal at <http://www.aifb.uni-karlsruhe.de/> and shows first promising results which we discuss w.r.t to our underlying assumptions. In addition, we present a system which shows how the interpretation of keywords can be used for a combination of intuitive exploration and search in KBs.

The paper is structured as follows: we begin with a discussion of related work in Section 2. Then, a generic approach for the interpretation of queries with background knowledge is presented in Section 3, followed by a detailed description of the translation of keyword queries to DL conjunctive queries in Section 4. Then in Section 5, we present the implementation of the approach as well as its evaluation. A discussion of the results points us directly to open issues for future work. We conclude in Section 6.

## 2 Related Work

Recently, substantial work has been performed on the translation of natural language questions to formal queries using an ontology or a database (cf. [5], [3], [4], [7]). While these approaches have been shown to yield remarkable results, it is not clear if users always want to specify a full natural language question. In fact, the success of commercial search engines shows that users are quite comfortable with using keywords. Thus, it seems important to also develop approaches which are able to interpret keywords such that they can be answered through a query to a database or a KB.

In this regard, there exists work on the translation of keywords to XML-based queries, e.g. to interpret keywords as X-Queries on XML data [8]. This is related to our approach because also the structure of (XML) elements is considered to interpret the relations among keywords. However the structure exploited there is less complex than the many relations among entities given by ontology axioms that we explore for our translation. Also, there has already been work on translating keywords to semantic queries. For instance, Royo et al. propose to map keywords to corresponding WordNet synsets [9]. While they claim to also be able to discover relations between keywords, it is not clear how this is achieved, especially given the fact that WordNet does not include any non-taxonomic relations besides part-of relations.

The approach closest to ours is the SemSearch approach presented by Lei et al. [6]. In fact, we agree with the analysis of Lei et al. that common approaches to semantic search are not particularly intuitive or user friendly as they either require posing formal (logical) queries or limit the expressive power of the user by using forms for example (compare the analysis of the semantic search state-of-the-art in [6]). Our approach is similar to SemSearch in the sense that we also aim at answering complex keyword queries by translating them into a logical query. However, our approach mainly differs in the way the query is computed. In SemSearch, the keywords are first interpreted as either instances, concepts or properties, respectively, which yields nine possible templates to be instantiated for the case of queries consisting of two keywords. Templates in fact fix

the structure of the resulting query a priori, i.e. it is assumed that entities denoted by keywords can be connected through a direct relation in the ontology. As queries with more than two keywords lead to a combinatorial explosion of the different possible combinations of entities, and thus would require a large number of templates, some heuristics are suggested to handle these complex queries (see [6] for more details). In contrast to SemSearch, we build on a more generic graph-based approach to explore the connections between the entities in the query. Our approach does not fix the structure of the queries in the form of templates a priori and does not assume the availability of direct connections between entities. In fact, the vicinity of the entities that is to be explored is based on a variable  $d$ , which can be set by the user. Within this range, many possibly indirect connections might be discovered and used for the generation of the formal query.

### 3 Answering User Queries in Ontology-Based Systems

In this section, we present an abstract framework describing the process of ontology-based IR, where the user poses a question to the system and the system answers the question using knowledge formalized in a logical language. In particular, we focus on scenarios where the language of the user question does not match the query language supported by the system. For this purpose, we define our ontology-based IR process as consisting of four models and describe the assumptions underlying our approach. We then present a generic approach for translating a user question into a formal system query.

#### 3.1 Models in Ontology-Based Information Retrieval

In line with models in classical IR, namely the query and resource model [1], we discuss four different models involved in ontology-based IR.

**The Mental Model  $O_U$ :** The mental model  $O_U$  corresponds to the *information need* that a user has in mind at the beginning of an IR task. Since the concrete mechanisms underlying human thought are far from completely understood, for the sake of the approach presented in this paper we postulate only very abstract properties of this model:  $O_U$  can be conceived as a set of (thought) entities that are relevant for the current information need and embedded in an association structure. These entities might be related to real world objects or to more abstract concepts. The entities in this association structure can be conceived as what the user knows. We assume that the user is looking for (some of the) entities missing in this structure, which we refer to as *gaps*.

**The User Question Model  $Q_U$ :** The user question model  $Q_U$  consists of elements, which in turn are constructed out of language primitives  $\mathcal{P}_U$  of a language  $\mathcal{L}_U$  (language of the user). This model is the result of the user translating elements in  $O_U$  to elements in  $\mathcal{P}_U$ . Moreover (depending on the expressive means of  $\mathcal{L}_U$ ), there might be also elements in  $Q_U$  explicitly denoting gaps (like, e.g. question words in a natural language). Naturally,  $Q_U$  must not be empty.

**The System Resource Model  $O_S$ :** This model consists of elements constructed out of language primitives  $\mathcal{P}_S$  of a formal KR language  $\mathcal{L}_S$  (language of the system). Independent from a concrete formal language used, these elements can be conceived

as a set of entities of a given ontology. As opposed to the abstract mental model  $O_U$ , the entities and structure of  $O_S$  are explicitly given and directly accessible. These elements constitute the knowledge (the KB) the system uses to answer the user question.

**The System Query Model  $Q_S$ :** This model represents the final question processed by the formal query engine of the system. It consists of elements constructed out of language primitives  $\mathcal{P}'_S$  of a query language  $\mathcal{L}'_S$ . When there is a formal semantics for  $\mathcal{L}'_S$  (query language of the system) it must be compatible with the semantics of  $\mathcal{L}_S$  for the query  $Q_S$  to be processable by the system. In particular, some elements in  $Q_S$  must correspond to elements in  $O_S$ . In fact, formal queries in many systems are specified using ontology elements of the underlying KR language. However, the query language  $\mathcal{L}'_S$  may have primitives additional to the ones available in  $\mathcal{L}_S$ . In particular, there must be primitives to specify the gaps, e.g. variables.

Note the correspondence of these models and the consequences for ontology-based IR: The more the entities and structure in  $O_U$  match the entities and structure in  $O_S$ , the higher the chance that  $O_S$  can be used by the system to fill the gaps, i.e. to answer the query. Also, the more related the syntax and semantics of  $\mathcal{L}_U$  and  $\mathcal{L}_S$ , the more straightforward is the mapping from  $Q_U$  to  $Q_S$ , i.e., the interpretation of the user query. Yet, in the following, we will restrict our attention to scenarios where the query language of the user  $Q_U$  and the language of the system  $Q_S$  differ considerably and propose to use an ontology-based system to interpret and answer the user question.

### 3.2 A Generic Approach for Ontology-Based Query Interpretation

In this section, we are not concerned with the actual answering step where the query engine processes the system query. Instead, we present a generic approach to deal with the preceding step, namely translating the user question to the system query. Similar to query processing, we propose an approach which relies on the knowledge in the KB for question interpretation. We will start with the clarification of our assumptions before the presentation of our approach.

**Assumption (A1) — Ontology-Mental Correspondence:** This assumption requires both an entity-wise and a structural correspondence between the mental model  $O_U$  and the system resource model  $O_S$ . That is, elements and the associative structure in  $O_U$  correspond to ontology entities and the structure in  $O_S$ , respectively.

**Assumption (A2) — Locality of Information Need:** This assumption requires those ontology entities  $O'_S \subseteq O_S$  that correspond to entities in the mental information need representation  $O_U$  to be connected over a maximum distance  $d$ . That is, for any two ontology entities  $a, b \in O'_S$  there has to be a direct connection  $\langle a, b \rangle$  or a sequence of  $x_i$  such that  $a = x_0$  and  $\langle x_0, x_1 \rangle, \langle x_1, x_2 \rangle, \dots, \langle x_{n-1}, x_n \rangle$ , and  $\langle x_n, b \rangle$  and  $n < d$ . There might be several such sequences that connect gaps with the two entities the user knows  $(a, b)$ . In such cases, we assume not only that there is a maximum distance but moreover that connections over smaller distances are more likely to contain the gaps that the user looks for.

The above assumptions are certainly too strict in the sense that users can not be assumed to fully think in term of ontological structures or in any KR language. However, we need to assume that they think in some structures which can be mapped to an

ontology. Otherwise, a system would have no chance in interpreting and answering a user's query. In this sense, our assumptions seem justified from a practical point of view. Thus, if there is no such correspondence, the system cannot fill the user gaps, i.e. answer the query. In addition, A2 helps to restrict attention to only a particular part of the ontology, as discussed in our approach presented in the following.

**Interpreting the User Question:** We present a generic approach to translate  $Q_U$  to  $Q_S$  which consists of three high level steps. First, the elements in the user question  $Q_U$  are mapped to ontology elements from  $O_S$  (Step 1). Then, further ontology elements are explored to better cover the initial information need in the mental model  $O_U$  (Step 2). Finally, from this more refined ontological representation of the need, the query  $Q_S$  will be derived (Step 3).

In step one, we make use of the correspondence stated in A1 and map elements of the user queries  $Q_U$  to ontology elements  $O'_S \subseteq O_S$ . Note that the user question may only partially capture the mental model. Also, not all elements of the user question can be mapped to corresponding ontology elements. Therefore, the identified ontology elements  $O'_S$  yet do not account for the entire mental model. Since we want in some way to "reconstruct" the mental model and find out the gaps, further computation is required in these cases to find missing elements.

In step two, the assumption on the locality of information need (A2) is used to explore connections among  $O'_S$  identified in step one using further elements in  $O_S$ . Due to A2, only elements in  $O_S$  that are connected with the identified elements  $O'_S$  within a specified range (maximum distance  $d$ ) have to be considered in the exploration. From this it also follows that after all the neighboring elements in this range have been explored for all  $O'_S$ , the discovered elements in  $O_S$  combined with  $O'_S$  can be assumed to approximate the user's mental model.

After reconstructing this mental model, the identified and discovered ontology elements need to be assembled into a formal query in the language  $\mathcal{L}'_S$ . The discussion on  $Q_S$  already pointed out that formal queries are specified using ontology elements (the *information part*). Additionally, they contain variables (the *question part*). As opposed to  $Q_S$ ,  $O_S$  does not contain variables, and thus, the identified elements  $O'_S$  map to the information part. Note that in the exploration step, the discovered elements may correspond to thought entities the user knows but has not explicitly specified in the question. Also, they might correspond to gaps, i.e. entities the user does not know and out of which only some might be interesting to him/her. While all the others map to the information part, the elements corresponding to the answer the user looks for map to variables of the question part.

**Illustrating Example:** We illustrate our approach with a simple example as depicted in Figure 11 where a user wants to retrieve all publications authored by Philipp Cimiano which are associated to the project X-Media. Let's assume that the user, on the basis of his information need, issues the query  $Q_U =$  "Philipp Cimiano X-Media publications".

The elements in the query are then mapped to the ontology elements Philipp Cimiano, X-Media and publication respectively. These elements, however, yet do not fully correspond to the information need of the user. Also, they still cannot be assembled into a system query that yields answers the user looks for due to missing



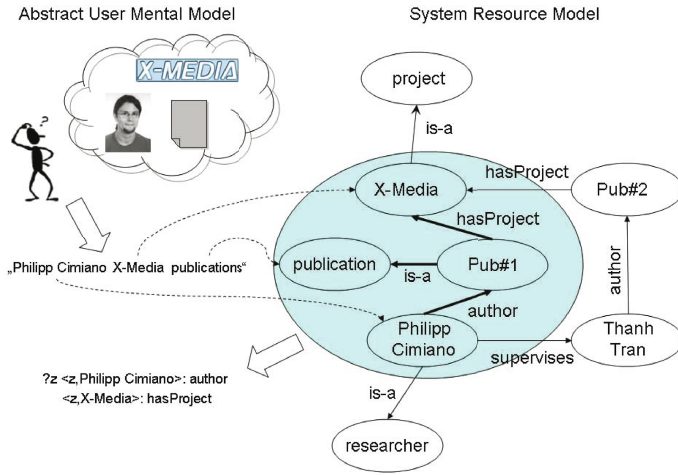


Fig. 1. Workflow for the query "Philipp Cimiano X-Media publications"

elements. These missing elements correspond to the entities in the mental model the user knows but does not specify such as the connection *is-a* between *X-Media* and *project*. In particular, the user does not make explicit the relation between Philipp Cimiano and *publication* connected in the ontology via the *author* relation. These missing elements correspond to what we call *gaps*, i.e. elements in the mental model that the user does not make explicit when specifying his/her information need. In our case, the user does not make explicit the connection *hasProject* between *X-Media* and *publication* while for sure s/he was thinking of it. Some of these gaps correspond to the information the user is looking for, i.e. *Pub#1* in our example. All these missing elements need to be made explicit in our translation into a formal query.

For this purpose, in step 2, our approach starts the KB exploration from the individual Philipp Cimiano and leads to the relations *author* and *is-a*. Assuming the exploration width is 2, we also reach the elements *Pub#1* and *researcher* from Philipp Cimiano. From the other elements in the query, i.e. *X-Media* and *publication*, we reach the relations *hasProject*, *is-a* as well as the elements *project*, *Pub#1* and *Pub#2* from *X-Media*, and *is-a* and *Pub#1* from *publication*. This shows how step-by-step the exploration builds up a graph where all elements of the initial user query are connected.

In step 3, the (possibly many) subgraphs which connect these elements are computed. These subgraphs correspond to the different questions the user possibly has. As highlighted in the circle in Figure 1 in our specific example there is only one such subgraph. However, in other scenarios, and in particular if the exploration range *d* is set higher, we are likely to obtain several such subgraphs. In such cases, A2 allows to rank queries, since it postulates that connections over smaller distances are more likely to contain the answer the looks for. Finally, the graph is translated into a corresponding query, e.g.  $Q_S = \langle x, \text{Philipp Cimiano} \rangle : \text{name} \wedge \langle x, y \rangle : \text{author} \wedge \langle y, z \rangle : \text{hasProject} \wedge \langle z, \text{X-Media} \rangle : \text{name} \wedge \langle y : \text{publication} \rangle$ . While the previous steps are rather generic, the mapping from the graph elements to the information part and question part of the



system queries depends on the query syntax of  $Q_S$  as well as the specific elements identified and explored in  $O_S$ .

This simple example demonstrates the high level steps as captured in the generic approach. It shall facilitate comprehension of more technical details of a procedure we propose for the specific translation of keyword queries to DL conjunctive queries presented in the following section.

## 4 Interpretation of Keywords Using DL Knowledge Bases

In this section, we present an instantiation of the generic approach described above to two specific languages  $Q_U$  and  $Q_S$ .  $Q_U$  is grounded to keyword queries, i.e.  $Q_U = (k_1, k_2, \dots, k_n)$  where the  $k_i$ 's stand for keyword and represent the primitives  $\mathcal{P}_U$ . The language of the user  $\mathcal{L}_U$  then simply consists in concatenations of the elements in  $\mathcal{L}_U$ . Thus, by keyword queries, we mean the standard type of queries supported by Google-style interfaces like the ones discussed in [10]. Further,  $Q_S$  is grounded to DL conjunctive queries. Such a query is defined as a conjunction of terms of the form  $x : C$  or  $\langle x, y \rangle : R$ , where  $C$  is a concept,  $R$  is a role, and  $x, y$  are variables or individuals taken from  $\mathcal{V}$  a set of variable names, or  $\mathcal{I}$  a set of individual names. If we conceive the variables as individuals, these terms are assertional statements of a DL language, where the first kind is referred to as concept terms and the latter kind is called role terms.

For the translation of keyword queries to DL conjunctive queries, we make use of  $O_S$ , a KB containing knowledge formalized in the form of DL axioms. In particular, the description logic in our approach is  $\mathit{SHOIN}(\mathbf{D})$ , the DL counterpart to OWL DL, such that, in addition to individuals and variables in query terms, we also have  $j : D$ , where  $j$  are data values taken from the set of values  $\mathcal{J}$  and  $D \in \mathcal{D}$  is the set of data ranges. Moreover, roles can be further divided into abstract roles (object properties)  $R$  and concrete roles (datatype properties)  $U$  such that possible terms occurring in a conjunctive query have the shape  $x : C$ ,  $j : D$ ,  $\langle x, y \rangle : R$  and  $\langle x, j \rangle : U$ .

Before the detailed presentation of the approach, we discuss the specialization of A1 to the particular setting described above, i.e. the correspondence of the mental model and the DL knowledge base.

**Assumption 1' (A1').** We assume that users' mental models are organized in a way similar to DL knowledge bases. More precisely, this means that the thought entities of the mental model  $O_U$  correspond to  $\mathit{SHOIN}(\mathbf{D})$  ontology entities in the disjoint union of the sets  $\mathcal{I}$  (individuals),  $\mathcal{J}$  (data values),  $\mathcal{C}$  (concepts),  $\mathcal{D}$  (data ranges),  $\mathcal{R}$  (object properties), and  $\mathcal{U}$  (data properties) and the associations in  $O_U$  correspond to associative interconnections of the types  $\langle i, C \rangle$ ,  $\langle i_1, R, i_2 \rangle$  and  $\langle i, U, j \rangle$  where  $i, i_1, i_2 \in \mathcal{I}$ ,  $j \in \mathcal{J}$ ,  $C \in \mathcal{C}$ ,  $R \in \mathcal{R}$ , and  $U \in \mathcal{U}$ . As given by the  $\mathit{SHOIN}(\mathbf{D})$  syntax, such connections are specified using the DL-axioms  $i \in C$  (concept membership),  $\langle i_1, i_2 \rangle \in R$  (object property membership) and  $\langle i, j \rangle \in U$  (data property membership).

Note that when compared to A1, A1' imposes stricter structural properties on the mental model. Namely, its structure is frame-based in the sense that elements of the mental model correspond to the entities and relations of a DL A-Box. We think that as the frame-based nature of DL seems to be an intuitive formalism to describe knowledge,

it might be also an intuitive way for users to think about (and to describe) the knowledge they are looking for. In what follows we describe the various steps of the concrete instantiation of the generic approach in more details.

#### 4.1 Step One — Mapping Terms to KB Entities

Due to A1', we assume that keywords are mapped to ontology entities, namely individuals, data values, concepts, data ranges, as well as object and data properties. In particular, the mapping can be defined as a function  $f$  which maps elements of the user question model  $Q_U$  to entities of system resource model  $O_S$ , i.e.  $f : Q_U \rightarrow O_S$ . For practical purposes, it is crucial that this function is "robust" in the sense that it also considers syntactic and spelling variants.

Using the query engine, entities in the KB can be retrieved via their URIs. In particular,  $f$  can be implemented as a retrieval operation performed by the engine, e.g. simply by passing the URI as input to the repository API. In order to cope with syntactic and spelling variants, Lucene<sup>1</sup> is actually used as the index and search engine. That is, URIs and labels of entities are indexed, and using the fuzzy search feature of Lucene, a query is generated for each entered keyword. The engine returns ontology entities ranked according to syntactic similarity to the respective keyword. As there is only one minor syntactic difference in the example from the last section, the highest ranked entities for  $Q_U = \text{"Philipp Cimiano X-Media publications"}$  are indeed Philipp Cimiano, X-Media and publication. However, in other scenarios, this implementation of  $f$  based on syntactic similarity may not always find an appropriate mapping for each keyword. These mapped entities  $O'_S := \{f(k_i) | Q_U = \langle k_0, \dots, k_n \rangle\}$  will then be fed into the exploration step, which we will discuss in the following.

#### 4.2 Step Two — Exploring Connections Among KB Entities

Due to A1', we can restrict ourselves to the exploration of connections of the type  $\langle i, C \rangle$ ,  $\langle i_1, R, i_2 \rangle$  and  $\langle i, U, j \rangle$ . Using these concept and property member axioms, we explore all ontology entities related to elements  $O'_S$  identified in step one according to the algorithm shown in Fig. 2.

Basically, the exploration encompasses the traversal to neighbors from each of the elements in  $O'_S$ . Then, depending on the type of the particular element  $e \in O'_S$ , different traversals are performed to build a graph connecting  $e$  with all the neighbors within the specified range  $d$ . For instance, given a concept, all individuals are retrieved via concept member axioms. Given a property, property member axioms are used to navigate to individuals and data values, respectively. Figure 3, for example, shows the pseudocode algorithm for the recursive traversal from a particular individual to its neighboring concepts, individuals and data values. Neighboring individuals and data values are retrieved using property member axioms. The value of  $d$  is reduced by one in each recursion step to ensure that this traversal is limited to a certain range. Note that, due to marking elements of  $O'_S$  globally as visited, any element of  $O'_S$  is traversed at most once. In the end, we obtain a graph  $g$  containing all entities out of  $O_S$  which have a graph-distance

<sup>1</sup> see <http://lucene.apache.org/java/docs/>

not greater than  $d$  to at least one of the elements of  $O'_S$ . We call this structure the  $d$ -neighborhood of  $O'_S$ . Possibly, if  $d$  is small, it might be the case that the computed graph is not connected.

KB EXPLORATION( $O'_S, d$ )

```

1  INPUT a set of entities  $O'_S$  matching the terms and the traversal width  $d$ 
2  OUTPUT the graph containing all or some of  $O'_S$ 
3  Initialize new empty graph  $g$ 
4  for  $e \in O'_S$ 
5  do if  $e$  is a concept
6      then for all  $i$  being instances of  $e$ 
7          do I-P-I TRAVERSAL( $e, d, g$ )
8  else if  $e$  is an object property
9      then for all  $i, j$  with  $\langle i, e, j \rangle \in O'_S$ 
10         do I-P-I TRAVERSAL( $i, d, g$ )
11         I-P-I TRAVERSAL( $j, d, g$ )
12 else if  $e$  is a data property
13     then for all  $i, j$  with  $\langle i, e, j \rangle \in O'_S$ 
14         do J-P-I TRAVERSAL( $j, d, g$ )
15 else if  $e$  is an individual
16     then I-P-I TRAVERSAL( $e, d, g$ )
17 else if  $e$  is a data value
18     then J-P-I TRAVERSAL( $e, d, g$ )
19  return  $g$ 

```

Fig. 2. KB Exploration algorithm

Note that the exploration simply incorporates all elements within a certain range. Thus, some discovered elements may not really be needed to connect elements in  $O'_S$ . Therefore, from this graph, only those paths are selected where the first and the last vertex correspond to an element in  $O'_S$ . In particular, a modified version of the depth first search (DFS) procedure over graphs is used for computing all paths  $p \in P$  for each possible pair  $(a, b) \in O'_S$  such that  $p = (v_1, e_1, \dots, e_n, v_n)$ , where  $v_1$  is constructed using  $a$  and  $v_n$  is constructed using  $b$  and none of the vertices is visited more than once. These paths are fed into the next step.

### 4.3 Step Three — Deriving DL Conjunctive Queries from Connections

This step comprises three substeps. First, all different subsets of paths (called *connections*) are computed from  $P$  discovered previously. Then, for each subset, a query is derived. Finally, the resulting queries are ranked. The three substeps are described in the following:

**Computing Possible Connections:** A question can be derived when all elements  $O'_S$  identified in step one are connected. When merging all the paths  $P$  computed in step two, we however obtain a graph which may contain many different subgraphs connecting all the elements  $O'_S$ . Hence, it is a priori not clear which subgraph to choose as the correct interpretation of the keyword-based query. Therefore, we first compute all these

```

I-P-I TRAVERSAL( $i, d, g$ )
1  INPUT the individual  $i$  to be explored, the traversal range  $d$ , and the intermediate graph  $g$ 
2  OUTPUT updated graph  $g$  containing entities connected to  $i$  within the range  $d$ 
3  if  $i$  not marked as visited and  $d > 0$ 
4    then
5      mark  $i$  as visited within  $O_S$ 
6       $C_i := \{c \mid i \text{ instance of } c\}$ 
7      add edge  $(i, \text{type}, c)$  to  $g$  for all  $c \in C_i$ 
8       $P := \{(i, p, j) \mid \langle i, p, j \rangle \in O_S\}$ 
9      for all  $(i, p, j) \in P$ 
10     do if  $j$  not marked as visited in  $O_S$ 
11       then add a new edge  $(i, p, j)$  to  $g$ 
12         if  $j$  is an individual
13           then I-P-I TRAVERSAL( $j, d - 1, g$ )
14           else J-P-I TRAVERSAL( $j, d - 1, g$ )

```

**Fig. 3.** Individual-to-Individual traversal algorithm

subgraphs and rank these at a second step. The subgraphs connecting the elements in  $O'_S$  are calculated by the recursive procedure shown in the pseudocode algorithm in Figure 4. The input to the algorithm is the set of paths  $P$  as computed previously as well as an initially empty set  $R \subseteq O_S$  of vertices which have not yet been assembled into a graph connecting all the vertices in  $O_S$  as well as a subset  $C$  of already connected vertices. The recursion starts by selecting some edge connecting two arbitrary vertices and enters further recursions to add additional vertices. In this way, all the possible tree-shaped subgraphs connecting elements in  $O_S$  are determined.

**Mapping Connections to Queries:** Each of these connection graphs  $G_C$  are then translated to a corresponding DL conjunctive query  $Q_S$  as follows: an edge in  $G_C$  of the form  $\text{type}(v_i, v_c)$  (representing the connection  $\langle i, C \rangle$ ) is mapped to concept terms of the form  $\langle x : C \rangle$ , where  $v_i$  is a vertex constructed using an individual,  $v_c$  is constructed using a concept, and  $x$  is an individual or a variable. The concept of  $v_c$  is used as concept of the term. When the individual of  $v_i$  matches some  $e \in O'_S$ , then it is used as constant, otherwise a variable is used for the term. As the same individual might be used in many edges, the same variable must be used for the same individual. Besides concept member axioms, also property member axioms are used to connect entities in the exploration. Edges constructed with these axioms are of the form  $\text{property}_n(v_i, v_j)$ , where  $v_i$  ( $v_j$ ) is constructed either using an individual or a data value (the connections  $\langle i_1, R, i_2 \rangle$  and  $\langle i, U, j \rangle$ ). These edges map to role terms of the form  $\langle x : y \rangle : R$ , where a vertex constructed using an individual is mapped to a variable or constant just as described above. When  $v_i$  ( $v_j$ ) is constructed using a data value, it is simply mapped to constants of the role term. As the exploration incorporates only these two types of edges, this mapping is thus complete for the translation from  $G_C$  to  $Q_S$ . In our example, only one connection graph with the edges  $\text{name}(\text{uri1}, \text{Philipp Cimiano})$ ,  $\text{author}(\text{uri1}, \text{pub\#1})$ ,  $\text{hasProject}(\text{pub\#1}, \text{uri2})$ ,  $\text{name}(\text{uri2}, X - \text{Media})$ ,  $\text{type}(\text{pub\#1}, \text{publication})$  exists. Using the above specified mapping, this connection graph is translated to the final query:  $Q_S = \langle x, \text{Philipp Cimiano} \rangle : \text{name} \wedge \langle x, y \rangle : \text{author} \wedge \langle y, z \rangle : \text{hasProject} \wedge \langle z, X - \text{Media} \rangle : \text{name} \wedge \langle y, \text{publication} \rangle$ .

```

CALCULATESUBGRAPHS( $P, C, R, G, g$ )
1  INPUT the paths  $P$  calculated by DFS for all matching vertices  $O'_s$ 
2  OUTPUT all different subgraphs connecting the vertices in  $O'_s$ 
3  if  $R = \emptyset$ 
4    then  $G = G \cup g$ 
5  if  $g = \emptyset$ 
6    then  $G = \text{newGraph}$ 
7      for  $\{i, j\} \subseteq R$ 
8        do for each path  $p$  between  $i$  and  $j$  (as calculated by DFS)
9          do add  $(i, p, j)$  to  $G$ 
10         CALCULATESUBGRAPHS( $P \setminus p, C \cup \{i, j\}, R \setminus \{i, j\}, G$ )
11  else for  $i \in R$ 
12    do for  $j \in C$ 
13      do for for each path  $p$  between  $i$  and  $j$ 
14        do
15          add  $(i, p, j)$  to  $G$ 
16          CALCULATESUBGRAPHS( $P \setminus p, C \cup \{i\}, R \setminus \{i\}, G$ )

```

**Fig. 4.** Algorithm for Computing Connections

**Rank Queries:** Finally, the computed subgraphs have to be ranked. From A2, it follows that the smaller the length of the paths connecting the elements  $O'_s$ , the more likely they match the initial question in the mental model of the user. Thus, queries are ranked by the length of the longest path of the respective connection graph.

## 5 Ontology-Based Search and Exploration with Keywords

In this section, we discuss our implementation of the approach and show how it can be incorporated into a system for exploring and searching KBs. The system is evaluated and results are discussed in the last section in the light of the underlying assumptions.

### 5.1 Implementation

The presented approach for the interpretation of keywords with respect to a given ontology is integrated in our system called XXploreKnow!, which has been designed to support a combination of search and exploration in knowledge bases. A detailed description of this system will be published elsewhere. We will now describe a possible interaction of a user with XXploreKnow!.

At the beginning, the user enters keywords which are processed by the Lucene search engine. Ontology entities returned by this engine enter the exploration process, in which neighboring entities up to a width of  $d$  are considered<sup>2</sup>. As a result, the system visualizes a subgraph connecting the matched entities to the user and highlights the entities matching the keywords. Depending on the action performed by the user, e.g. clicking on the “search” or “xplore” button, subsequent interactions consist of either further exploration of the graph or inspection of the search results. With “xplore”, the user can

<sup>2</sup> Currently, the parameter  $d$  must be configured in the implementation. It has been set to 3 in our experiments.

expand nodes shown in the graph-based visualization to traverse to neighboring concepts and individuals connected via properties—as captured by concept restrictions and property member axioms. In addition, from an individual, the user can navigate to its types, and then along the concept hierarchy as specified by subclass axioms. By default, only assertional knowledge is retrieved (in order to keep the browsing performant) and shown in the visualization. During this exploration, the user can drag and drop elements from the visualization to the query view below the keywords to further refine the query. With “search”, the user’s query is sent to the inference engine. In this case, the different possible queries are ranked as described in previous sections and presented to the user, who can choose among different queries. The results, which may contain also inferred facts, are then finally shown to the user in a separate view.

## 5.2 Evaluation

In order to carry out an evaluation of the system, we have asked colleagues at the institute AIFB to provide queries in the way they would interact with a system capable of processing keyword based queries, along with the natural language description of the query. The request was sent by e-mail and 12 people responded. Some queries which were obviously out of the scope of the knowledge base were removed, resulting in a total of 42 different queries. These queries were incorporated only as an evaluation set and not used for the development or tuning of the approach. Examples for queries with different number of keywords posed by our users are: “*projects Blohm*” (Retrieve all projects that Sebastian Blohm is working on), “*phone Rudi Studer*” (Retrieve the phone number of Rudi Studer) or “*publications SmartWeb Pascal Hitzler 2002*” (Retrieve all publications published by Pascal Hitzler within SmartWeb in 2002). For the evaluation, one of the authors manually assigned conjunctive queries according to the natural language description. A query generated by our approach is regarded as correct if it retrieved the same answers as the hand crafted query. In line with work on question answering ([5], [4]), we evaluate the approach in terms of precision, recall and F-Measure. Precision  $P$  is defined as the number of correctly translated keyword queries (based on equivalence of results) divided by the number of cases for which the system was able to construct a query. Recall  $R$  is defined as the number of correctly translated keyword queries divided by all the keyword queries of the evaluation set, i.e. 42 in our case. The  $F_1 = \frac{2 * P * R}{P + R}$  measure is then the harmonic mean between precision and recall.

In case the query is selected by hand from the different queries generated, our system obtains a precision  $P = 85\%$ , a recall  $R = 52\%$  and a F-Measure  $F_1 = 64\%$ . In case we automatically choose the highest ranked question instead, the results are slightly lower with a precision of  $P = 69\%$ , a recall of  $R = 43\%$  and an F-Measure of  $F_1 = 53\%$ .

## 5.3 Discussion

Our evaluation has been performed with the knowledge base from which our institute portal is automatically generated<sup>3</sup>. The underlying ontology is the SWRC ontology ([1]), which allows for the representation of researchers, their publications, active projects etc.

<sup>3</sup> see <http://www.aifb.uni-karlsruhe.de/>

The evaluation has been carried out involving our colleagues, who visit and update the portal pages frequently, but do not know the underlying ontology in detail. Thus, some of the queries the users in our experiments asked the system contain keywords which do not correspond to entities in the knowledge base. Obviously, this is a problem for our approach as it violates assumptions A1' as well as the generic A1. According to our recall measure (43-52%), at least about half of the keyword queries fulfill assumption A1' and can thus be mapped to appropriate ontology elements. The higher precision of 69-85% on the other hand shows that, given that A1' is fulfilled, the generated query is correct in most cases. In fact, we found that most of the errors in our approach are produced in step 1. This means that the Lucene engine does not return the appropriate ontology elements in some cases. This problem could be for example addressed by integrating additional lexical knowledge about words as found in resources such as WordNet [12].

A further issue is related to our assumption A2, i.e. the assumption that the ontology entities the keywords map to are connected via paths of up to a given length  $d$ . We have experimented with a length  $d$  of 3 in our approach. Possibly, a higher recall could be achieved by using a higher value for  $d$ , but it is also probable that much more "background noise" would be introduced, thus making the selection of the relevant query more difficult. Overall, our assumptions have proved to be very valuable. Our first assumption (A1 / A1') states that users conceptualize their information need in terms compatible with the underlying ontology. While such an assumption is quite simplistic on the one hand and rather strict on the other, it turned out to be necessary as questions which do not fulfill this assumption are anyway out of the conceptual range of the system. From a practical point of view, this assumption is thus necessary. Assumption A2, which assumes that the ontology elements are connected with paths of a maximal length turned out to be crucial in order to restrict the search space to a specific part of the KB.

## 6 Conclusions

We have presented a generic approach for mapping queries in a user language into an expressive logical language. In particular, we have presented a particular instantiation of our generic approach which translates keyword queries into DL conjunctive queries using knowledge available in the KB. We have clarified in particular the assumptions on which our approach is based on. We have presented the current implementation of the system as well as first results of an evaluation of the translation process. The evaluation shows promising results w.r.t. precision, but still a lower recall which can be definitely increased by integrating lexical knowledge into the process of matching keywords to ontology elements. In the light of these evaluation results, we have argued that our assumptions are indeed reasonable and necessary for the interpretation and the answering of queries using ontologies.

Besides the integration of lexical knowledge to improve recall, we intend also to improve the runtime performance of our approach. will focus future work on boosting the performance. So far, the process of interpretation so relies mainly on assertional knowledge, resulting in a large number of A-Box queries that need to be processed during the exploration. We plan to exploit the available T-Box knowledge for a "guided exploration" of the connections between ontology entities to reduce the number of A-Box queries.



One major problem our approach suffers from is the fact that it does not consider that keywords can be ambiguous with respect to labels in the ontology and simply considers the first matching ontology element to start the exploration. Currently, in case of ambiguities, the exploration would have to be performed for each of the possible interpretations of a query term. However, the alternatives to explored might be exponential in the number of possible interpretations of the keywords. Future work will thus aim at a more appropriate treatment of ambiguities.

Finally, we will further develop the presented system to support an integrated approach for combined search and exploration in knowledge bases.

*Acknowledgements.* The research presented in this paper was partially supported by the European Commission in the context of the X-Media (IST-2006- 026978) and the NeOn (IST-2006-027595) projects.

## References

1. Baeza-Yates, R., Ribeiro-Neto, B.: Modern information retrieval. Addison-Wesley, Reading (1999)
2. Stojanovic, N., Studer, R., Stojanovic, L.: An approach for step-by-step query refinement in the ontology-based information retrieval. In: Proceedings of the IEEE/ACM Web Intelligence Conference (WI), pp. 120–137. ACM Press, New York (2004)
3. Lopez, V., Pasin, M., Motta, E.: Aqualog: An ontology-portable question answering system for the semantic web. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 546–562. Springer, Heidelberg (2005)
4. Cimiano, P., Haase, P., Heizmann, J.: Porting natural language interfaces between domains – a case study with the ORAKEL system –. In: Proceedings of the International Conference on Intelligent User Interfaces (IUI), pp. 180–189 (2007)
5. Popescu, A., Etzioni, O., Kautz, H.: Towards a theory of natural language interfaces to databases. In: IUI 2003. Proceedings of the International Conference on Intelligent User Interfaces, pp. 149–157 (2003)
6. Lei, Y., Uren, V., Motta, E.: Semsearch: A search engine for the semantic web. In: Staab, S., Svátek, V. (eds.) EKAW 2006. LNCS (LNAI), vol. 4248, pp. 238–245. Springer, Heidelberg (2006)
7. Bernstein, A., Kaufmann, E.: GINO – a guided input natural language ontology editor. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 144–157. Springer, Heidelberg (2006)
8. Li, Y., Yang, H., Jagadish, H.: NaLIX: An interactive natural language interface for querying xml. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, pp. 900–902. ACM Press, New York (2005)
9. Royo, J., Mena, E., Bernard, J., Illarramendi, A.: Searching the web: From keywords to semantic queries. In: ICITA 2005. Proceedings of the Third International Conference on Information Technology and Applications, pp. 244–249. IEEE Computer Society Press, Los Alamitos, CA, USA (2005)
10. Clarke, C., Cormack, G., Tudhope, E.: Relevance ranking for one to three term queries. *Information Processing and Management* 36, 291–311 (2000)
11. Sure, Y., Bloehdorn, S., Haase, P., Hartmann, J., Oberle, D.: The swrc ontology - semantic web for research communities. In: Bento, C., Cardoso, A., Dias, G. (eds.) EPIA 2005. LNCS (LNAI), vol. 3808, pp. 218–231. Springer, Heidelberg (2005)
12. Fellbaum, C.: WordNet, an electronic lexical database. MIT Press, Cambridge (1998)



# RDFSyc: Efficient Remote Synchronization of RDF Models<sup>\*</sup>

Giovanni Tummarello<sup>1</sup>, Christian Morbidoni<sup>2</sup>, Reto Bachmann-Gmür<sup>3</sup>, and Orri Erling<sup>4</sup>

<sup>1</sup>Digital Enterprise Research Institute, Galway, Ireland

<sup>2</sup>SeMedia, DEIT, Università Politecnica delle Marche, Ancona, Italy

<sup>3</sup>Talis Information Limited, Birmingham, U.K.

<sup>4</sup>OpenLink Software

g.tummarello@gmail.com, c.morbidoni@deit.univpm.it,  
reto@gmuer.ch, oerling@openlinksw.com

**Abstract.** In this paper we describe RDFSyc, a methodology for efficient synchronization and merging of RDF models. RDFSyc is based on decomposing a model into Minimum Self-Contained graphs (MSGs). After illustrating theory and deriving properties of MSGs, we show how a RDF model can be represented by a list of hashes of such information fragments. The synchronization procedure here described is based on the evaluation and remote comparison of these ordered lists. Experimental results show that the algorithm provides very significant savings on network traffic compared to the file-oriented synchronization of serialized RDF graphs. Finally, we provide the design and report the implementation of a protocol for executing the RDFSyc algorithm over HTTP.

**Keywords:** RDF synchronization algorithm, MSG, graph decomposition.

## 1 Introduction and Definitions

Remote synchronization of data files is a procedure by which local information (e.g. A data file) is updated over a network in order to be made identical with a remote one (or vice versa). Synchronizing could be trivially achieved by copying the entire remote file locally and then comparing it with the local one, but this is largely undesirable due to the performance issues in comparing the entire data file and most of all due to the bandwidth cost of frequent full data transfers.

In 1998, the rsync algorithm was developed [1] to efficiently synchronize remote binary files. rsync operates under the assumption that the changes will be significantly lower in size compared to the data file itself and that these are likely to happen in “clusters”, that is, in localized spots rather than distributed across the file. When this is the case, rsync can achieve synchronization by transferring data in quantity just slightly higher than the size of the changes. As such, rsync and others comparable algorithms that later followed are today the backbone of data replication across the Internet.

---

<sup>\*</sup>The work presented in this paper was supported (in part) by the Lion project supported by Science Foundation Ireland under Grant No. SFI/02/CE1/1131 and (in part) by the European project DISCOVERY No. ECP-2005-CULT-038206.

In this paper we provide an algorithm for the efficient synchronization of RDF models. RDF Models cannot be efficiently synchronized by the rsync or similar algorithms due to the RDF semantics itself. Serializing an RDF model could in theory result in a factorial number of ordering for the composing triples and even more when blank nodes are involved. Remote RDF synchronization has been highlighted as a very important but open problem [2].

One could think of serializing the graph into a deterministic, canonical way, e.g. by ordering the triples in lexicographical order. This is partially possible as we will see, but the results of a simple rsync synchronization will be shown to be still unsatisfactory, especially when graphs contain blank nodes (e.g. FOAF personal profile documents, OWL/RDFS ontologies, etc.).

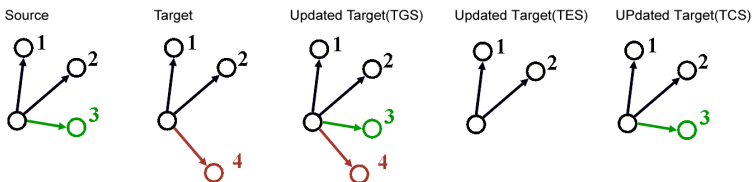
Our proposed algorithm, which we call RDFSsync, works by decomposing RDF graphs in minimal subsets of triples and then creating hashes that can be sorted and efficiently synchronized. In this way, we will show how the RDFSsync algorithm can exhibit on RDF models the same behavior of rsync for general data files: as long as the number of triples involved in changes is small compared to the size of the entire model, as is the case for frequent updates, the network traffic and the computational power required for the synchronization will be very small.

With more and more data being made available on-line on the Semantic Web as HTTP retrievable data (open linked data model or RDF dumps), the RDFSsync algorithm seems of great importance to enable the Semantic Web itself to scale. With respect to this, we conclude this paper by illustrating how the RDFSsync algorithm can be transparently exposed by a web server providing its service over HTTP and triggered by a standard content negotiation procedure.

The paper is structured as follows: the basic theory for the decomposition of RDF models into Minimum Self-contained Graphs (MSG) is illustrated in section 2. This will be used to derive methodologies for remote RDF synchronization, illustrated in section 3. Experimental results will be illustrated and discussed in section 4, while additional issues are covered in section 5. Section 6 will illustrate the algorithm as implemented on top of HTTP.

### 1.1 Different Sync Modalities

Unlike rsync, which can only be used to make remote files identical, RDFSsync is capable of performing different “kinds” of synchronization between a *target* and a *source* model.



**Fig. 1.** Starting from two graphs and considering one of them the 'source' and the other the 'target', we show how the target will be modified by running RDFSsync in different modes. Edges and nodes with the same color and number are equals, different colors and numbers represent different RDF statements.

Using the algorithms we propose, it will be possible to cause the *target*:

- to be equal to the *merge* of both graphs (Target Growth Sync, TGS);
- to delete information that is not known by the *source* (Target Erase sync, TES);
- to be equal to the *source* (Target Change Sync, TCS).

Figure 4 shows these different kinds of exchanged and the resulting RDF graphs. Here, the definition of *merge* and *equals* are strictly derived from RDF Semantics [3]. This means in practice that:

- B-nodes IDs will not be preserved;
- Sync is not required to transfer redundant information that might be contained in the graphs. This means that the only *lean* versions of two graphs (as defined in [3]) are required to be isomorphic for the graphs to be considered equal;
- Serialization format idiosyncrasies (e.g. RDF/XML comments) are ignored.

## 2 The Minimum Self Contained Graph Theory

In this section we will illustrate the Minimum Self-Contained Graph (MSG) theory. This and the following section expand on [4] by considering additional cases and conditions. The theory set forth in [4] is here reported in full to serve as base for the discussion of the RDFSyc algorithm.

Let's first define what is the minimum “standalone” fragment of an RDF model. As blank nodes are not addressable from outside a graph, they must always be considered together with all surrounding statements, i.e. stored and transferred together with these. MSGs are the smallest components of a lossless decomposition of a graph which does not take into account inference (e.g. the OWL methods for detecting identity of nodes). Discussion about RDFSyc with respect to inference is given in section 5.3.

We will here give a formal definition of MSG (Minimum Self-Contained Graph) and will prove some important properties.

**Definition 1.** *An RDF statement involves a name if it has that name as subject or object.*

**Definition 2.** *An RDF graph involves a name, if any of its statements involves that name.*

**Definition 3.** *Given an RDF statement  $s$ , the Minimum Self-Contained Graph (MSG) containing that statement, written  $MSG(s)$ , is the set of RDF statements comprised of the following:*

- *The statement in question;*
- *Recursively, for all the blank nodes involved by statements included in the description so far, the MSG of all the statements involving such blank nodes;*

This definition recursively builds the MSG from a particular starting statement; we now show however that the choice of the starting statement is arbitrary and this leads to a unique decomposition of any RDF graph into MSGs.

**Proposition 1.** The MSG of a ground statement is the statement itself.

**Theorem 1.** If  $s$  and  $t$  are distinct statements and  $t$  belongs to  $MSG(s)$ , then  $MSG(t) = MSG(s)$ .

**Proof:** We first show that  $MSG(t) \subset MSG(s)$  and then  $MSG(s) \subset MSG(t)$ .  $MSG(t) \subset MSG(s)$ : This is straightforward from Definition 3 and knowing that  $t \in MSG(s)$ .  $MSG(s) \subset MSG(t)$ : Let  $t$  be a ground statement. Then  $MSG(t) = t$  for Proposition 1. But then  $t \notin MSG(s)$ , which contradicts the hypothesis  $t \in MSG(s)$ .

Hence,  $t$  is not a ground statement. Then there is either a single blank node connecting the statements  $t$  and  $s$  (e.g., writing statements in  $n3$ ,  $t = ":a :p :_id1"$ ,  $s = ":_id1 :q :b"$ ) or there is a sequence of blank nodes connecting  $t$  and  $s$ . In the first case, there is a blank node of  $s$  involving  $t$  (by hypothesis, since  $t \in MSG(s)$ ). So  $s \in MSG(t)$  and  $MSG(s) \subseteq MSG(t)$ . In the second case, there is a blank node of  $s$  involving  $MSG(t)$ , also in this case  $s \in MSG(t)$  and  $MSG(s) \subseteq MSG(t)$ . In other case, where  $s$  does not involve any blank node of  $t$  or of  $MSG(t)$ , the hypothesis  $t \in MSG(s)$  is contradicted.

**Theorem 2.** Each statement belongs to one and only one MSG.

**Proof:** Let  $s \neq t \neq u$  be distinct statements, and let  $s$  belong to different MSGs:  $s \in MSG(t)$  and  $s \in MSG(u)$ . By Theorem 1,  $MSG(s) = MSG(t)$  and  $MSG(s) = MSG(u)$ , hence  $MSG(t) = MSG(u)$ , so the three MSGs considered are actually the same.

**Corollary 1.** An RDF model has a unique decomposition in MSGs.

**Proof:** This is a consequence of Theorem 2 and of the determinism of the procedure.

### 3 MSG Based Graph Decomposition and Merging

As a consequence of Corollary 1, after a graph has been decomposed into MSGs, it can be incrementally transferred between parties with granularity down to one MSG at a time. As a consequence of theorem 2, such a transfer would be maximally network efficient since statements would never be repeated. While the above results intuitively apply to most of the graphs, there are a few special cases which require particular attention.

#### 3.1 Non-lean Graphs

In the same way as in relational databases table entries are never duplicated, RDF graphs are defined as being set of triples rather than a collection of triples, that is, triples are never duplicated. However, even without containing duplicate triples an RDF graph may contain redundant information.

The reason of this is that the RDF semantics gives blank nodes the meaning of existentially quantified variables, so while it is legal to have a graph containing multiple isomorphic subgraphs, such a graph is said to be 'non lean' as it expresses exactly the same knowledge as it would have after removing all but one of the isomorphic subgraphs. In other words, a non-lean graph is a graph in which some of its triples can be removed without changing the meaning of the information expressed in the graph.

In the case of a non-lean graph, the decomposition into MSGs could result in a certain amount of MSGs to be completely indistinguishable if it were not for the ids of the blank nodes they contain, so they are isomorphic. In such situations the MSG is said to be 'repeated' in the decomposition.

While such a redundant information can be kept in RDF serialization formats and most RDF triplestores currently available do not remove redundancies, implementations of RDFSyc are not required to transfer redundancies: in fact the only reason for an RDFSyc implementation to keep redundancies in some cases is the computational cost of complete leanification.

In the case of the decomposition into isomorphic MSGs a partial leanification can be done in the interest of saving in terms of network bandwidth and storage requirements as more convenient processing of the target graph(s).

For these reasons RDFSyc considers two graphs equivalent even if the same MSG is repeated in one of the decompositions and not in another one: this might result in two graphs with different number of triples after the synchronization, but still the lean versions of the resulting graphs will be isomorphic.

### 3.2 Removing Redundant MSGs

There are other cases in which, with respect to the RDF semantics, an MSG can be removed from a decomposition without changing the meaning of the graph resulting from merging the MSGs. This happens when the graph resulting from removing an MSG *a* (i.e. the union of the remaining MSGs) has a subgraph which is an instance of *a*, in which case *a* can be removed without changing the content expressed by the graph. With respect to RDF semantics the MSG *b* is an instance of the MSG *a*, if they are isomorphic except for a number (zero or more) of blank nodes in *a* which correspond to grounded nodes in *b*.

In these cases, for the purpose of RDFSyc, the MSG can be safely removed. Note however that other applications of MSGs, such as tracking versions and provenance, which are outside the scope of this paper, may not allow removing such MSGs. So in the general case the identity criterion of a set of MSGs is not the same as of the graph resulting from the merge of this set.

### 3.3 Canonical Serialization of MSGs and MSG's Hashes

MSGs are standalone RDF graphs and, as such, they can be processed with algorithms like canonical serialization in order to provide a sort of digest or hash value of the graph (as discussed in [4]). We use an implementation of the algorithm described in [5], which is part of the RdfContextTools Java library<sup>1</sup>, to obtain a canonical string representing the MSG and then we hash it to an appropriate number of bits to reasonably avoid collisions (math and common sense say 128-160 bits will suffice). This hash acts as a unique identifier for the MSG.

There might be issues in certain situations where the canonical serialization described in [5] will behave in a non-deterministic way. It can happen for some graph structures involving several bnodes, especially where bnodes have no property or label attached. Such cases are luckily not very frequent in real-world use cases and in particular in the RDF graph we experimented with (see section 4).

As there is a finite number of possible serialization alternatives, it is always possible to compute all the hashes that an MSG, in one such very hard case, has and treat them as being equivalent in the synchronization procedure.

---

<sup>1</sup><http://semedia.deit.univpm.it/tiki-index.php?page=RdfContextTools>

### 3.4 Canonical Serialization and RDF Graphs Synchronization

As a graph can be decomposed unequivocally into a set of MSGs, it can be canonically represented by the ordered list of the identifiers (hashes) of its composing MSGs. In the RDFSsync algorithm, such lists are created independently at each end and ordered by the binary value itself. The synchronization is then performed in 2 steps:

1. A diff between the source and the target ordered lists of MSGs is performed;
2. Such diff indicates which MSGs have to be requested from the other side and which should be deleted in the local model.

To perform the diff between the source and the target ordered list of MSGs, we first need to transfer or locally reconstruct the remote list. For this purpose, two procedures can be employed: the first, trivial one is to directly transfer the list, the second is to create a copy of the remote list, using the standard rsync, from the local list.

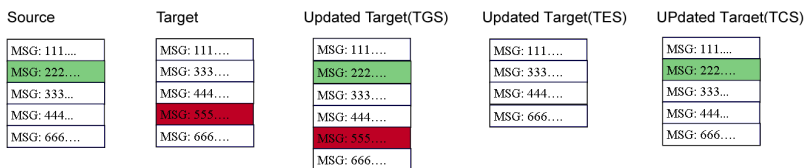
The latter approach can be shown to be highly efficient in case of small differences between the two lists, since rsync is optimized for differences which result in shifting of data blocks within the file. This is the case when small differences exist between a source and a target model; the few added or removed MSGs in the ordered list have the effect of “shifting” the remaining MSG hashes.

In case of MSGs hashes lists, big changes (e.g. a lot of MSGs added to one model) result in a great amount of hashes to be inserted in random positions of the list, which cause almost all the file to be transferred, plus, of course, the overhead of the rsync operation (calculating hashes of file sections, transferring and comparing them). The experimental data, as shown in the next section, can be used to select the best approach to follow, given an approximate estimate (e.g. an expectation) of the amount of differences between the models (e.g. one would choose the rsync-based synchronization if the RDFSsync is executed very often compared to the changes in the model).

Once the two lists are available to the target host, a diff is computed in order to obtain:

- The list of MSGs to be requested from the remote model (in case of a TCS or TGS sync), which is then sent to the remote host which complies to the request.
- The list of MSGs to be deleted in the local model (in case of a TCS and TES sync).

Figure 2 illustrates this process, showing how the different RDFSsync methodologies can be applied to two remote MSG lists.



**Fig. 2.** Two ordered lists of MSGs are processed by RDFSsync in different modes, producing different results at the targets side

<sup>2</sup><http://wymiwyg.org/rdf-utils>

## 4 Experimental results

In this section we show and discuss the results obtained from actual runs of RDFSyc, as implemented in the RDFSContextTools. Runtime results of this implementation (the synchronized graphs) have been validated with the diff utility independently developed within the 'RDF Utils' project<sup>2</sup>. We show the performance of the algorithm in three notable cases.

One, labeled *SyntGraph no bnodes*, deals with a synthetically generated graph composed completely by ground triples. This graph is 1.07 MB in size and is composed by 8000 triples (therefore 8000 MSGs). The performance as evaluated on this graph is completely comparable with any other made completely of ground triples such as DBpedia dataset. The second one, labeled *SyntGraph bnodes*, also deals with a synthetically generated graph, this time with a moderate number of blank nodes (approximately 600, with MSGs composed by 2, 3 and 4 triples). The graph is 1.3 MB in size and has 9000 triples in 7800 MSGs. The third one, labeled *DBWorld Graph* is a fragment of a real world graph which makes a more extensive use of blank nodes<sup>3</sup>.

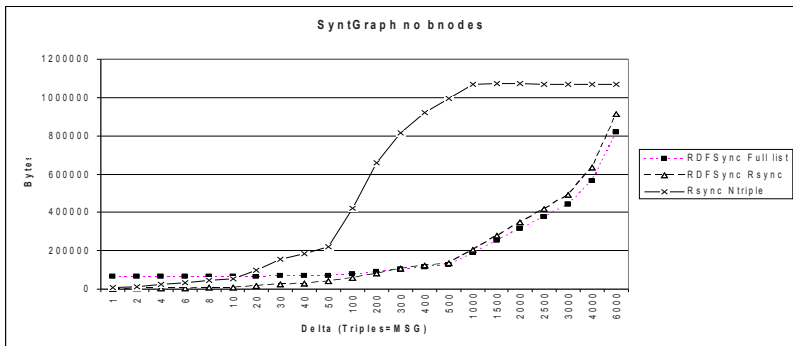


Fig. 3. Traffic vs Delta MSG for a graph that makes no use of bnodes

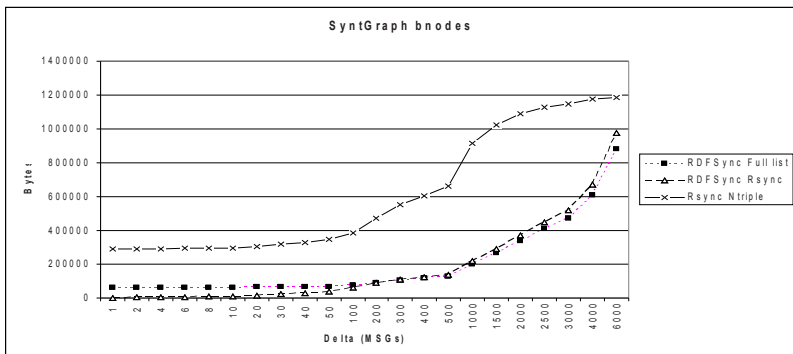


Fig. 4. Traffic VS Delta for a graph with a moderate number of bnodes

<sup>3</sup><http://sw.deri.org/~aharth/2005/08/dbworld/dbworld.rdf>, representing calls for papers in RDF.

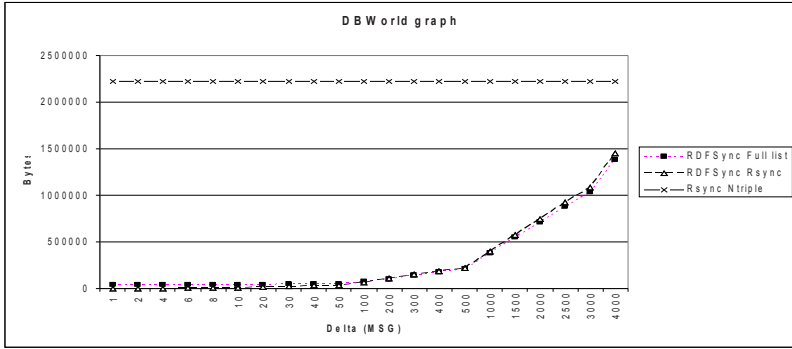


Fig. 5. Traffic vs delta MSGs for a graph that makes intense use of bnodes

This graph is 2.1 MB and contains approximately 13000 triples in 5000 MSGs. Performance on this graphs will be comparable with that on others with similar characteristics e.g. DBLP dump in RDF<sup>4</sup>.

The algorithms that we compare are:

- *RDFSyc Full list*: by graph decomposition we produce a list of 64 bits MSG hashes. This is entirely copied on the other side and then the missing ones are requested;
- *RDFSyc rsync*: the list of hashes, created as above, is synchronized itself with rsync. The missing MSGs are then copied;
- *rsync*: rsync is applied on a lexicographically sorted list of triples (Ntriples).

In every case the traffic shown is inclusive of all the algorithm overhead (hashes synchronization plus the actual transport of the MSGs serialized in Ntriples).

The scale is the same for all the graphs and is pseudo logarithmic for the delta axis while linear on the traffic. As the lines are close in the SyncGraph no bnodes graph for the first part (low delta), the numeric results are reported in Table 1.

Table 1. The data from the low delta runs on the SyncGraph no Bnodes

Delta(in MSGs)	RDFSyc Full list	RDFSyc Rsync	Rsync Ntriple
1	64126	1895	7403
2	64250	2743	12535
4	64504	4445	22805
6	64756	5693	33073
8	65008	7841	43341
10	65258	9539	53607
20	66506	17459	99935
30	67764	25801	156273
40	69036	28667	182599
50	70290	39237	221999
100	76582	57169	420507

<sup>4</sup><http://lists.w3.org/Archives/Public/www-rdf-interest/2004Dec/0015>



As evident from the experimental data, applying the proposed algorithm gives very high bandwidth saving as opposed to the alternative rsync Ntriple algorithm in almost all cases. When bnodes are used in the graph, the difference is as much as the entire graph size for any delta (*DBWorld Graph*). This can be explained with the blank nodes Ids (which are usually random generated by the triplestores) that have to be synchronized at the both ends. If these are too many, there will be more than one change per rsync block size thus causing the transfer of the entire dataset.

Performances are dramatically different also when a small number of blank nodes are used (*SyntGraph bnodes*) as the blank node impose an almost constant weight which is otherwise not present for the RDF sync algorithm (especially in its rsync format). The different for small updates is huge, as much as 150 to 1 for a single delta MSG (1.8 k on the RDFSyc algorithm vs 290k of rsync).

The most “difficult” case is the *SyntGraph no bnodes*. Even in this case however, RDFSyc outperforms rsync approximately 4 to 1 (for the single delta MSG) to reach approximately 8 to 1 past the 100 MSG mark (which is only approximately 1.2 percent of the graph size).

With respect to the difference between sending the entire list or not, these data shows that this is seldom a good strategy (if not in cases of very large expected delta).

## 4.1 Optimizations

The data transfer can be further optimized by using weaker hashes (a lower number of bits) with a master hash calculated at both ends list of hashes is as supposed to (no collision has detected). With this procedure in place, the MSG hashes could be of 32 bits, thus cutting the list size in half, at the sole cost of having to redo the sync with stronger hashes in very rare cases when the weak ones fail.

## 5 Additional issues

### 5.1 Complexity

Computationally speaking, evaluating the MSGs each time one wants to RDFSyc is a time consuming procedure; although such algorithm is  $O(n)$  with  $n$  the number of triples, the large number of API calls to explore the graph usually determines a slow execution. The solution is to use caching techniques so that each time a new information is inserted, the MSG is calculated. MSGs are by definition immutable (mutating one can be seen as removing a statement and adding another one), so this is possible to cache them with great efficiency.

In the implementation constructed using OpenLink Virtuoso<sup>5</sup>, the MSG hashes are calculated in a server side stored procedure, saving on client-server communication delays. For graphs involving relatively few blank nodes, the MSG calculation time is near linear since all triples of a subject are read serially and translated from internal ID to the IRI (Internationalized Resource Identifier)<sup>6</sup>.

---

<sup>5</sup><http://www.openlinksw.com/virtuoso/>

<sup>6</sup><http://www.w3.org/International/O-URL-and-ident>

With a 2GHz Xeon processor, about 13000 MSG's can be calculated per second, provided a warm cache. An auxiliary table is used for sorting the MSG hashes, with a two-part key, consisting of a graph ID and MSG hash and a dependent part of subject, predicate and object. Keeping an up to date MSG table increases storage consumption by about 25%. Retrieval of MSG's requires only a lookup of this table and typically no extra joining.

For graphs with hundreds of millions of triples, making the MSG list is a time consuming process. This can be alleviated by keeping an update log table which records additions and deletions of triples with an associated timestamp. These changes can then be reflected on the MSG table and removed from the log table. The maintaining of a log table adds an overhead of approximately 15% to the insert or delete times and adds practically no IO since the additions are always in ascending order.

## 5.2 Non lean MSGs

It can happen, in situations in which an MSG is not lean, that is it has a duplicated statement where, for example, the subject and the predicate are identified by the very same URI but the objects are two distinct blank nodes. In this case, this MSG can be considered to be equivalent to a lean version (where one of the duplicated statements is missing), but the hashes of the two MSGs turn out to be different.

This can be avoided by leanifying each MSG before carrying out the hash, a procedure that will be addressed in future versions of the RDFSyc implementation.

## 5.3 Shortcomings of the Heuristic Canonical Serialization

It is possible that, by using the canonical serialization as described in [5], two isomorphic MSGs give two different hashes. This is true in cases where many consecutive blank nodes are used and these are connected to the same URI. In this case, labeling of blank nodes becomes difficult and a 2 step algorithm can fail. It is conjectured that an N step canonical serialization algorithm could probably take care of all these cases but as it is not proven and as the complexity of such algorithm would necessarily be higher, for RDFSyc we follow a different approach.

In case an MSG is in fact serialized differently at the two ends, this would show as "missing" from one side and "added" from the other. Once transferred, the "additional" one is in fact checked against all the "missing ones" for isomorphism in order to rule out such case. Such checking is usually very fast given that MSGs are usually small, furthermore a simple node count and or a simple lexicographical triple ordering can rule out trivial cases.

## 5.4 Inferencing

It is to be noticed that, beyond RDF-entailment, this paper does not take into account other reasoning capabilities as provided by RDFS [6] and OWL [7]. The algorithm is therefore applicable to "base" RDF models, that is, sets of triples with no consideration for automatic inferences (e.g. same-as or inverse-functional-properties).

Under a certain point of view, this is somehow "cleaner" as it does not require that the sending and the receiving party agree on ontologies. In particular scenarios, however, the decomposition into MSGs could lead to very big chunks of data, for

example for databases which contain a lot of anonymous resources identified by inverse functional properties. In this case the MSG decomposition would lead to bigger components than a decomposition taking into account (inverse) functional properties.

An example of such a store would be a graph obtained by merging a number of personal FOAF files, where bnodes are often used to represent a person. In such cases the RDFSyc procedure would still give correct results but the decomposition would degrade in efficiency, generating big MSGs containing information about more than one person (e.g. sub-groups of persons connected by *foaf:knows* properties). To avoid this, inverse functional properties can be used to stop the recursion in the MSG definition, similarly to what suggested in [8].

## 5.5 Atomicity and Integrity

Given that the MSGs hash are content based, the RDFSyc algorithm poses no real questions of atomicity and integrity. As each MSG is atomic per se, as long as one arrives entirely it can be added to the synchronizing database. If the flow of MSGs is interrupted, the synchronization can be restarted and it would only cover the missing MSGs (it would not repeat those that were previously transferred).

## 6 RDFSyc over HTTP

RDFSyc, as implemented to perform the experiments in the previous section, operates over a TCP/IP connection. In this section we will however illustrate a modification of the RDFSyc algorithm to operate on top of HTTP. Such modification has very important practical consequences: it is well known in fact that the modern Internet highly favors (if not enforces) HTTP over any other protocol as a way to reach out from any local network configurations (e.g. local firewalls). So, while the TCP/IP based implementation is certainly useful for server to server scenarios, the HTTP version is practically fundamental for client to server scenarios.

The main challenge of the HTTP based synchronization protocol is comparing large graphs and efficiently detecting a usually small set of differences in the MSG checksum list by using a stateless request/response paradigm.

In the following, *receiver* refers to the party initiating the synchronization and being updated, while *sender* is the party providing the data. The protocol does a sequential merge of the sorted MSG checksum lists on either side: since the lists often have largely the same content, the protocol keeps a pointer on the receiver and another on the sender. The pointers are said to be in sync if they point to the same checksum on either side.

The lists of MSG checksums are divided into blocks of 256 checksums. This works reasonably if there is a difference between the lists every 1000-2000 checksums, but can be varied based on difference estimates (e.g. based on the time since the last update).

The synchronization step consists of the following:

- The receiver sends the checksum at its sync pointer plus a set of block checksums for checksum blocks starting at its sync pointer;
- The sender replies with the offsets of the blocks that are different;

- The receiver sends the checksum blocks for the blocks that were different;
- The sender replies with the MSG serializations that are needed to sync the differing blocks. The message also contains the highest checksum covered in the step, indicating the sync pointer value on the sender.

The client sets its sync pointer to the sender's sync pointer. If the receiver has this same checksum, we repeat the process. If the receiver has a next checksum that is higher than the sender's next checksum, the receiver requests the MSG serializations with checksums between the receiver's sync pointer and the sender's sync pointer. If the receiver has no further checksums, it asks for the remaining MSG serializations. When both sync pointers are at end the process is completed. Details of special cases are omitted for brevity.

As mentioned, such sync process is stateless. It can be repeated any number of times and will terminate rapidly if both sides are already in sync. Locking of data in the data store is usually minimal since the transaction unit is small.

## 6.1 Performances

With the corpus of DBPedia<sup>7</sup>, we have about 30 million triples accumulated over several years of Wikipedia editing. These are derived from 1.6 million articles. Assuming a rate of change of 20% of articles per year and assuming a change influenced half the triples in each, we'd have 3 million changed triples per year. This would be an average of 8219 per day, or that is a checksum every 3650 checksums would change on the average. If we had a checksum block of 100 checksums, we could send units of 4000 checksums, represented as 40 block checksums of 100 checksums, equals 320 for 8 byte checksums plus overhead and expect one mismatched block in each. We would exchange the information in the mismatched block, amounting to one serialized MSG, under 100 bytes compressed. Thus, we would cover an average of 3650 MSG's with 2 round trips and about 500 bytes. Transferred. Thus, for a day's worth of changes, we'd have 16438 messages totaling approximately 8.2 megabytes. A full HTTP download of the same dataset would require 1.6 Gigabytes of traffic.

## 7 Specific Applicability

The Semantic Web is centered around the idea of distributed and cooperative metadata production and consumption, hence the remote RDF model synchronization is clearly an important goal to achieve. This is even more so if one considers that most of the semantic web applications currently known (client side, such as Piggy Bank<sup>8</sup>, Tabulator<sup>9</sup>, DBin[9]) or server side work by downloading remote RDF data locally in order to then perform inference, queries and data merging as needed. To make sure that the local data is up to date with the remote data, such applications today follow the "RSS approach" and continuously download the file to get updates. Clearly this is not a scalable approach when databases of considerable dimensions are involved, hence the use of the RDFSyc procedure to enable frequent updates at low cost.

---

<sup>7</sup><http://dbpedia.org/>

RDFSyc can also be used to create powerful centralized RDF based services, by enabling efficient synchronization across a grid structure. Graph synchronization might be useful in the aggregation of news feeds available in RDF, such as RSS 1.0 [10] or more recently as AtomRDF<sup>10</sup> or AtomOWL<sup>11</sup>: since the same news is often published in different feeds, using RDFSyc would prevent multiple downloading of the same entry. Another context of applicability, which has strong similarities with the one just described, could be within systems like URIQA [11], or in general following the linked data paradigm, where a server answers to client's requests about a URI/URL, returning a subgraph representing the knowledge of the server about the URI/URL. Using RDFSyc in this case could allow clients to efficiently update their knowledge about a resource from an authoritative server.

As said in section 1.1, RDFSyc-based update operations can be performed in different modalities. These modalities address distinct scenarios and social environments. The TGS mode, for example, perfectly fits a scenario like the one described in [12], where peers typically want to know everything that has been said around a topic. The TGS modality would be used to merge the news from different sources (feeds) into one target model. The TCS mode could be preferable in a URIQA like scenario, where information owned by the server are to be considered authoritative and clients might want to act as 'mirrors', thus deleting triples which are no more on the server.

Furthermore we think RDFSyc might prove very valuable in mobile or wireless environments, where bandwidth efficiency issues are even more important.

## 8 Related Works

While the problem of finding diffs and generating patches for RDF graphs has been investigated, to the best of our knowledge, our algorithm is the first to address their efficient remote synchronization.

An approach to diffs and patches of RDF graphs is described in [2], where the authors introduce the concept of functionally ground nodes, which are blank with an inverse functional property value. Functionally ground nodes behave like named resources, once an ontology has defined which properties are inverse functional, hence the diff algorithm becomes straightforward. This approach works under some assumptions on the RDF graph it applies to. All the nodes have to be ground, or functionally ground so it is not applicable to the general case, where blank might be non functionally ground (e.g. the 'root node' of an `rdf:Bag` construct). Furthermore, the methodology relies on the OWL layer for identifying the inverse functional properties, limiting its scope to scenarios where an ontology has been previously agreed upon and is owned by each party involved in the exchange.

In this paper we considered the case of generic RDF graphs and we intentionally based our approach on the RDF layer as we think that ontologies and reasoning capabilities should be applied at a different level. Choosing the appropriate ontology (or ontologies, as data could be heterogeneous) to present, edit and interact with the data, should be left, in our opinion, to domain aware applications, which could make use of RDFSyc, at a lower level, for import and merge 'pure' RDF.

---

<sup>8</sup> [http://simile.mit.edu/wiki/Piggy\\_Bank](http://simile.mit.edu/wiki/Piggy_Bank)

<sup>9</sup> <http://www.w3.org/2005/ajar/tab>

<sup>10</sup> <http://djpowell.net/blog/entries/Atom-RDF.html>

<sup>11</sup> <http://atomowl.org>

The idea of decomposing an RDF graph in small parts, which in our case are MSGs, has been investigated also in [8], where the concept of “RDF molecules” is introduced in order to obtain smaller components by taking functional and inverse functional properties into account. The price to pay for this approach is an higher complexity of the decomposition and of the algorithms to achieve it, as well as the requirement of a shared set of ontological beliefs. Furthermore, the original concept of molecules does not provide a deterministic decomposition so that extensions had to be defined to allow this decomposition in implementations such as the RDF diff and patch utility in WYMIWYG `rdf-utils`<sup>12</sup>. This approach has been further developed by HP laboratories in Bristol, leading to the Graph Versioning System (GVS)<sup>13</sup>.

Many works in literature deal with the problem of synchronizing different 'versions' of the same structured, XML based knowledge. [13], [14] and [15], describe algorithms and methodologies to merge and patch XML structured documents, both in the case of two-way synchronization (where the two files are independently created) and three-way synchronization (where the two files are separately obtained refinements of the same source file, that is available to the merging algorithm).

These algorithms are based on the analysis and comparison of DOM structures of the documents or of sequence and order of XML tags and PCDATA. These methods are not efficient if applied to RDF/XML, as RDF can be serialized in different ways and two different serializations could have the same information content.

## 9 Conclusions

We described a methodology to perform an efficient synchronization of RDF models called RDFSynC. RDFSynC is based on RDF Semantics only and it is therefore a general purpose tool independent of the application domain and independent of the used ontologies. The sync acts purely on the level of the content expressed by RDF graphs as defined by RDF Semantics.

Experimental results show that the algorithm provides very significant saving on network traffic compared to a simple `rsync` on a ordered list of triples. Such savings are even more evident when small amounts of differences exists, in other words the more frequent are the updates, the more efficient they are. This ultimately enables scenarios where large datasets (e.g. Even those of the size of DBpedia) can be kept in sync even multiple times a day with local (personal or intranet) copies without gigabytes of traffic locally and or terabytes of traffic remotely.

Finally, the procedure has been adapted also to operate over the HTTP protocol thus enabling end user clients to use it directly.

## References

1. Tridgell, A.: Efficient Algorithms for Sorting and Synchronization, PhD Thesis, Australian National University (1998)
2. Berners-Lee, T., Connolly, D.: Delta: an ontology for the distribution of differences between RDF graphs. In: MIT Computer Science and Artificial Intelligence Laboratory (2004)
3. RDF Semantics, W3C Recommendation (2004), <http://www.w3.org/TR/rdf-mt/>

<sup>12</sup><http://wymiwyg.org/rdf-utils>

<sup>13</sup><http://gvs.hpl.hp.com/>

4. Sayers, C., Karp, A.H.: RDF Graph Digest Techniques and Potential Applications, HP Technical Report (2004)
5. Carroll, J.: Signing RDF Graphs, TechnicalReport HPL-2003-142, HP Lab (2003)
6. RDF Vocabulary Description Language 1.0: RDF Schema (2004), <http://www.w3.org/TR/rdfschema/>
7. OWL Web Ontology Language Overview, <http://www.w3.org/2001/sw/WebOnt/>
8. Ding, L., Finin, T., Peng, Y., da Silva, P.P., McGuinness, D.L.: Tracking RDF Graph Provenance using RDF Molecules. In: Proceedings of the Fourth International Semantic Web Conference (2005)
9. Tummarello, G., Morbidoni, C., Nucci, M.: Enabling Semantic Web communities with DBin: an overview (2006)
10. RDF Site Summary (RSS) 1.0 (2000), <http://web.resource.org/rss/1.0/>
11. URIQA The URI Query Agent Model (2003), <http://sw.nokia.com/uriqa/URIQA.html>
12. Tummarello, G., Morbidoni, C., Pettersson, J., Piazza, F., Puliti, P.: RDFGrowth, a P2P annotation exchange algorithm for scalable Semantic Web applications (2004)
13. Lindholm, T., Fault-tolerant, A.: Three-way Merge for XML and HTML, Internet and Multimedia Systems and Applications (EuroIMSA), Grindelwald, Switzerland (2005)
14. La Fontaine, R.: Merging XML files: a new approach providing intelligent merge of XML data sets, XMLEurope (2002)
15. Tancred Lindholm, XML Three-way Merge as a Reconciliation Engine for Mobile Data. In: Third International ACM Workshop on Data Engineering for Wireless and Mobile Access, San Diego, California (2003)



# Sindice.com: Weaving the Open Linked Data

Giovanni Tummarello, Renaud Delbru, and Eyal Oren

Digital Enterprise Research Institute  
National University of Ireland, Galway  
Galway, Ireland

**Abstract.** Developers of Semantic Web applications face a challenge with respect to the decentralised publication model: where to find statements about encountered resources. The “linked data” approach, which mandates that resource URIs should be de-referenced and yield meta-data about the resource, helps but is only a partial solution. We present Sindice, a lookup index over resources crawled on the Semantic Web. Our index allows applications to automatically retrieve sources with information about a given resource. In addition we allow resource retrieval through inverse-functional properties, offer full-text search and index SPARQL endpoints.

## 1 Introduction

The Semantic Web can be seen as a large knowledge-base formed by sources that serve information as RDF files or through SPARQL endpoints. A fundamental feature of the Semantic Web is that the graphs are decentralised: it has no single knowledge-base of statements but instead anyone can contribute statements by making them available in a public web space. These sources might have nothing in common, but by using shared identifiers (URIs) and shared terms, their information can be merged to provide useful services to both humans and software clients.

This decentralised nature of the Semantic Web, much like that of the Web, is one of its most fascinating characteristics. But for developers of Semantic Web applications, automatically finding relevant sources of information is a big challenge: *how and where to find statements about certain resources?*

This paper introduces Sindice, a scalable online service that addresses exactly this question. Sindice crawls the Semantic Web and indexes the resources encountered in each source. A simple API then offers to Semantic Web application developers the ability to automatically locate relevant data sources and integrate the data from these sources into their applications.

As shown in Fig. 1, Sindice collects RDF documents from the Semantic Web and indexes these on resource URIs, IFPs and keywords. The figure shows some example RDF documents that mention Tim Berners-Lee, either by using his URI directly or by using inverse functional properties (IFPs) that uniquely identify him. Sindice offers a user interface through which human users can find these documents, based on keywords, URIs, or IFPs. More importantly, Sindice allows Semantic Web agents and clients such as Disco<sup>1</sup> to retrieve and integrate these

<sup>1</sup> <http://www4.wiwiw.fu-berlin.de/rdf-browser/>



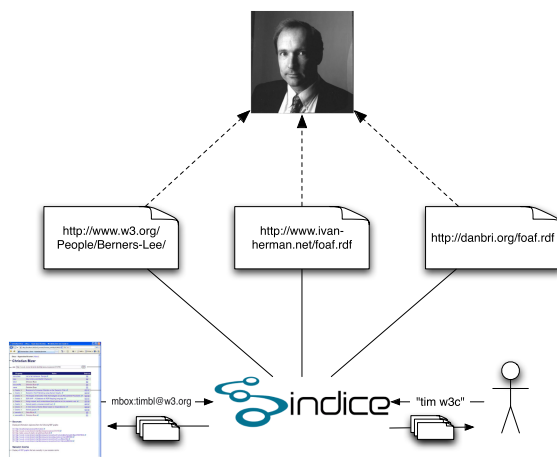


Fig. 1. Linking disparate information on the Semantic Web

results into a unified information corpus for their users. Note that Sindice may return sources regardless of whether they reference each other; Sindice may also return documents that use different identifiers for the same concept, using inverse functional properties for consolidation.

### 1.1 Motivation: Data and Linked Data

The amount of semantically structured data available on the Semantic Web has recently grown considerably. Large and important data collections, e.g. DBLP, Wikipedia, CiteSeer, SwissProt, Geonames, are now available as retrievable RDF datasets or SPARQL query endpoints. Projects such as Bio2RDF eau *et al.*(2007)Belleau, Nolin, Tourigny, Rigault *et al.*1 (e) are providing real time translation and harmonization of identifiers over vast amounts of large bioscience databases. Moreover, there is a clear trend toward embedding more semantic information in conventional web pages with techniques such as GRDDL and RDFa.

These developments make the Semantic Web a practical reality in terms of open availability of significant data. But availability of data and syntactic compatibility (e.g. RDF) is just a first step toward implementing the vision of the Semantic Web as an open and world-wide distributed source of knowledge. The next step is a Semantic Web of combined and interconnected datasets, or as an alternative, of client applications which can see such data as interconnected. Interlinked datasets with common vocabularies are not yet widespread, as shown by and Finin(2006)n (i) in their study of maintenance and reuse of Semantic Web ontologies.

Several projects<sup>2</sup> promote the interlinked nature of data, whose main principles are, (i) that all items should be identified using URI references (instead of blank nodes); (ii) that all URI references should be resolvable on the Web

<sup>2</sup> <http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>

to RDF descriptions; and (iii) that every RDF triple should be interpreted as a hyperlink to be followed by Semantic Web browsers and crawlers (Lee(2006) (ermann *et al.*(2007)Sauer mann, Cyganiak, and Völkel); u (as *et al.*(2006)Miles, Baker, and Swick); 1 (i).

The linked data approach relates a resource URI to a resolvable web address making the creator of the identifier the natural “official” source of information about the resource. On the one hand this approach creates the preconditions for successful Semantic Web crawling of the Semantic Web by applications and spiders. Also, it fits scenarios where entities have a clear official ownership such as personal FOAF profiles.

But on the other hand, the linked data approach alone is not sufficient to locate all relevant information about a resource, exactly because it only leads to “official” information. As a parallel in current Web search, we may consider a user looking for information about a particular mobile phone: not only the information linked from the producer’s homepage is interesting but also other opinions from the audience at large. To aggregate, relate and link disparate sources together that provide information about the same resources, we need for a crawled overview of information from the whole (Semantic) Web.

## 1.2 Usage Scenario

Sindice, online at <http://sindice.com>, allows its users to find documents with statements about particular resources. Sindice is in the first place not an end-user application, but a service to be used by any decentralised Semantic Web client application to locate relevant data sources. As an application service Sindice can be accessed through its Web API, for human testing and debugging we also offer an HTML front-end.

Fig. 2 displays the results of searching for the URI of Tim Berners-Lee as displayed on the HTML interface. The application interface returns the same results but in various machine-processable formats such as RDF, XML, JSON and plain text, an example is shown in Listing 1.1. In this example, several documents are returned, each of which mentions Tim Berners-Lee’s URI. The results are ranked in order of general relevance and some further information is given to enable users to choose their preferred source.

**Listing 1.1.** Documents mentioning Tim Berners-Lee (RDF/XML)

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<rdf:RDF xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description rdf:about="http://www.w3.org/People/Berners-Lee/card#i">
    <rdfs:seeAlso rdf:resource="http://www.w3.org/People/Berners-Lee/card"/>
    <rdfs:seeAlso rdf:resource="http://danbri.org/foaf.rdf"/>
    <rdfs:seeAlso rdf:resource="http://heddley.com/edd/foaf.rdf"/>
    <rdfs:seeAlso rdf:resource="http://www.eyaloren.org/foaf.rdf"/>
    <rdfs:seeAlso rdf:resource="http://people.w3.org/simon/foaf"/>
    <rdfs:seeAlso rdf:resource="http://www.ivan-herman.net/foaf.rdf"/>
  </rdf:Description>
</rdf:RDF>
```

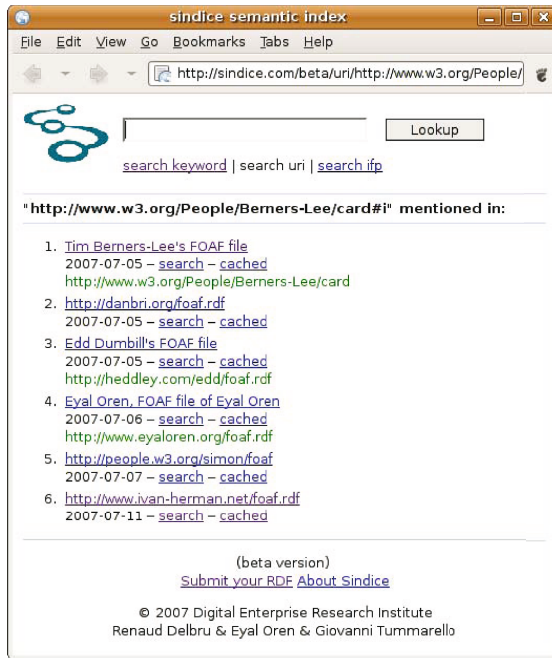


Fig. 2. Searching for documents mentioning Tim Berners-Lee (Web interface)

Sindice enables Semantic Web clients such as Piggy Bank *h et al.*(2007)Huynh, Mazzocchi, and Kargery (u) or Tabulator *ers-Lee et al.*(2006)Berners-Lee, Chen, Chilton, Connolly *et al.*r (e) to find documents with information about a given resource, identified through an explicit URI, an inverse functional property or a keyword search. This capability fits well on top of many existing Semantic Web clients. The immediate use for Sindice inside such clients is to enable a “find out more” button, to be shown next to the available information about a resource.

Upon pressing that button, the client would contact Sindice for a ranked list of documents with more information about the resource. The user would be presented with a ranked list of these documents including a human-readable source description. The user could then choose the sources of interest (or those considered trustworthy), after which the client application could import the information from these documents. The user could maybe also select to “always consider these domains as providers of good information” to allow fully automated information import during subsequent lookups.

For clients that implement the linked data principles, integration with Sindice is trivial. Sindice behaves as a “good citizen” of the linked data Web: it provides all results as RDF that themselves follow the “linked data” principles. For example, Fig. 3 shows results from Sindice in the Disc<sup>3</sup> Semantic Web browser: all resulting documents are browseable resources themselves and can easily be followed.

<sup>3</sup> [http://www4.wiwiss.fu-berlin.de/rdf\\_browser/](http://www4.wiwiss.fu-berlin.de/rdf_browser/)

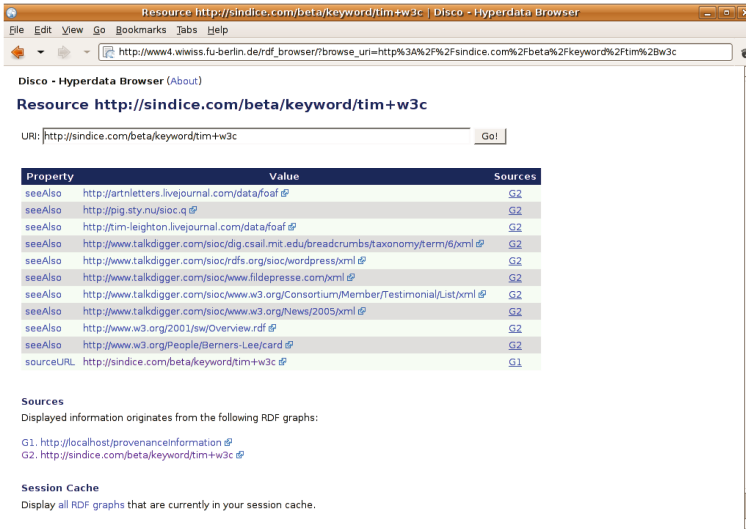


Fig. 3. Application integration: Sindice results in the Disco browser

While Sindice supports, uses, and promotes the linked data model (namely in its crawling and ranking), it also supports locating information about URIs that are not URLs and cannot be de-referenced such as telephone numbers or ISBN numbers. But most importantly, Sindice helps locating statements about resources made outside their “authoritative” source.

### 1.3 Design Principles

Sindice only acts as locator of RDF resources, returning pointers to remote data sources, and not as a query engine. Sindice is thus conceptually more close to standard Web search engines but with specific Semantic Web concepts, procedures and metrics, rather than to Semantic Web search engines such as SWSE *n et al.*(2007)Hogan, Harth, Umbrich, and Deckerg (o) or Swoogle *n et al.*(2005)Finin, Ding, Pan, Joshi *et al.*n (i) which in general aim at providing general query capabilities over the collections of all the Semantic Web statements.

By only providing pointers to sources, Sindice can avoid many of the complex issues which Semantic Web search engines must face. These issues include trust, global entity consolidation policies, voluntary or involuntary denial of services by queries of excessive complexity or on excessive data, etc. While difficult at global scale, such problems are easier to handle on the application level, e.g. by using direct user interaction, domain-specific policies or heuristics. Sindice’s design supports applications that give the user full control over the considered data sources: only the information from explicitly appointed sources is used.

For this class of applications, a simple API as offered by Sindice is probably most of what is needed to connect to the global Semantic Web, without relegating control over which data sources to consider and which to ignore.

## 2 Sindice Architecture

This section introduces the functional and non-functional requirements on the Sindice architecture and analyses whether building such a service is technically feasible on commodity hardware.

### 2.1 Requirements

The requirements for Sindice can be divided in functional and non-functional requirements. In terms of base functionality, Sindice offers three services to client applications: (i) it parses files and SPARQL endpoints while crawling or when “pinged” explicitly; (ii) it looks up resources (identified by their URI or by a combination of an inverse-functional property and identifying value) and returns URLs of RDF documents where these resources occur; and (iii) it searches full-text descriptions and returns the URLs of sources in which these resources occur. To fulfil these requirements, the abstract Sindice API thus consists of four methods:

- `index(url) => nil`: parses and indexes document or SPARQL endpoint at given URL,
- `lookup(uri) => url[]`: looks up a resource with given URI, returns a ranked list of sources in which that resource occurs,
- `lookup(ifp, value) => url[]`: looks up a resource uniquely identified with property-value pair, returns a ranked list of sources in which that resource occurs,
- `lookup(text) => url[]`: looks up a textual query, returns a ranked list of sources in which the given terms occur.

Additionally, we have three non-functional design requirements. First, we want to minimise the index size, so as to allow indexing of the whole (current) Semantic Web on a single commodity node without networked storage or disk-arrays. Secondly, we want to minimise lookup times, so as to allow applications to use Sindice by default to lookup more information for any encountered resource. Thirdly, we want to allow continuous live updates of the index so as to keep the index up-to-date.

### 2.2 Architecture Design

The architecture consists of several independent components that operate in several pipelines to achieve crawling, indexing, and querying. Each pipeline will be discussed in detail in Section 3. Here we briefly introduce the overall architecture, as shown in Figure 4.

The Web frontend is the main entry point, divided in a user interface for human access and an HTTP API for machine access. Then, there are several components for crawling and indexing RDF documents. The crawler autonomously harvests RDF data from the Web and adds it to the indexing queue. If pinged (through the human interface or the API) to parse new documents, these are

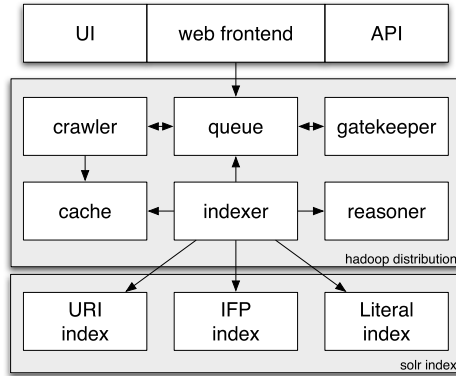


Fig. 4. Sindice architecture

also added to the queue. The gatekeeper evaluates each entry in the queue and decides whether, and with which priority, we want to index it, based on whether we have seen the document before, its last modification date, its content digest, etc. The indexer extracts URIs, IFPs and keywords from each document (using the reasoner for IFP extraction) and adds these to their respective index. During lookup, the interface components only need to pass the queries to the relevant index, gather the results, and generate the required output such as HTML pages with appropriate layout.

The three indices store occurrences of resource URIs, resource IFPs and literals in RDF documents. The URI index contains an entry for each resource URI that lists the document URLs where this resource occurs. The IFP index is similar, except that instead of explicit resource URIs, the uniquely identifying pair (*property, value*) is used as index key, again pointing to a list of document URLs where this pair occurs. This index allows lookup of resources with different URIs that actually identify the same real-world thing. The literal index contains an entry for each token (extracted from the literals in the documents), again pointing to a list of document URLs.

In designing the index, we optimise for disk space and lookup times. Since the only required access pattern is from resource to mentioning sources, an inverted index of URI occurrences in documents is a natural structure. In general, lookup on such an index can be performed in almost constant time over the size of the index.

### 2.3 Feasibility

Before detailing the internals of Sindice, we analyse its feasibility. We analyse a representative sample of Semantic Web data and analyse graph-theoretical properties that allow us to predict the required index size using inverted index structures.

As an indication of required index size, we aim to store, on a single commodity machine, at least a billion unique resources. To predict the project index size for indexing such resources, we have crawled Semantic Web data for around four weeks and collected some 3.2 million unique resources. Our crawl seems a representative collection of Semantic Web data: the SWSE search engine currently contains around 11 million unique URIs excluding blank nodes<sup>4</sup>, whereas Finin(2006)n (i) estimated the Semantic Web to contain around 10 million documents in August 2006.

The required index space in terms of indexed resources depends primarily on how often each resource is mentioned, i.e. the ratio between resources (URIs) and documents (URLs). We analysed how often the same URIs were mentioned across different documents, which is plotted in Fig. 5. The left graph shows a zoomed result in log-scale, the right graph shows the complete data in log-log-scale. These graphs demonstrate that distribution (reuse) of URIs over documents follow a power-law and therefore exhibit scale invariance. This scale-free property means that the ratio of URIs/URLs will remain constant as the Semantic Web grows which means that we can estimate the average number of times a resource will be mentioned independent of the size of the Semantic Web.

The power law found in URI occurrences is not surprising since choosing URIs is a social process and thus prone to properties such as preferential attachment that result in scale-free networks. Our result corresponds with an earlier analysis and Finin(2006)n (i) showing other properties of Semantic Web graphs to follow power-law behaviour as well.

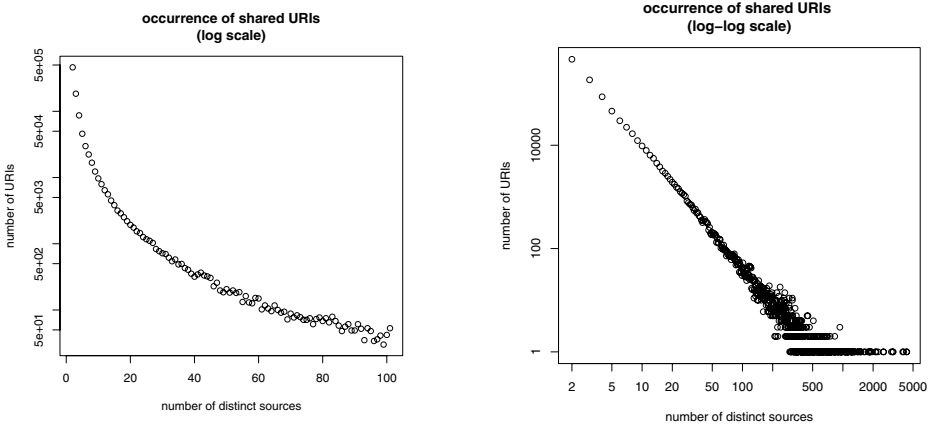


Fig. 5. Occurrence of same resources (URIs) in different documents

With respect to feasibility, the index size of our crawl on the simple persistent hashtable of URI occurrences was around 2.5GB for 3.2 million URIs. Given the scale-invariance of the URI/URL ratio we can extrapolate from this data

<sup>4</sup> Personal communication with A. Hogan.

and estimate to need around 785 bytes per resource; indexing a billion unique resources would thus require around 785GB, an ordinary capacity for commodity harddisks.

### 3 Inside Sindice

The Sindice service indexes RDF graph and then enables users or Semantic Web applications to find the location of an sources through querying. We conceptualise these two tasks by two pipelines: an *indexing* pipeline and a *querying* pipeline.

#### 3.1 Indexing Pipeline

The indexing pipeline performs a sequence of tasks, described in Fig. 6, to index an RDF graph.

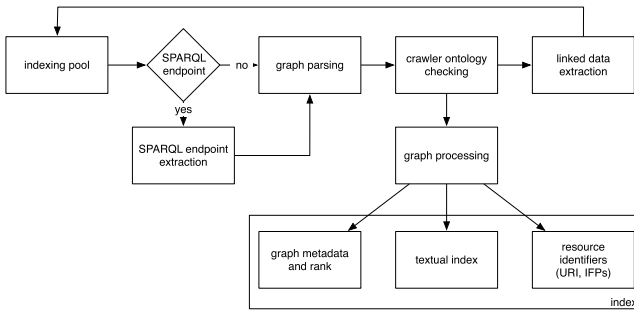


Fig. 6. Overview of the indexing pipeline

**Scheduler.** The indexing pipeline takes as input an RSS feed or an external ping that specifies an RDF graph location. The URL, corresponding to the RDF graph location, is injected into the scheduler. The scheduler acts as a collector of URLs waiting to be processed and avoid overloading Web servers. The scheduler maintains a small hashtable with metadata for each visited source (e.g. visiting time and hash). We only revisit sources after a threshold waiting time and we only reparse the source’s response if the content hash has changed.

**Graph extraction.** The URLs in the scheduler can be of two types: RDF resources and SPARQL end points. In the first case, we retrieve the file and send it to the parser. In the case of a SPARQL endpoint, we send to the database one or more queries to extract its full content.

**Graph parsing.** Every time an RDF graph is retrieved successfully, it is sent to the graph parser. The parser first verifies the validity of the RDF graph. It then extracts all URIs from the graph and injects them into the scheduler, following the linked data principle to treat every URI as a hyperlink to more information.



**Graph processing.** The graph processor then extracts and indexes the full-text and all resource identifiers in the graph. We also extract indirect identifiers, namely pairs  $(p, o)$  for all inverse functional properties that are mentioned in the graph. Extracting these IFPs requires two additional steps: the recursive expansion of the graph with all its schema information, followed by OWL inferencing to identify all IFPs in these schemas (described below).

*Finding inverse functional properties.* A graph typically does not explicitly mention which of its properties are inverse functional; rather, the graph would refer to one or more schema definitions, which may recursively depend on other schemas.

We therefore proceed as follows: we fetch the definition of each property, following the linked data principle, by de-referencing its URI and importing the returned definition into the RDF graph. This “schema fetching” task is repeated recursively until fixpoint. At this point we perform inferencing using the OWL “ter Horst” fragment `orst(2005)r (e)`. This fragment is expressive enough for finding inverse functional properties and has efficient entailment procedures. After this reasoning step, we query the combined model for IFPs and proceed with the creation of identifiers as explained above.

*Reasoning cache.* To improve the graph processing we cache the reasoner’s output. Since reasoning over a large schema to find all inverse functional properties is computationally expensive it could quickly form an indexing bottleneck. However, the reasoning results need to be separated for each document, to prevent malicious users from “infecting” results on a global scale by defining a property to be inverse functional (while it is not).

The graph processor therefore uses a caching system based on the set of properties explicitly used in a graph. For example, if a FOAF profile mentions some properties and classes such as `foaf:name`, `foaf:Person`, we only perform the graph reasoning if that exact set of properties and classes has not occurred before. Then, after graph expansion and identifying the set of IFPs (e.g. `foaf:mbox` and `foaf:personalHomepage`) we cache the found IFPs using the set of properties and classes in the original document as cache key.

### 3.2 Querying Pipeline

The querying pipeline is on the other hand splitted into three main stages: the index retrieval, the ranking phase and the result generation.

**Index retrieval.** The query is looked up in the inverted index, which can be implemented either as an on-disk hashmap or in an information retrieval engine. The list of results is cached for later reuse, and is invalidated daily to keep the result up-to-date.

**Ranking phase.** After index retrieval the results are ranked according to various metrics. Since for popular resources our index could easily return many hundreds or thousands sources, providing a ranked list is crucial.

We have designed a ranking function that requires only little metadata for each source and is relatively fast to compute; it does not construct a global ranking of all sources but ranks sources based on their own metadata and external ranking services. The following metadata values for each source are computed and their value combined using an unweighted average:

- Hostname: we prefer sources whose hostname is the same as the resource’s hostname, in support of the linked data paradigm. For example, we consider that more information on the resource <http://eyaloren.org/foaf.rdf#me> can be found at the source <http://eyaloren.org/foaf.rdf> than at an arbitrary “outside” source, such as <http://g1o.net/g1ofoaf.rdf>.
- External rank: we prefer sources hosted on sites which rank high using traditional Web ranking algorithms. These rankings can be purchased or can be estimated using various techniques and Dhillon(2006)v (a).
- Relevant sources: we prefer sources that share rare terms (URIs, IFPs, keywords) rather than common terms with the requested terms. This relevance metric is comparable to the TF/IDF relevance metrics and Baeza-Yates(1992)a (r) in information retrieval.

**Result generation.** Once the resulting data sources are ranked into order of importance, Sindice can export them into different syntaxes, such as the HTML Web interface, RDF, XML, JSON, and plain text.

### 3.3 Crawling Ontology

On February the 2nd 2007, Geonames experienced what has been called “the first distributed denial of service on the Semantic Web”<sup>5</sup>. What happened, however, was not due to malicious behaviour but rather due a Semantic Web Crawler simple a bit too fast in following the linked data paradigm and resolving each of the 6.4 million URL/URI on the server. As the result of each one of these calls is a query, the load on the server was in the end very high leading to the denial of service. It has been suggested that this could have been avoided if the geoname database RDF dump, which was in fact being made available, had been imported as an alternative to full site crawling. But how could the spider have known that?

Our crawling ontology<sup>6</sup> helps Semantic Web spiders and clients alike in indexing and operating over large quantity of Semantic Linked data. Through this ontology, a site administrator can avoid denial of services and ensure that the data will be used in the form which is optimal for the task.

The crawling ontology augments the existing `robots.txt` protocol; all relevant statements should be located in a `srobots.rdf` file, on the site root as done in the `robots.txt` file. The main class to be used is the `DataEquivalenceStatement`, which states that data represented in three possible ways (linked data, SPARQL

<sup>5</sup> <http://geonames.wordpress.com/2007/02/03/friendly-fire-semantic-web-crawler-ddos/>

<sup>6</sup> <http://sindice.com/srobotsfile>

endpoint, or data dump) is equivalent, allowing the client to choose one of these representations

To follow the protocol, a client retrieving a resource URL should first check for the traditional `robots.txt`; next, it should check for the existence of `srobot.rdf`, to see whether the URL is maybe part of a larger dataset that can be downloaded instead. To increase awareness of the `srobot.rdf` file, a link to it can be also provided as a "see also" statement both in the RDF data dump as a description of the file location URL itself and in the RDF returned by resolving the linked data URLs. Multiple Data Equivalence Statements can be used as necessary for different data sets served by the same host.

At the time of writing, we are working on an updated version of this ontology after feedback from RDF data providers. The new version of this ontology<sup>7</sup> covers the material discussed above as an extension of the Sitemap protocol<sup>8</sup> for web document discovery.

## 4 Related Work

We are aware of two Semantic Web search engines that index the Semantic Web by crawling RDF documents and then offer a search interface over these documents. SWSE<sup>9</sup> crawls not only RDF documents but also "normal" HTML Web documents and RSS feeds and converts these to RDF *et al.*(2007)Harth, Umbrich, and Deckerr (an *et al.*(2007)Hogan, Harth, Umbrich, and Decker); g (o). SWSE stores all triples found in the crawling phase including their provenance and offers rich queries, comparable to SPARQL, over these quads. Similarly, Swoogle *n et al.*(2005)Finin, Ding, Pan, Joshi *et al.*n (i) crawls and indexes the Semantic Web data found online.

Some differences between these engines and Sindice have already been highlighted during this paper. To the best of our knowledge, none of these engines seem to display continuous crawling capabilities, probably due to the cost and complexity of updating an index which can answer relational queries. Also, although SWSE allows IFP lookups in its query capabilities, it does not perform reasoning to extract these IFPs but instead extracts only several hardcoded properties. Finally, none of these engines provide indexing based on "linked data" paradigm reasoning, SPARQL endpoint indexing and the ability to index large repositories consciously through the Sitemap extension.

Table 1 shows an overall comparison of our approach against on the one hand traditional Web search engines such as Google or Yahoo! and on the other hand Semantic Web (SW) search engines such as SWSE or Swoogle. Whereas traditional Web search focuses on document retrieval for HTML documents, and SW search focuses on building a global database of retrieved triples, we provide a document retrieval service for RDF documents. Sindice is thus conceptually

<sup>7</sup> <http://sw.deri.org/2007/07/sitemapextension/>

<sup>8</sup> <http://www.sitemaps.org/protocol.html>

<sup>9</sup> <http://swse.deri.org/>

**Table 1.** Approaches in (Semantic) Web information retrieval

	Web search	SW search	Sindice
focus	document retrieval	global database	SW document retrieval
orientation	Web documents	triples/quads	RDF documents
URI lookup	-	+	+
IFP lookup	-	±	+
scalability	+	±	+
full queries	-	+	-
SPARQL indexing	-	-	+

close to traditional Web search engines but employs different ways of finding documents and indexes more than only natural language texts).

Finally, our service is related to <http://pingthesemanticweb.com>, which maintains a list of recently updated documents and currently lists over seven million RDF documents. The service does not perform indexing and does not allow lookups over its content, it does offer a periodical dump of updated documents. We see the service as a companion to Sindice and we in fact use it as input to our crawling pool.

## 5 Conclusion

We have presented Sindice, a public API to locate the sources of Semantic Web annotations, be these RDF files or SPARQL endpoints. By committing to this simple service Sindice provides a much needed interweaving framework for the Semantic Web achieving very high scalability while maintaining overall neutrality on issues such as trust, reputation, ontologies and identifiers. Such neutrality is key to the use of Sindice regardless of the needs and purpose of the Semantic Web client: clients will be free to chose their sources according to their preferences, requirements and possibly aided by the assistance and supervision of the end user.

Apart from the Sindice service as such, the contributions of this paper are: a crawling ontology which can avoid the high inefficiency of blindly navigating sites that support the “linked data” paradigm; a strategy for calculation of IFPs as found in crawled RDF files ; a simple strategy to locate human-readable descriptions of RDF files and their web ranking; a result strategy strategy and finally the discovery of a power-law behaviour regarding URI reuse on the Semantic Web.

In future work, we aim to provide a Sindice client-side library that implements useful procedures such as smushing or equality reasoning, using “linked data” practices and involving potentially many Sindice calls. We are also working on a distributed index, and potentially a distributed pingging architecture as well. Finally, Sindice services could be provided over UDP, removing the time needed for TCP handshake to provide a more responsive service.

*Acknowledgements.* This material is based upon works supported by the Science Foundation Ireland under Grants No. SFI/02/CE1/I131 and SFI/04/BR/CS0694.

## References

- Belleau, F., Nolin, M.-A., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: Towards A Mashup To Build Bioinformatics Knowledge System. In: Proceedings of the WWW Workshop on Health Care and Life Sciences Data Integration for the Semantic Web (2007)
- Berners-Lee, T.: Linked Data. W3C Design Issues (2006)
- Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator: Exploring and analyzing linked data on the Semantic Web. In: Proceedings of the ISWC Workshop on Semantic Web User Interaction (2006)
- Davis, J.V., Dhillon, I.S.: Estimating the global pagerank of web communities. In: KDD. Proceedings of the International Conference on Knowledge Discovery and Data Mining, pp. 116–125 (2006)
- Ding, L., Finin, T.: Characterizing the Semantic Web on the web. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
- Finin, T.W., Ding, L., Pan, R., Joshi, A., et al.: Swoogle: Searching for knowledge on the semantic web. In: AAAI 2005. Proceedings of the National Conference on Artificial Intelligence (2005)
- Frakes, W.B., Baeza-Yates, R.A.: Information Retrieval: Data Structures & Algorithms. Prentice-Hall, Englewood Cliffs (1992)
- Harth, A., Umbrich, J., Decker, S.: Multicrawler: A pipelined architecture for crawling and indexing Semantic Web data. In: ISWC 2007. Proceedings of the International Semantic Web Conference (2007)
- Hogan, A., Harth, A., Umbrich, J., Decker, S.: Towards a scalable search and query engine for the web (Poster presentation). In: Proceedings of the International World-Wide Web Conference (2007)
- ter Horst, H.J.: Combining RDF and part of OWL with rules: Semantics, decidability, complexity. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 668–684. Springer, Heidelberg (2005)
- Huynh, D., Mazzocchi, S., Karger, D.: Piggy bank: Experience the Semantic Web inside your web browser. *Journal of Web Semantics* 5(1), 16–27 (2007)
- Miles, A., Baker, T., Swick, R.: Best Practice Recipes for Publishing RDF Vocabularies. In: W3C Working Draft (2006), <http://www.w3.org/TR/swbp-vocab-pub/#redirect>
- Sauermann, L., Cyganiak, R.: M. Völkel. Cool URIs for the Semantic Web. Tech. Rep. TM-07-01, DFKI (2007)

# *combiSQORE*: An Ontology Combination Algorithm

Rachanee Ungrangsi<sup>1</sup>, Chutiporn Anutariya<sup>1</sup>, and Vilas Wuwongse<sup>2</sup>

<sup>1</sup> School of Technology, Shinawatra University  
99 Moo 10 Bangtoey, Samkok, Pathum Thani, 12160 Thailand  
{rachanee, chutiporn}@shinawatra.ac.th

<sup>2</sup> School of Engineering and Technology, Asian Institute of Technology  
P.O. Box 4, Klong Luang, Pathum Thani, 12120 Thailand  
vw@cs.ait.ac.th

**Abstract.** Automatic knowledge reuse for Semantic Web applications imposes several challenges on ontology search. Existing ontology retrieval systems merely return a lengthy list of relevant single ontologies, which may not completely cover the specified user requirements. Therefore, there arises an increasing demand for a tool or algorithm with a mechanism to check concept adequacy of existing ontologies with respect to a user query, and then recommend a single or combination of ontologies which can entirely fulfill the requirements. Thus, this paper develops an algorithm, namely *combiSQORE* to determine whether the available collection of ontologies is able to completely satisfy a submitted query and return a single or combinative ontology that guarantees query coverage. In addition, it ranks the returned answers based on their conceptual closeness and query coverage. The experimental results show that the proposed algorithm is simple, efficient and effective.

## 1 Introduction

Ontology is employed as a means for knowledge sharing and reusing in the *Semantic Web* [3]. References [4, 9, 13] discuss two typical scenarios for ontology reuse in the Semantic Web. The first one envisions that a user expresses his/her requirements as a query and submits it to an ontology search engine to retrieve the most appropriate ontology. If the returned result partially satisfies the user requirements, the user is then required to make additional modification efforts which are considerably less compared to those needed to construct a new ontology from scratch.

On the other hand, the second scenario, which is called *automatic knowledge reuse*, addresses the problem of automatically and dynamically finding a single or combinative ontology for *next generation Semantic Web applications* [11], such as Magpie [7] and PowerAqua [9, 13]. Magpie [7] is a semantic browser which assists users while they surf the Web by highlighting instances of chosen concepts in the current Web page based on an internal instantiated ontology. The second application, PowerAqua [9, 13], is an ontology based question answering system that derives answers to questions asked in natural language by exploiting an underlying ontology. Currently, in both tools, the employed ontology is manually selected by the user and only one ontology can be exploited at a time. To allow cross-domain question

answering in the case of PowerAqua, and enable an extended coverage of the semantic browsing with Magpie, a mechanism for dynamically finding and combining the relevant knowledge among online ontologies and semantic data becomes essential.

Existing ontology retrieval systems, such as *Swoogle* [6], *OntoKhoj* [11], and *OntoSearch* [16], merely return a lengthy list of single ontologies, but none of them can ensure that all query conditions are met by at least one of the returned results. Furthermore, due to the sparseness of knowledge in a Web-accessible ontology database, it is possible that there exists no single ontology which satisfies all user requirements [13]. However, to date there is no algorithm or tool which can deal with these significant complications.

This paper proposes a simple yet efficient and effective algorithm, namely *combiSQORE*. It does not only enable users to check the concept sufficiency of an ontology collection with respect to a given query, but also computes a sub-optimal combination of ontologies that jointly cover the query when no single ontology can fulfill the specified requirements. In addition, it returns the rankings which rank both single and combinative ontologies based on *conceptual closeness* and *query coverage*.

*combiSQORE* algorithm is developed as an extension of *SQORE* (*Semantic Query based Ontology Retrieval Framework*) [2, 14]. *SQORE* enables users to precisely and structurally formulate their ontology requirements in terms of a *semantic query*. Each query is evaluated by considering the semantic closeness between the query itself and the resultant ontology which is quantified by *SQORE*'s *similarity measures*. Comprehensive experiments have been conducted on real-world ontologies to evaluate and demonstrate *combiSQORE*'s effectiveness. The results have shown that the proposed algorithm can generate irreducible combinations of ontologies with a reasonable cost and provide useful rankings.

The paper is organized as follows. Sect. 2 reviews related works and Sect. 3 informally introduces *SQORE*. Sect. 4 develops *combiSQORE* algorithm and Sect. 5 illustrates the algorithm via an example. Sect. 6 discusses the conducted experiments and their results, and followed by conclusions and future work in Sect. 7.

## 2 Related Work

Ontology search engines are crucial to enable scientists and practitioners to find and reuse Web-accessible ontologies efficiently. Several ontology retrieval systems have been developed in the last few years (e.g. *Swoogle* [6], *OntoKhoj* [12], and *OntoSearch* [16].) However, these systems mainly focus on automatically crawling the Web for collecting ontologies and employ traditional keyword search mechanisms to retrieve relevant ontologies. As a result, they fail to capture the structural and semantic information about the user-desired domain concepts and relations. Furthermore, they usually return a large number of ontologies, but cannot guarantee query coverage which is a mandatory requirement for *automatic ontology reuse* in Semantic Web applications, such as an ontology-based browser *Magpie* [7], an ontology-based question answering system *PowerAqua* [9, 13], etc.

Another interesting approach is *CORE* [8] and its extension, *WEBCORE* [4], which retrieves keyword-related ontologies from an ontology database, and applies multiple criteria to generate several rankings, and finally combines all the rankings to

obtain the final ranking. However, some of these ranking criteria require users to provide applications and data for the evaluation. Furthermore, in its last step, a user is demanded to manually evaluate the resultant ontologies in order to enable a collaborative assessment. Thus, this approach cannot readily be applied to automatic ontology reuse in Semantic Web applications.

*Swoogle* [6] and *OntoKhoj* [12] implement their *PageRank*-like algorithms based on the computed ontology referral network. *ActiveRank* [1] introduces several metrics for ontology ranking based on the taxonomic structure information such as class names, shortest paths, linking density and positions of focused classes in the ontology. However, these three approaches cannot be used for ranking the returned result that consists of both single and combinations of ontologies.

*PowerAqua* [9, 13] proposes a framework to determine ontology combinations for a given query by using *OntoCombination* algorithm and compute ranking based on the generality of ontology concepts. However, such an algorithm produces a set of ontologies ranked by the coverage of each individual ontology, but does not compute an optimal or sub-optimal combination that maximizes the query coverage.

### 3 SQORE: Architectural Overview

Fig. 1 illustrates *SQORE*'s system architecture [2, 14] which comprises four main components: i) a *semantic query*, ii) a *retrieval engine*, iii) an *ontology database*, and iv) a *semantic lexical database*. It employs *XML Declarative Description (XDD) theory* [15] as its theoretical foundation for modeling ontology databases and evaluating semantic queries, which does not only facilitate ontology matching and retrieval, but also support reasoning capability to enhance the matching results. Furthermore, when a query term and an ontology term do not exactly match ( $=$ ), it determines other possible semantic relations between them (i.e. equivalence ( $\equiv$ ), broader ( $\supseteq$ ), narrower ( $\subseteq$ ) and unknown ( $\neq$ )) by employing a referenced lexical database, such as *WordNet* [10]. Then, the system computes the *semantic similarity score* between a given query and an ontology in the collection, which ranges from 0 (strong dissimilarity) to 1 (strong similarity).

By enhancing *SQORE* with the proposed *combiSQORE* algorithm, the system can then determine whether or not an ontology collection is conceptually sufficient for a user query, and recommend a single or combinative ontology which completely cover

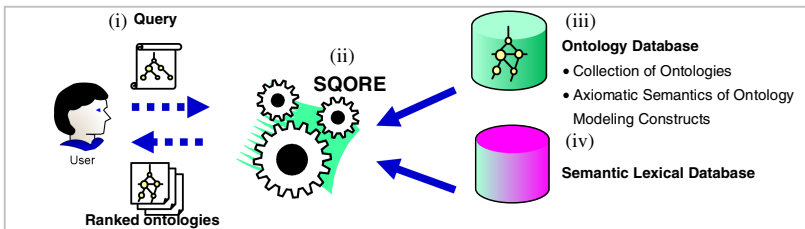


Fig. 1. SQORE System Architecture



the query. Finally, the system computes *semantic similarity scores* between the query and the returned ontologies (either single or combinative) based on *conceptual similarity*, and *query coverage* and uses these scores for the rankings.

SQORE defines four measures used for calculating similarity scores as follows:

- **Element Similarity Score ( $SS_E$ ):** The similarity score of any two given elements  $x$  and  $y$ , denoted by  $SS_E(x, y)$ , depends on their semantic relation determined by the referenced lexical database as explained earlier. For any two given restrictions  $r(a_1, b_1)$  and  $r(a_2, b_2)$ , their similarity is equal to the product of  $a_1$ - $a_2$  similarity score and  $b_1$ - $b_2$  similarity score i.e.,  $SS_E(a_1, a_2) * SS_E(b_1, b_2)$ . When  $x$  and  $y$  do not belong to the same type, for instance  $x$  is a class name and  $y$  a property name, their similarity score is undefined.
- **Best Similarity Score ( $SS_B$ ):** Based on the element similarity score  $SS_E$ ,  $SS_B(x, O)$  represents the similarity between a given element  $x$  of a query and an ontology  $O$  by finding the highest similarity score between  $x$  and each element  $y$  that is semantically defined by  $O$ . In other words, the element  $y$  in  $O$  that is most similar to  $x$ , will be used for measuring the closeness between  $x$  and  $O$ . This measure is a key metric in the combiSQORE algorithm.
- **Satisfaction Score of Mandatory conditions ( $SS_M$ ) and Optional conditions ( $SS_O$ ):** In SQORE, a semantic query comprises mandatory conditions and optional conditions. If an ontology semantically satisfies all mandatory conditions of a given query, then that ontology will be included in the answer. Optional conditions, on the other hand, are useful for expressing additional means for measuring the extent of closeness between the ontology and the query.
- **Query-Ontology Similarity Score (SS):** This similarity score represents the semantic closeness between a query and an ontology, which is measured by the satisfaction degree of the ontology with respect to the mandatory and optional conditions of the query.

## 4 Algorithms: Ontology Combination and Ranking

Formally, the problem of finding an ontology combination is: *Given a semantic query and a set of ontologies, determine a minimal ontology subset that satisfies all conditions in the query, and maximizes the conceptual closeness between the ontology subset and the query.* This problem is equivalent to the *knapsack problem*, which is widely-known to be NP-complete. Therefore, rather than developing an optimal solution, this paper proposes a backward greedy algorithm for construction of an *irreducible* ontology subset, which satisfies all conditions in the query.

### 4.1 Notations and Definitions

Throughout this section, let  $\mathcal{ODB} = \{O_1, \dots, O_n\}$  be an ontology collection consisting of  $n$  ontologies and  $\mathcal{Q} = \{q_1, \dots, q_m\}$  be a semantic query comprising  $m$  conditions. As means for measuring the relevance of an ontology  $O$  in  $\mathcal{ODB}$  with respect to a condition  $q$  of  $\mathcal{Q}$ , SQORE [2, 14] defines  $SS_B(q, O)$  as the (best) similarity score between  $q$  and  $O$ , which ranges from 0 (strong dissimilarity) to 1 (strong similarity).

Based on  $SS_B(q, O)$ , let  $S(q, \mathcal{ODB}) \subseteq \mathcal{ODB}$  be the set of ontologies relevant to a condition  $q$ , defined as follows:

$$S(q, \mathcal{ODB}) = \{ O \in \mathcal{ODB} : SS_B(q, O) > 0, q \in \mathcal{Q} \} \quad (1)$$

**Definition 1.** An ontology collection  $\mathcal{ODB}$  is *sufficient* to satisfy a semantic query  $\mathcal{Q}$  if and only if

$$\forall q \in \mathcal{Q}, S(q, \mathcal{ODB}) \neq \emptyset. \quad \square$$

Intuitively speaking, if  $S(q, \mathcal{ODB})$  is the empty set, one can derive that there exists no ontology in  $\mathcal{ODB}$  that can satisfy such a query condition  $q$  in  $\mathcal{Q}$ . Therefore, an ontology collection  $\mathcal{ODB}$  is said to be *sufficient for* a semantic query  $\mathcal{Q}$ , if there exists a non-empty subset of  $\mathcal{ODB}$  which jointly satisfies all conditions in  $\mathcal{Q}$ ; otherwise  $\mathcal{ODB}$  is *insufficient*.

**Definition 2.** Let  $\mathcal{R} \subseteq \mathcal{ODB}$ .  $\mathcal{R}$  is a *query result* of  $\mathcal{Q}$  if  $\mathcal{R}$  is sufficient for  $\mathcal{Q}$ .  $\mathcal{R}$  is a *candidate query result* of  $\mathcal{Q}$ , if  $\mathcal{R}$  is a *query result* and *minimal (irreducible)*. That is, any subset of a candidate query result  $\mathcal{R}$  must not be a candidate query result of  $\mathcal{Q}$ , and hence removing any ontology  $O$  from  $\mathcal{R}$  leads to an unsatisfactory of some query conditions  $q$  in  $\mathcal{Q}$ .  $\square$

Next, an algorithm, namely *combiSQORE*, which can generate a candidate query result of  $\mathcal{Q}$ , is devised.

## 4.2 CombiSQORE Algorithm

Fig. 2 presents *combiSQORE* algorithm, which takes three input parameters: a semantic query  $\mathcal{Q}$ , a set of ontologies  $\mathcal{ODB}$  and a sequence  $l$ , and returns a candidate query result  $\mathcal{R}$  of  $\mathcal{Q}$ . Firstly, it determines whether or not the ontology collection  $\mathcal{ODB}$  is sufficient to satisfy  $\mathcal{Q}$ . If  $\mathcal{ODB}$  is insufficient for  $\mathcal{Q}$ , the algorithm exits and returns the empty set—no query result for  $\mathcal{Q}$ . If  $\mathcal{ODB}$  is sufficient,  $\mathcal{ODB}$  itself is a query result for  $\mathcal{Q}$ . Therefore,  $\mathcal{R}$  is initially assigned to be equal to  $\mathcal{ODB}$ . The next for-loop then minimizes  $\mathcal{R}$  by considering each ontology  $O$  in  $\mathcal{R}$  according to the input sequence  $l$ . If  $\mathcal{R} - \{O\}$  is insufficient for  $\mathcal{Q}$ ,  $O$  cannot be removed from  $\mathcal{R}$ ; otherwise  $\mathcal{R}$  is minimized by taking  $O$  out. This iteration continues until there is no ontology remaining in the sequence  $l$ . The algorithm then returns  $\mathcal{R}$  as a candidate query result.

One can see that with a different ontology sequence  $l$ , *combiSQORE* may produce different candidate query result  $\mathcal{R}$  for a particular query  $\mathcal{Q}$  and ontology collection  $\mathcal{ODB}$ , since the sequence  $l$  determines the order of removing an ontology from an initial query result in order to finally obtain a candidate query result. Note that the conducted experiments show that strategically generated input sequences can improve the algorithm performance (to be discussed in more details in Section 6).

Let  $m$  denote the size of a given query  $\mathcal{Q}$  and  $n$  the size of an ontology collection  $\mathcal{ODB}$ . The complexity of *combiSQORE* is  $O(mn^2 \log n)$  or  $O(n^2 \log n)$  when  $m \ll n$ .

```

Algorithm combiSQORE( $Q, \mathcal{ODB}, l$ )
Input:    $Q$ : a semantic query,
            $\mathcal{ODB}$ : an ontology collection,
            $l$ : a predetermined sequence of ontologies in  $\mathcal{ODB}$ 
Output:  $\mathcal{R}$ : a candidate query result
if  $\exists q \in Q$  such that  $S(q, \mathcal{ODB}) = \emptyset$ 
    do EXIT //  $\mathcal{ODB}$  is insufficient for  $Q$ 
 $\mathcal{R} = \mathcal{ODB}$ 
for each ontology  $O$  in the sequence  $l$ 
    do  $T = \mathcal{R} - \{O\}$ 
        if  $\exists q \in Q$  such that  $S(q, T) = \emptyset$ 
             $\mathcal{R} = \mathcal{R}$  //  $T$  is insufficient for  $Q$ 
        else
             $\mathcal{R} = T$  //  $T$  is sufficient for  $Q$ 
return  $\mathcal{R}$ 

```

**Fig. 2.** combiSQORE: an ontology combination algorithm

**Theorem 1.** If an ontology collection  $\mathcal{ODB}$  is *sufficient* for a given semantic query  $Q$ , then  $\mathcal{P} \supseteq \mathcal{ODB}$  is also *sufficient* for  $Q$ .

*Proof:* Assume that there exists  $\mathcal{P} \supseteq \mathcal{ODB}$  that is insufficient for  $Q$ . Then, by definition, there exist  $q \in Q$  such that

$$\begin{aligned}
 S(q, \mathcal{P}) &= \emptyset \\
 \{O \in \mathcal{P} : SS_B(q, O) > 0\} &= \emptyset \\
 \{O \in \mathcal{P} : SS_B(q, O) > 0\} \cap \mathcal{ODB} &= \emptyset \cap \mathcal{ODB} \\
 \{O \in \mathcal{P} \cap \mathcal{ODB} : SS_B(q, O) > 0\} &= \emptyset \cap \mathcal{ODB} \\
 \{O \in \mathcal{ODB} : SS_B(q, O) > 0\} &= \emptyset \quad // \text{since } \mathcal{P} \supseteq \mathcal{ODB} \\
 S(q, \mathcal{ODB}) &= \emptyset
 \end{aligned}$$

which contradicts the assumption that  $\mathcal{ODB}$  is sufficient for  $Q$ .  $\square$

**Theorem 2.** A candidate query result  $\mathcal{R}$  returned by combiSQORE is *irreducible*.

*Proof:* For the sake of contradiction, let  $X \subseteq \mathcal{R}$  and  $X \neq \emptyset$ , and assume that  $\mathcal{R} - X$  is a query result of  $Q$ . For an ontology  $O \in X$ , let  $i$  be the iteration in which combiSQORE considers to remove  $O$  and let  $\mathcal{R}_i \supseteq \mathcal{R}$  be the query result at the beginning of this iteration. For the ontology  $O$  to remain in the query result, it must be that  $\mathcal{R}_i - \{O\}$  is insufficient to satisfy all query conditions; otherwise combiSQORE would have removed  $O$  from  $\mathcal{R}_i$ . Therefore,  $\mathcal{R}_i - \{O\}$  is not a query result of  $Q$ . Since

$\mathcal{R} - X$  is a query result of  $\mathcal{Q}$ , and  $\mathcal{R} - X \subseteq \mathcal{R} - \{O\} \subseteq \mathcal{R}_i - \{O\}$ , from Theorem 1 one can obtain that  $\mathcal{R}_i - \{O\}$  is also a query result of  $\mathcal{Q}$ , which contradicts.  $\square$

### 4.3 Ranking Mechanism

Two criteria, namely *query coverage* and *conceptual closeness* are considered to compute semantic similarity score which is used for ranking query results generated by combiSQORE. Firstly, query coverage is defined to determine how well an ontology combination  $\mathcal{R}$  satisfies a given query  $\mathcal{Q}$ . Intuitively, it is measured by computing the ratio of the number of conditions satisfied by  $\mathcal{R}$  to the total number of conditions in  $\mathcal{Q}$ , hence its value ranges from 0 to 1. Since a candidate query result produced by combiSQORE guarantees to satisfy all conditions in  $\mathcal{Q}$ , its query coverage is 1.

**Definition 3 (Query Coverage Score: QS).** The query coverage between a semantic query  $\mathcal{Q}$  comprising  $m$  conditions  $q_1, \dots, q_m$  and a set of ontologies  $\mathcal{R}$  consisting of  $n$  ontologies  $O_1, \dots, O_n$  is measured by:

$$QS(\mathcal{Q}, \mathcal{R}) = \frac{\left| \left\{ q \in \mathcal{Q} : S(q, \mathcal{R}) \neq \emptyset \right\} \right|}{M} \tag{2}$$

$\square$

Next, the conceptual closeness between a query and a candidate query result comprising one or more ontologies will be formalized, by redefining certain semantic similarity measures developed by SQORE [2, 14], which simply capture the conceptual similarity between a query  $\mathcal{Q}$  and a single ontology  $O$ . Intuitively, based on  $SS_B(q, O)$  which defines the (*best*) *similarity score* between a condition  $q$  in  $\mathcal{Q}$  and the ontology  $O$ , SQORE defines the *query-ontology similarity score*:  $SS(\mathcal{Q}, O)$  to represent the conceptual closeness between  $\mathcal{Q}$  and ontology  $O$  by simply aggregating the similarity scores between all conditions in  $\mathcal{Q}$  and  $O$ .

Therefore, in order to measure the *conceptual closeness* between  $\mathcal{Q}$  and a combination of ontologies  $\mathcal{R}$ , the *query-combinative-ontology conceptual similarity score*:  $SS_C(\mathcal{Q}, \mathcal{R})$  is formalized here by aggregating the maximum similarity score between a query condition  $q$  in  $\mathcal{Q}$  and an ontology in  $\mathcal{R}$  as follows.

**Definition 4 (Query-Combinative-Ontology Conceptual Similarity Score: SS<sub>C</sub>).** The conceptual closeness between a semantic query  $\mathcal{Q}$  comprising  $m$  conditions  $q_1, \dots, q_m$  and a set of ontologies  $\mathcal{R}$  consisting of  $n$  ontologies  $O_1, \dots, O_n$  is measured by:

$$SS_C(\mathcal{Q}, \mathcal{R}) = \frac{\sum_{i=1}^m \max_{O \in \mathcal{R}} SS_B(q_i, O)}{m} \tag{3}$$

$\square$

Finally, QS and SS<sub>C</sub> are combined in order to measure the semantic similarity between  $\mathcal{Q}$  and a combination of ontologies  $\mathcal{R}$ , as follows.

**Definition 5 (Query-Combinative-Ontology Similarity Score: SS).** The semantic similarity between a semantic query  $Q$  comprising  $m$  conditions  $q_1, \dots, q_m$  and a set of ontologies  $\mathcal{R}$  consisting of  $n$  ontologies  $O_1, \dots, O_n$  is measured by:

$$SS(Q, \mathcal{R}) = QS(Q, \mathcal{R}) * SS_c(Q, \mathcal{R}) \quad (4)$$

□

Next section elaborates more details by means of an example.

## 5 An Example

Let  $\mathcal{ODB}$  be an ontology database comprising eight real-world OWL ontologies from different sources as shown in Table 1. Assume that a query  $Q$  comprising eight conditions is submitted, and the  $SS_B$  matrix measuring the similarity between each ontology and query condition is given in Table 2. From the table, one can see that O2, O3 and O6 have the highest similarity scores, and are ranked 1<sup>st</sup>, 2<sup>nd</sup> and 3<sup>rd</sup>, respectively. Moreover, Table 2 also depicts that each query condition is satisfied by more than one ontology in the collection. Thus,  $\mathcal{ODB}$  is sufficient to satisfy  $Q$ . However, there exists no single ontology that can satisfy all query conditions, which results in a need for combiSQORE algorithm to generate candidate query results and compute the ranking.

**Table 1.** An example of ontology database  $\mathcal{ODB}$

Ontology	URI
O1	<a href="http://swrc.ontoware.org/ontology">http://swrc.ontoware.org/ontology</a>
O2	<a href="http://ebiquity.umbc.edu/ontology/person.owl">http://ebiquity.umbc.edu/ontology/person.owl</a>
O3	<a href="http://annotation.semanticweb.org/iswc/iswc.owl">http://annotation.semanticweb.org/iswc/iswc.owl</a>
O4	<a href="http://ontoware.org/frs/download.php/18/semiport.owl">http://ontoware.org/frs/download.php/18/semiport.owl</a>
O5	<a href="http://morpheus.cs.umbc.edu/aks1/ontosem.owl">http://morpheus.cs.umbc.edu/aks1/ontosem.owl</a>
O6	<a href="http://www.csd.abdn.ac.uk/~cmckenzi/playpen/rdf/akt_ontology_LITE.owl">http://www.csd.abdn.ac.uk/~cmckenzi/playpen/rdf/akt_ontology_LITE.owl</a>
O7	<a href="http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl">http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl</a>
O8	<a href="http://protege.stanford.edu/plugins/owl/owl-library/ka.owl">http://protege.stanford.edu/plugins/owl/owl-library/ka.owl</a>

Let the input sequence  $l$  of combiSQORE be (O5,O2,O3,O7,O8,O6,O1,O4). The algorithm starts with an initial query result  $\mathcal{R}$  comprising all ontologies. Then, it iteratively checks whether removing an ontology from  $\mathcal{R}$  according to the order of the input sequence makes  $\mathcal{R}$  insufficient for  $Q$  or not. If  $\mathcal{R}$  remains sufficient, that ontology is removed from  $\mathcal{R}$ ; otherwise,  $\mathcal{R}$  is unchanged. For instance, removing Ontology O6 from the query result  $\mathcal{R} = \{O1, O4, O6\}$  will cause  $q_7$  and  $q_8$  unsatisfied. Thus, O6 cannot be removed from  $\mathcal{R}$ .

With respect to the given ontology collection  $\mathcal{ODB}$ , the submitted query  $Q$  and the input sequence  $l$ , combiSQORE generates the candidate query result  $\mathcal{R} = \{O1, O6\}$ , which is irreducible because removing either O1 or O6 will make some query

**Table 2.**  $SS_B$  matrix between a query condition in  $\mathcal{Q}$  and an ontology in  $\mathcal{ODB}$

Query Conditions in $\mathcal{Q}$	O1	O2	O3	O4	O5	O6	O7	O8
<b>q1:</b> <owl:Class rdf:ID="Student" />	1	1	1	1	1	1	1	1
<b>q2:</b> <owl:Class rdf:ID="PhDstudent"/>	1	1	1	1	0	1	0.6	1
<b>q3:</b> <owl:Class rdf:ID="Professor"/>	0.4	1	0.4	0.4	1	0.8	1	0
<b>q4:</b> <rdf:Property rdf:ID="supervise"/>	1	0	0	0	1	0	0	1
<b>q5:</b> <owl:Class rdf:about="PhDStudent" > <rdfs:subClassOf rdf:resource ="Student"/> </owl:Class>	1	1	1	1	0	0	0	0
<b>q6:</b> <rdf:Property rdf:about="firstname">	0	1	1	0.6	0	0.8	0.6	1
<b>q7:</b> <rdfs:domain rdf:resource ="Student"/>	0	1	1	0	0	0.8	0	1
<b>q8:</b> <rdfs:range rdf:resource ="xsd:String"/>	0	1	1	0	0	0.8	0	0
<b>SEMANTIC SIMILARITY SCORE:</b>	<b>0.55</b>	<b>0.87</b>	<b>0.8</b>	<b>0.5</b>	<b>0.37</b>	<b>0.65</b>	<b>0.4</b>	<b>0.62</b>

**Table 3.** Sample input sequences and their output combinations

Input Sequence	Candidate Query Result
<b>Seq1:</b> (O5,O2,O3,O7,O8,O6,O1,O4)	{O1,O6}
<b>Seq2:</b> (O2,O3,O6,O8,O1,O4,O7,O5)	{O4,O5,O6}
<b>Seq3:</b> (O5,O7,O4,O1,O8,O6,O2,O3)	{O3,O8}
<b>Seq4:</b> (O8,O7,O6,O4,O3,O5,O2,O1)	{O1,O2}

conditions unfulfilled. In addition, since different input sequences may yield different candidate query results, Table 3 gives other possible results.

In order to rank the top three single ontologies (i.e., O2, O3 and O6) together with the four candidate query results of Table 3, Table 4 illustrates their computed scores: query coverage score, conceptual closeness score and similarity score with the corresponding rankings shown in the followed brackets. With a focus on the final similarity scores, a combinative ontology, namely {O1,O2}, is ranked 1<sup>st</sup>, because it can satisfy all query conditions with highest conceptual closeness scores, while single ontologies fail to fulfill certain conditions and have lower conceptual closeness scores.

**Table 4.** Different rankings based on three ranking criteria

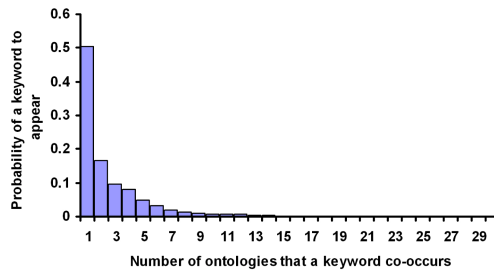
Ontologies	Query Coverage Score (QS)	Conceptual Closeness Score (SS <sub>C</sub> )	Similarity Score (SS=QS*SS <sub>C</sub> )
<b>O2</b>	0.875 (5)	0.87 (5)	0.761 (5)
<b>O3</b>	0.875 (5)	0.8 (6)	0.7 (6)
<b>O6</b>	0.75 (7)	0.65 (7)	0.488 (7)
<b>{O1,O6}</b>	1 (1)	0.9 (4)	0.9 (4)
<b>{O4,O5,O6}</b>	1 (1)	0.925 (2)	0.925 (2)
<b>{O3,O8}</b>	1 (1)	0.925 (2)	0.925 (2)
<b>{O1,O2}</b>	1 (1)	1 (1)	1 (1)

## 6 Experiments and Results

This section evaluates combiSQORE algorithm in terms of its performance and the validity of its rankings by means of experiments. An ontology database used in the experiment comprised 63 ontologies collected from three different domains: computer science, food and stock, while queries were automatically created by randomly selecting usable exact keywords from Wikipedia pages as shown in Table 5. The *total number of keywords* indicates the number of keywords extracted from the Wikipedia pages without considering stop words. The number of *usable exact keywords* represents the number of extracted keywords that can exactly match with concepts (classes) in the ontology database. The number of *usable related keywords* includes synonyms, hypernyms and hyponyms of the usable exact keywords which appear in the ontology collection.

**Table 5.** Statistics of Wikipedia pages used for generating keywords

Domain	Wikipedia page	Total Keywords	Usable Exact Keywords	Usable Related Keywords
Stock	<a href="http://en.wikipedia.org/wiki/Stock">http://en.wikipedia.org/wiki/Stock</a>	493	202	2027
Food	<a href="http://en.wikipedia.org/wiki/Food">http://en.wikipedia.org/wiki/Food</a>	672	259	732
Comp.Sc.	<a href="http://en.wikipedia.org/wiki/Computer_science">http://en.wikipedia.org/wiki/Computer_science</a>	283	107	733
<b>TOTAL</b>		1448	568	3492



**Fig. 3.** Richness of knowledge in the ontology collection

Fig. 3 presents the richness of knowledge in the ontology database based on how often exact and related keywords appear in different number of ontologies varying from one to twenty-nine. The graph shows that the probability that a keyword will appear in only one ontology is approximately 0.5. However, the probability of a keyword to co-occur in a higher number of ontologies decreases dramatically. Hence, given a random set of keywords, the chance that they all will co-occur in the same ontology is considerably low.

The experiment has been designed to test not only how well the algorithm performs in average, but also to investigate the impact of input sequences to the algorithm performance. Therefore, the experiment was performed as follows. Firstly,

a set of  $n$  keywords were randomly selected to formulate an input query, varying from  $n = 1$  to 10. Then, obtain the set of relevant ontologies from SQORE system, and apply combiSQORE algorithm with a designated input sequence. Certain analyses on the obtained results were then performed, as illustrated by Fig. 4 and Fig. 5. Note that each data point shown in the graphs represents the average value obtained from at least 50 trials or more.

## 6.1 Algorithm Performance

Fig. 4 illustrates the average number of relevant ontologies returned from SQORE, the average number of single ontologies that can satisfy all query conditions regardless to the conceptual similarity, and the average size of candidate query results. As expected, when the number of query conditions increases, the number of retrieved ontologies also increases whereas the number of single ontologies that can satisfy all query conditions decreases to zero. This result reflects the need for ontology combinations in order to entirely cover all conditions. In addition, the experimental result has shown that the average size of ontology combinations is approximately 3 for ten query conditions, which is acceptable for ontology integration.

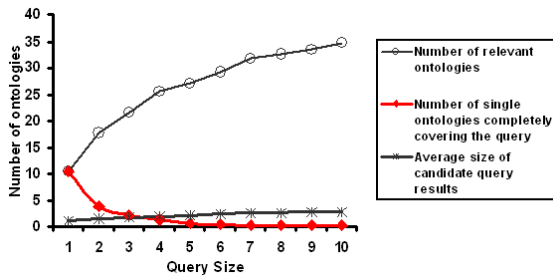


Fig. 4. Comparisons of resultant ontologies, individuals and combinations

As discussed earlier, with different input ontology sequences, combiSQORE may yield different combinative ontologies because a sequence determines the order of removing an ontology from an initial query result in order to finally obtain a candidate query result. Therefore, the algorithm performance is suspected to be improved if such a sequence is strategically generated. Intuitively, to maximize the conceptual closeness, the sequence should be sorted in ascending order of the similarity score. Since the similarity score tends to be proportional to the query coverage, the conducted experiment examined the three types of input sequences: (i) *random* ones, (ii) ones arranged in *ascending* order of the similarity score, and (iii) ones arranged in *descending* order of the similarity score. In addition, to illustrate the effectiveness of combiSQORE algorithm, results are also compared to three common approaches for selecting and combining ontologies regarding to similarity scores: (i) selecting only the highest-scored ontology, (ii) combining the two highest-scored ontologies and (iii) combining the three highest-scored ontologies.



Fig. 5 then presents the average query coverage scores and the average conceptual closeness scores of the computed results based on six different approaches as mention above. As expected, the sequences arranged in ascending order give the best candidate query results, whereas the random ones perform moderately well with the average similarity score of 0.8, which is considerably high. Furthermore, it clearly shows that combiSQORE with input sequences in ascending order of the similarity scores outperforms combining the highest-scored ontologies because the results by combiSQORE always completely satisfy the user query with higher conceptual closeness scores.

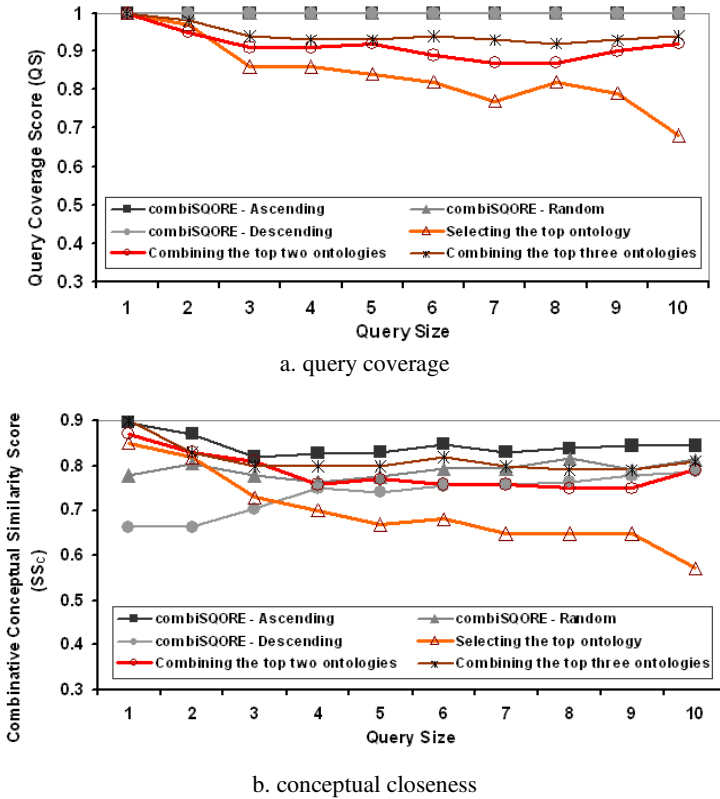


Fig. 5. Comparisons between the candidate query results of combiSQORE in different input sequences and those of other common approaches

### 6.2 Ranking Evaluation

In order to evaluate the practicality of the proposed ranking mechanism, a preliminary experiment was conducted. In the experiment, the ontology database and the formulated query of Section 5 was presented to four participants with a request to rank the top three single ontologies (i.e., O2, O3 and O6) together with the four candidate query results of Table 3 based on query coverage, conceptual closeness and similarity scores. Table 6a shows the average rankings proposed by the participants.

**Table 6.** Ranking evaluation results

a. Average ranks given by participants

Ontologies	Users – Query Coverage	Users - Conceptual Closeness	Users – Overall	combiSQORE- QS	combiSQORE- SS <sub>c</sub>	combiSQORE- SS
O2	5	6	6	5	5	5
O3	7	5	5	5	6	6
O6	5	6	7	7	7	7
{O1,O2}	1	2	2	1	1	1
{O1,O6}	3	4	4	1	4	4
{O3,O8}	3	2	3	1	2	2
{O4,O5,O6}	2	1	1	1	2	2

b. Pearson Correlation Coefficient for combiSQORE wrt. participant ranking

combiSQORE	PCC
Query coverage score (QS)	0.815
Conceptual closeness score (SS <sub>c</sub> )	0.917
Similarity score (SS)	0.918

*Pearson Correlation Coefficient (PCC)* [5] is employed to measure the similarity between the average participant rankings and the system rankings. If the calculated PCC value is closer to 1, it indicates a stronger linear relationship between the two rankings. Table 6b shows that the PCC values of the three rankings, based on query coverage, conceptual closeness and similarity scores, are significantly high, which imply that the rankings proposed by combiSQORE are very close to the participant rankings.

## 7 Conclusions and Future Work

This paper has proposed *combiSQORE*, a novel approach for computing and ranking ontology combinations, which can completely cover the specified user requirements. By integrating a number of ontologies, each partially satisfying the given requirements, the approach generates a minimal query result that can fulfill all requirements. The primary objective of the proposed approach is not only to enable automatic knowledge reuse for Semantic Web applications, but also to offer alternatives for ontology engineers and practitioners during their ontology search and development processes. In addition, it can also be applied to Web-service discovery applications in order to find sub-optimal sets of Web services that can meet all user requirements.

With a focus on a mechanism for ranking the generated ontology combinations, this paper has also developed simple methods to measure the *conceptual similarity*, and *query coverage* of an ontology combination with respect to a given query. These two criteria are then used to compute meaningful and practical rankings with the promising experimental results. In addition, modification (integration) cost is another metric that users are concerned. Future research direction includes an emphasis on discovering the inter-relationships among the ontologies in a combination, and

integrating such information to compute an accurate modification (integration) cost. Moreover, an enhancement by incorporating combiSQORE algorithm into the current system available on-line at <http://ict.shinawatra.ac.th:8080/sqore> is under way.

## References

1. Alani, H., Brewster, C.: Metrics for Ranking Ontologies. In: Proc. 4th Int. EON Workshop, 15th Int. WWW Conference, Edinburgh (2006)
2. Anutariya, C., Ungrangsi, R., Wuwongse, V.: SQORE – A Framework for Semantic Query based Ontology Retrieval. In: DASFAA 2007. Proc. 12th Int. Conf. Database Systems for Advanced Applications, Bangkok. LNCS, vol. 4443, pp. 924–929 (2007)
3. Berners-Lee, T., Handler, J., Lassila, O.: The Semantic Web. Scientific American, Singapore (May 2001)
4. Cantador, I., Fernández, M., Castells, P.: Improving Ontology Recommendation and Reuse in WebCORE by Collaborative Assessments. In: Proc. 4th Int. Workshop Evaluation of Ontologies for the Web, 15th Int. WWW Conf., Edinburgh (2006)
5. Conover, W.J.: Practical Non-Parametric Statistics, 2nd edn. John Wiley and Sons, Chichester (1980)
6. Ding, L., Finin, T., Joshi, A., Pan, R., Scott Cost, R., Peng, Y., Reddivari, P., Doshi, V., Sachs, J.: Swoogle: a search and metadata engine for the Semantic Web. In: Proc. 13th ACM Int. Conf. Information and Knowledge Management, DC (November 2004)
7. Dzbor, M., Domingue, J., Motta, E.: towards a Semantic Web browser. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 690–705. Springer, Heidelberg (2003)
8. Fernández, M., Cantador, I., Castells, P., CORE.: A Tool for Collaborative Ontology Reuse and Evaluation. In: Proc. 4th Int. Workshop Evaluation of Ontologies for the Web, 15th Int. WWW Conf., Edinburgh (2006)
9. Lopez, V., Motta, E., Uren, V.: PowerAqua: Fishing the Semantic Web. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 394–410. Springer, Heidelberg (2006)
10. Miller, A., WordNet, A.: lexical database for English. Communications of the ACM 38(11) (1995)
11. Motta, E., Sabou, M.: Next Generation Semantic Web Applications. In: Proc. 1st Asian Semantic Web Conf., China (2006)
12. Patel, C., Supekar, K., Lee, Y., Park, E.K.: OntoKhoj: a Semantic Web portal for ontology searching, ranking and classification. In: Proc. 5th ACM Int. Workshop Web Information and Data Management, Louisiana (November 2003)
13. Sabou, M., Lopez, V., Motta, E., Uren, V.: Ontology Selection: Ontology Evaluation on the Real Semantic Web. In: Proc. of the 4th Int. EON Workshop, Evaluation of Ontologies for the Web, 15th Int. WWW Conf., Edinburgh (2006)
14. Ungrangsi, R., Anutariya, C., Wuwongse, V.: SQORE-based Ontology Retrieval System. In: DEXA 2007. Proc. 18th Int. Conf. Database and Expert Systems Applications, Regensburg, Germany, vol. 4653, pp. 720–729 (2007)
15. Wuwongse, V., Anutariya, C., Akama, K., Nantajeewarawat, E.: XML Declarative Description (XDD): A Language for the Semantic Web. IEEE Intelligent Systems 16(3), 54–65 (2001)
16. Zhang, Y., Vasconcelos, W., Sleeman, D.: OntoSearch: An ontology search engine. In: Proc. 24th SGAI Int. Conf. Innovative Techniques and Applications of AI, UK (2004)

# PORE: Positive-Only Relation Extraction from Wikipedia Text\*

Gang Wang, Yong Yu, and Haiping Zhu

Apex Data & Knowledge Management Lab,  
Department of Computer Science and Engineering,  
Shanghai Jiao Tong University, Shanghai, 200240, China  
{gavinwang, yyu, zhu}@apex.sjtu.edu.cn

**Abstract.** Extracting semantic relations is of great importance for the creation of the Semantic Web content. It is of great benefit to semi-automatically extract relations from the free text of Wikipedia using the structured content readily available in it. Pattern matching methods that employ information redundancy cannot work well since there is not much redundancy information in Wikipedia, compared to the Web. Multi-class classification methods are not reasonable since no classification of relation types is available in Wikipedia. In this paper, we propose *PORE (Positive-Only Relation Extraction)*, for relation extraction from Wikipedia text. The core algorithm *B-POL* extends a state-of-the-art *positive-only learning* algorithm using bootstrapping, strong negative identification, and transductive inference to work with fewer positive training examples. We conducted experiments on several relations with different amount of training data. The experimental results show that *B-POL* can work effectively given only a small amount of positive training examples and it significantly outperforms the original positive learning approaches and a multi-class SVM. Furthermore, although *PORE* is applied in the context of Wikipedia, the core algorithm *B-POL* is a general approach for Ontology Population and can be adapted to other domains.

**Keywords:** Relation Extraction, Ontology Population, Positive-Only Learning.

## 1 Introduction

The Semantic Web builds on not only ontologies but also the contents conforming to the ontologies. According to a recent study [1], although Semantic Web data is growing steadily on the Web, the space of instances is sparsely populated (most classes (>97%) have no instances and the majority of properties (>70%) have never been used to assert data). Consequently, Ontology Population and Annotation are of great importance for the realization of the Semantic Web.

It is of great benefit to extract semantic content from Wikipedia, the largest free online encyclopedia (<http://www.wikipedia.org>). Völkel et al. [12] provided an extension to be integrated into Wikipedia to allow the creation of an open semantic

---

\* This work is funded by IBM China Research Lab.

knowledge base. Auer and Lehmann [13] recently argued that means for creating semantically enriched structured content are already available and used by Wikipedia authors. They presented a pattern-matching approach to extract the structured content and proposed strategies that require only minor modifications of the wiki systems for improving the quality of the creation of structured content. In this paper, we plan to go a step further. We propose an approach that exploits the structured content readily available in Wikipedia to semi-automatically extract semantic relations between Wikipedia entities from the free text. The relations are already defined in the structured tables, along with a set of relation instances. This is an Ontology Population task, where only a relatively small amount of relation instances are available for learning while no negative examples are provided.

A great amount of research work has been conducted to extract relations using a small amount of seed instances. DIPRE [3] paradigm based work Snowball [4], Espresso [5], and [6], etc. employed bootstrapping based pattern matching approaches. The approaches exploited *information redundancy* of the Web — instances to be extracted will tend to appear in uniform contexts repeatedly. However, compared to the Web, information redundancy cannot be guaranteed in Wikipedia.

Work conducted in [19] [20] performed multi-class relation classification [21] based on a hierarchical classification of relation types. However, relations to be extracted from Wikipedia are more fine-grained and diverse so that no such relation type classification is available in Wikipedia. Consequently, it is not reasonable to employ multi-class classification.

In this paper, we propose *PORE (Positive-Only Relation Extraction)*, a new approach to extracting relation instances from Wikipedia text. The core algorithm *B-POL* builds on top of a state-of-the-art *positive-only learning (POL)* approach [14] [15] that initially identifies strong negative examples from unlabeled data and then iteratively classifies more negative data until convergence. *B-POL* makes several extensions to *POL* to work with fewer positive examples without sacrificing too much precision. Specifically, a conservative strategy is made to generate strong initial negative examples, resulting in high recall at the first step. The newly generated positive data identified by *POL* are added for training and the underlying *POL* approach is invoked again to generate more positive data. The method iterates until no positive data can be generated anymore. It exploits unlabeled data for learning and is transformed to a transductive [22] learning method that is believed to work better with sparse training data. Furthermore, it is built on top of a state-of-the-art statistical learning algorithm SVM. These settings enable the effective learning with fewer positive examples. To the best of our knowledge, no work has been done on using positive-only learning (classification) algorithms for relation extraction.

We conducted experiments on several relations, each of which has different amount of training instances. We evaluated the results against a manually constructed *gold standard* and it showed that the core algorithm *B-POL* outperforms a simple transductive version of *POL* and a transductive *POL* with a conservative strategy. *B-POL* also significantly outperforms a multi-class SVM approach. Last but not least, although *PORE* is applied in the context of Wikipedia, the core algorithm *B-POL* is a general approach for Ontology Population and can be adapted to other domains.

The rest of the paper is organized as follows. Section 2 compares our work with other ongoing relevant research work. In Section 3, we elaborate on the core algorithm, *B-POL*. We give in Section 4 the description of Wikipedia and the

features, as well as the filtering process. Section 5 describes the experiments and evaluation. Finally, we conclude this paper and present future work in Section 6.

## 2 Related Work

DIPRE [3] based methods [4] [5] [6] exploited information redundancy on the Web and the pattern/relation duality by using pattern matching combined with bootstrapping. Exploring the Web for redundancy information is reasonable. However, such systems need to estimate the confidence of patterns and instances, which is a rather difficult task. Other methods exploiting information redundancy can be found in [7] [8]. These systems generally face the problem that many parameters need to be specified for each relation.

LEILA [18] automatically generated negative examples using information about the cardinality of relations. Work conducted in [19] [20] employed semi-supervised learning algorithms and achieved good performance using only a small amount of labeled examples. They performed multi-class classification in which all the relation types are already defined [21]. Mori et al. [9] described an approach for extracting relations in social networks. Work by Wang et al. [21] was conducted on the ACE corpus using various features. Schutz and Buitelaar [27] described *RelExt* for extracting relations in the football domain. Tang et al. [10] proposed Tree-CRF for semantic annotation on semi-structured data. Ramakrishnan et al. [2] described a schema-driven approach to relation extraction from biomedical text.

The *Semantic Wikipedia* project described in [12] provided an extension to be integrated into Wikipedia to allow the creation of an open semantic knowledge base. A recent study [13] directly extracted structured tables of relations from Wikipedia using pattern matching. YAGO [31] built an ontology by extracting relations from Wikipedia categories. It mainly employed heuristic rules and WordNet during the extraction and presented results of high quality. However, the approach is somehow limited to the extraction of certain types of relations due to the fact that it did not explore the free text which is the main source of relations. Ruiz-Casado et al. [25] described an extraction pattern-based method for extracting *is-a* and *part-whole* relations from Wikipedia text to enrich WordNet. Wang et al. [24] exploited various features in Wikipedia to enhance the extraction of relations from Wikipedia text. However, the method requires manual tuning of the similarity thresholds for each pattern, which is tedious and impractical for large scale applications. In this paper, we employ feature-based SVM classification [26], which is believed to be more robust, to extract mainly non-taxonomic relations from Wikipedia.

## 3 B-POL

In this section, we present the core algorithm, *B-POL*. It builds on top of two similar state-of-the-art *positive-only learning* approaches PEBL [14] and Roc-SVM [15] that initially identify strong negative examples from unlabeled data and then iteratively classify more negative data until no such data can be found.

Prior to the illustration of the learning framework, we first formulate the relation extraction problem as a positive-only binary classification task.

Given a collection  $C$  of co-occurrence contexts of entity pairs, a given relation type  $R$  as well as a set of entity pairs as training data (the corresponding co-occurrence contexts in  $C$  are denoted as  $P$ , the positive set), the task is to assign the relation type  $R$  to occurrences (in the unlabeled set  $U = C - P$ ) that indicate the relation. (Each co-occurrence context is represented as a vector of relevant features which are explained in Sec. 4)

The original positive-only classification method proposed in [14] and [15] is an inductive learning algorithm [22] because they output a final classifier that can make predictions on unseen data. Since it is believed that transductive inference is generally suited to the problems with a small amount of training data [22], we transformed the original method into a transductive one. We call the adaptation of the positive-only learning method as *T-POL* (*Transductive Positive-Only Learning*), which is shown in Fig. 1.

Algorithm: *T-POL* ( $P, U$ )

Input: positive set  $P$ , unlabeled set  $U$

Output: a set  $P_u$  of examples finally classified as positive

1. Use a weak classifier  $\Psi$  to classify using  $P$  and  $U$ . The data in  $U$  classified as positive is  $P_0$ , the strong negatives  $N_0 \leftarrow U - P_0$
2. Set  $N \leftarrow \Phi, i \leftarrow 0$
3. Do loop
  - 3.1  $N \leftarrow N \cup N_i$
  - 3.2 Use  $\nu$ -SVM to classify  $P_i$  with positive set  $P$  and negative set  $N$ 
    - 3.2.1  $N_{i+1} \leftarrow$  examples from  $P_i$  classified as negative
    - 3.2.2  $P_{i+1} \leftarrow$  examples from  $P_i$  classified as positive
  - 3.3  $i \leftarrow i + 1$
  - 3.4 Repeat until  $N_i = \Phi$
4.  $P_u \leftarrow P_i$ , return  $P_u$

**Fig. 1.** Transductive Positive-Only Learning method (*T-POL*)

In step 1 of *T-POL* algorithm, a weak classifier  $\Psi$  is employed to draw an initial approximation of “strong negatives”, which are the negative data located far from the boundary of the positive class in the universal feature space. Rocchio [16] and OSVM (One-Class SVM) [28] were employed as the weak classifier  $\Psi$  in [15] and [14], respectively. In step 3.2 of *T-POL*,  $\nu$ -SVM [23] is employed to maximize the margin using the positive data and the current version of negatives.  $\nu$ -SVM is a version of SVM with a soft margin and is necessary for *T-POL* to cope with noises in the training data [14]. The rate of noise in training data is controlled by the parameter  $\nu$ , which can generally be set to a low value (e.g. 0.01).  $\nu$ -SVM maximizes the margin at each iteration and thus progressively improves the approximation of negative data. Consequently, the class boundary eventually converges to the true boundary of the positive class in feature space [15].

However, in step 1, the weak classifier  $\Psi$  in [15] and [14] tends to generate too many false negatives from  $U$ , which results in low recall in later iterations. As pointed out in [14], classifier  $\Psi$  should generate pure negatives  $N_0$  excluding false negatives

by sacrificing precision in  $P_0$ . The precision of step 1 does not affect the accuracy of the final boundary as far as it approximates a certain amount of negative data because the final boundary will be determined by step 2-4. Motivated by this, we only select the “strongest” negatives identified by  $\Psi$ . The modified classifier  $\Psi$  based on Rocchio is named *Roc-SN*, which is shown in Fig. 2.

Algorithm: *Roc-SN* ( $P, U, c$ )

Input: positive set  $P$ , unlabeled set  $U$ , the percentage  $c$  of the “strongest” negatives out of all negatives identified by Rocchio.

Output: a set  $N_0$  of “strongest” negatives

--- Each instance is represented as  $i$ , with corresponding vector  $\vec{i}$

1. Construct two prototype vectors:

$$1.1 \vec{c}^+ \leftarrow \alpha \frac{1}{|P|} \sum_{i \in P} \frac{\vec{i}}{\|\vec{i}\|} - \beta \frac{1}{|U|} \sum_{i \in U} \frac{\vec{i}}{\|\vec{i}\|}$$

$$1.2 \vec{c}^- \leftarrow \alpha \frac{1}{|U|} \sum_{i \in U} \frac{\vec{i}}{\|\vec{i}\|} - \beta \frac{1}{|P|} \sum_{i \in P} \frac{\vec{i}}{\|\vec{i}\|}$$

2. Set  $N_0 \leftarrow \emptyset$

3. For each instance  $i$  in  $U$  do loop

$$3.1 s_i \leftarrow sim(\vec{c}^-, \vec{i}) - sim(\vec{c}^+, \vec{i})$$

3.2 If  $s_i > 0$  then  $N_0 \leftarrow N_0 \cup i$

4.  $N_0 \leftarrow$  top  $\lfloor c \times |N_0| \rfloor$  instances with largest  $s_i$ , return  $N_0$

**Fig. 2.** Modified version of Rocchio for identifying the “strongest” negatives (*Roc-SN*)

In Rocchio classification, the classifier is built by constructing positive and negative prototype vectors (the unlabeled data are treated as negatives). If the similarity (*sim*, cosine similarity) between the test instance  $i$  and the negative prototype vector is larger than that between  $i$  and the positive one,  $i$  is added to negative set. The parameters  $\alpha$  and  $\beta$  adjust the relative impact of positive and negative instances and are set to 16 and 4, respectively in text classification tasks [15]. In *Roc-SN*,  $s_i$  is used to measure the “strength” of the negative instance  $i$ . Parameter  $c$  is used to determine the percentage of instances that are selected as the “strongest” negatives out of the entire set of negatives identified by Rocchio. In this way, the top  $\lfloor c \times |N_0| \rfloor$  negatives with the largest “strength” are finally retained in *Roc-SN*. The smaller  $c$  is, the purer the generated “strongest” negatives are. This means a smaller  $c$  could generally bring higher recall while a larger  $c$  would give higher precision as it identifies more negatives. It is obvious that it degenerates to the original Rocchio classifier when  $c = 1$ .

However, when the positive examples are too few, *T-POL* would end up fitting tightly around the few positive training examples, resulting in low recall [14]. Having observed that precision is not directly influenced when positive examples are under-sampled, we extend *T-POL* by adding the positive data ( $P_u$ ) newly generated by *T-POL* to the set of training examples and invoking *T-POL* again to generate more



positive data. The algorithm iterates until no positive data can be returned from  $T$ - $POL$ . This bootstrapping version of  $T$ - $POL$  gives the core algorithm,  $B$ - $POL$ , which is illustrated in Fig. 3.

Algorithm:  $B$ - $POL$  ( $P$ ,  $U$ )  
 Input: positive set  $P$ , unlabeled set  $U$   
 Output: a set  $P_u$  of examples classified as positive

1. Set  $P_u \leftarrow \Phi$ ,  $i \leftarrow 0$
2. Do loop
  - 2.1  $i \leftarrow i + 1$
  - 2.2 Set  $P_u^{(i)} \leftarrow$  positive examples returned from  $T$ - $POL(P \cup P_w \cup U)$
  - 2.3  $P_u \leftarrow P_u \cup P_u^{(i)}$ ,  $U \leftarrow U - P_u^{(i)}$
  - 2.4 Repeat until  $P_u^{(i)} = \Phi$
3. Return  $P_u$

**Fig. 3.** Bootstrapping POL ( $B$ - $POL$ )

This kind of bootstrapping is commonly referred to as self-training, which has been reported to perform well especially in natural language processing tasks [30]. In  $B$ - $POL$ , the classifier uses its own predictions to re-train itself. It is such reinforcement that contributes to the high recall when fewer positive examples are provided.

## 4 PORE

$PORE$  works as follows: 1) extracting entity features from semi-structured data of Wikipedia; 2) extracting entity-pair co-occurrence context from Wikipedia text; 3) for each relation, filtering out irrelevant instances using the positive training data extracted from the structured content of Wikipedia; 4) conducting relation classification on the filtered set of instances using  $B$ - $POL$ . The positive instances output by  $B$ - $POL$  are manually examined and the true positives are finally stored as RDF triples.

### 4.1 Wikipedia

Wikipedia is a hypertext document collection with a rich link structure. Generally in each page of Wikipedia, the first sentence serves as the definition of an entity (entry). The *bold italic* phrase in the definition is a self-reference to the current entry. Each article in Wikipedia is assigned at least one category. In some articles, an infobox containing a picture gives a general description of an entity. In each infobox within an article, there are a set of properties defined to describe the entity. Each property generally demonstrates a relation between two entities. The entity described by the current article can be viewed as the subject of the relations. The objects are connected by relation predicates and are mainly internal links that point to other entities in Wikipedia or just literal text or, in some cases, external links pointing to web pages outside Wikipedia. Fig. 4 gives a snapshot of the article “*Annie Hall*”, which demonstrates the (semi-) structured contents associated with a Wikipedia entry.

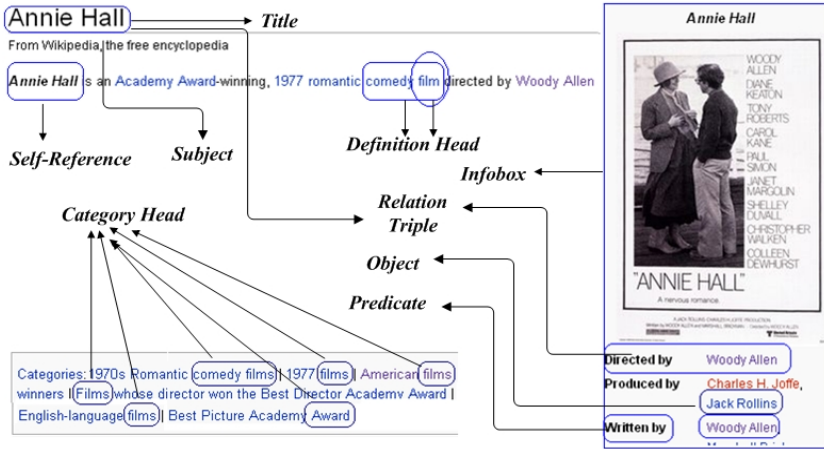


Fig. 4. (Semi-)Structured Contents in Wikipedia (from entry *Annie Hall*)

## 4.2 Feature Engineering

Feature based relation extraction using SVM is a popular approach and gives the current best reported results on ACE corpus in [26]. We separate *entity features* which describe Wikipedia entities from *context features* which describe co-occurrence contexts of pairs of Wikipedia entities.

**Entity Feature Extraction.** As shown in Fig. 4, a Wikipedia entity (entry) is described by definition, categories as well as predicates in the infobox. Wang et al. [24] argued that the Wikipedia entity features are more powerful than traditional Named Entity Recognition (NER) since they give more fine-grained descriptions for an entity.

For *definition features*, we heuristically extract the head word of the first base noun phrase (BNP) following a *be*-verb (i.e. *is*, *was*, *are*, *were*, etc.). For example, in the sentence “Annie Hall is an Academy Award-winning, 1997 romantic comedy film directed by Woody Allen.”, the word “film” and the augmented word “comedy\_film” are extracted as entity features for the Wikipedia entity “Annie Hall”.

For *category features*, since the name of each category is a noun phrase, heuristically, the head word of the first base noun phrase in the category phrase is extracted. Take the entry “Annie Hall” for example, “film” and the augmented version “comedy\_film” are extracted from category “Romantic comedy films”.

For *infobox features*, names of the predicates, with each white space character replaced by an underscore (e.g. “produced\_by”, “written\_by”, etc.) are kept.

**Context Feature Extraction.** Context features are derived from the co-occurrence of entity pairs in a sentence. As in the sentence “In the film “Heavenly Creatures”, directed by Peter Jackson, Juliet Hulme had TB, and her fear of being sent ...”, there are three hyperlinked entities (as indicated by the underscore). For each pair of entities, e.g. (Heavenly Creatures , Peter Jackson), tokens to the left of Heavenly Creatures, those to the right of Peter Jackson, and those in between the two entities

are extracted and encoded as the context features. For the details of how to encode the context features using the tokens, one may refer to the technical report [29] which provides a formal definition of the features.

### 4.3 Data Filtering

The number of the entity pairs can be very large, and thus it is inefficient if they are directly classified. Furthermore, because of the highly skewed data distribution, the recall of the SVMs would decrease. In Snowball [4], named entity types of a relation are used to filter data. In the same way, we use the entity features for filtering.

We first define a feature selection method. We denote the complete set of data as  $C$  and the positive set in  $C$  as  $P$ . To define a score of a feature  $f$ , we further denote the set of data from  $P$  containing  $f$  as  $P_f$  and the set of data from  $C$  containing  $f$  as  $C_f$ . The feature scoring function is shown in equation (1).

$$\text{score}(f) = |P_f| \times \log\left(\frac{|C|}{|C_f|}\right). \quad (1)$$

It can be observed that features of an entity are usually diverse, expressing different aspects of the entity. Nevertheless, it is reasonable to assume that entities in a given relation at a given argument position (subject or object) share a certain degree of commonality [24]. We use equation (1) to score features of entities at each argument position (subject or object) and select top  $k$  features with the highest scores. The value of  $k$  is set according to the following heuristics:

- $k = \lfloor 10\% * \text{\#entity features} \rfloor$ , (if  $k = 0$ , then  $k = 1$ ; if  $k > 15$ , then  $k = 15$ ).

The selected features are called *Salient Entity Features*. For convenience, the salient features of entities at subject (object) position are called *Salient Subject (Object) Features*. The set of entity pairs from which features of the left-hand-side entity intersect with the *Salient Subject Features* and meanwhile features of the right-hand-side entity intersect with the *Salient Object Features* are kept. We denote the set of entity pairs finally kept as  $C'$ , and then the unlabeled set  $U = C' - P$ . Finally, we apply *B-POL* to classify  $U$  using  $P$  (see Sec. 3).

As in the previous example, “*In the film "Heavenly Creatures", directed by Peter Jackson, Juliet Hulme had TB, and her fear of being sent ...*”, although there are several pairs of entities, pairs such as  $\langle \textit{Peter Jackson}, \textit{Juliet Hulme} \rangle$ ,  $\langle \textit{Heavenly Creatures}, \textit{Juliet Hulme} \rangle$ , etc. will be filtered out when we are extracting *film-director* relation. This is because the *Salient Subject Features* and the *Salient Object Features* constructed using the positive training data are  $\langle \textit{film}, \textit{drama\_film}, \textit{movie}, \dots \rangle$  and  $\langle \textit{director}, \textit{film\_director}, \dots \rangle$ , respectively, which do not have intersection with those of the filtered-out pairs.

## 5 Evaluation

For the experiments, we used the data from the *Wikipedia XML* corpus [17]. Our work is concerned with extracting relations between Wikipedia entities and thereby only the

internal links are considered. In the current experimentation, the definitions of relations as well as the corresponding training instances come from the infoboxes of Wikipedia. Nevertheless, one can still define other relations and provide corresponding training instances to make *PORE* work. Here we focus on extracting relations from free text, so the highly structured pages with titles like “List of” or “Lists of” and the disambiguation pages are not considered. We finally obtained 644,508 pages.

In the experiments, all NLP tasks are performed using the OpenNLP toolkit (<http://opennlp.sourceforge.net/>). Stemming is performed by Snowball stemmer shipped with Lucene (<http://lucene.apache.org>). In the context feature extraction, we only keep links whose anchor text represents a proper noun.

We focus on evaluating the performance of the core part, *B-POL*. Two methods are selected as baselines. One is the simple transductive version of the original positive-only learning method using Rocchio<sup>1</sup>, namely *T-POL*. The other is *T-POL* with the modified Rocchio, namely *Roc-SN*.

The experiments are conducted over a subset of 10,000 pages randomly selected from the *Wikipedia XML* corpus. There are about 130,000 pairs of entities in the subset. In order to evaluate the performance using precision, recall and F1, we need to construct a *gold standard* set from the selected subset of pages for each relation. However, it is impractical to manually label the 130,000 pairs. Neither can we randomly sample a smaller subset since the distribution of the target relations is highly skewed. We also use the Wikipedia entity features to pre-filter the irrelevant pairs like what we do in Sec. 4.3. However, we do not directly take the original method in Sec. 4.3 since it is part of our approach to be evaluated. In contrast, we use the entire set of entity features. The construction of the *gold standard* is illustrated as follows.

1. Use Lucene to build an inverted index of the entity pairs using the entity features.
2. For each relation, we obtain a set of instances from the corresponding Wikipedia infoboxes. Then we find out the instance occurrences in the inverted index. Taking the occurrences as the positive set  $P$  and the entire entity pairs as the unlabeled set  $U$ , we use equation (1) to calculate scores of the subject (object) features.
3. Use Lucene to build a BooleanQuery *subject\_query* (*object\_query*) by selecting all the subject (object) entity features as query terms and taking the corresponding feature scores (calculated using equation (1)) as query weights. A final BooleanQuery in the form of “*subject\_query* AND *object\_query*” is submitted to Lucene.
4. From the ranked list of entity pairs, we retain the top 1000 pairs only.
5. The 1000 pairs are manually examined by three human subjects. The correct entity pairs that achieve agreements, along with their co-occurrence context, are added to the *gold standard*.

For the algorithms in the experiments, we use LibSVM [11] which supports  $\nu$ -SVM (Sec. 3) to implement the *POL* methods. In terms of the specific SVM model, we choose RBF (Radial Basis Function) kernel. According to [11], it can handle non-linear relations between class label and attributes, and it subsumes linear

---

<sup>1</sup> As mentioned in Sec. 3, transductive inference is believed to perform better than the inductive counterpart when handling small amount of training data. As a result, we do not take the original inductive one as a baseline.

kernel. In the experiments,  $\nu$  of  $\nu$ -SVM is set to a theoretically motivated fixed parameter, 0.01 (Sec. 3). We use the default parameters provided in LibSVM for other parameter settings.

Prior to the demonstration of the results, we introduce the following denotations.

- P: the set of training data (entity pair occurrences).
- U: the set of the unlabeled instances after filtering (test data).
- GS: the set of instances in the *gold standard*.

In the infoboxes of the *Wikipedia XML* collection, there are currently 9,197 relations and 953,550 relation instances. Considering both the time and space limitations, currently we only select several relations for demonstration. The selection criteria are as follows: 1) there are a sufficient number of ground truth instances in the remaining data after filtering, making it possible to show the performance with different amount of training data; 2) the relations are somehow typical, so they can reflect different aspects of problems which need to be addressed; 3) the relations are from different domains.

Table 1 gives the information about the four relations that are tested in our experiments. The ‘‘Source’’ means the infobox from which the relation and its instances are extracted.  $\#(GS \cap U)$  indicates the performance of the data filtering using the *Salient Entity Features*. It can be seen that recall (calculated by  $\#(GS \cap U) / \#GS$ ) is relatively high. Precision at this stage does not matter much since the unlabeled data will be tested by *B-POL*.

**Table 1.** Information about the four relations

Relation	Source	#GS	#U	$\#(GS \cap U)$
album-artist	album_infobox#artist	274	392	260
film-director	infobox_movie#director	121	286	115
university-city	infobox_university#city	74	208	71
band-member	infobox_band#current_members	117	477	103

Fig. 5 demonstrates the performance of *T-POL* and *B-POL* with different number of training data and different settings for parameter  $c$  (Sec. 3). F1-scores are plotted with different values of parameter  $c$  (it is used in each invocation of *Roc-SN*). It is obvious that the results of *T-POL* at  $c = 1$  are also the results of *T-POL'*. Each value of F1-score is averaged over 20 trials. Table 2 gives the results of *B-POL* and *T-POL*. At each invocation of *Roc-SN*,  $c$  is set to a random value that ranges from 0.1 to 1.0. The results in Table 2 are averaged over 50 trials to achieve high reliability.

***B-POL* vs *T-POL* and *T-POL'*.** From Fig. 5, especially the results of *album-artist*, *film-director* and *band-member* relations, *B-POL* consistently outperforms *T-POL* at nearly all settings of parameter  $c$ . The averaged F1-scores in Table 2 also demonstrate the significant improvement of *B-POL* over *T-POL* and *T-POL'*. *B-POL* significantly increases recall by sacrificing not too much precision.

For the *university-city* relation, the gap is smaller. F1-scores of *T-POL'* and *T-POL* even surpass that of *B-POL* when  $\#P=40$ . This is largely due to the reason that when the number of the positives in the unlabeled data is small, the bootstrapping strategy of *B-POL* would not benefit from the improvements in recall but just lowering

precision. As described in Table 1, the size of the *gold standard* of *university-city* is small. When  $\#P=40$ , the number of positives in the unlabeled set is smaller than  $\#P$ . In this case, the original *POL* methods work better. We also found that the co-occurrence contexts for *university-city* relation are quite general (e.g. “<university>, <city>”, “<university> in <city>”, “<university>, at <city>”). The bootstrapping strategy of *B-POL* brings more errors during further iterations, which results in much decrease in precision. However, the increase of recall brought by *B-POL* produces larger F1-scores when the amount of training data is much less (i.e.  $\#P=10$ ).

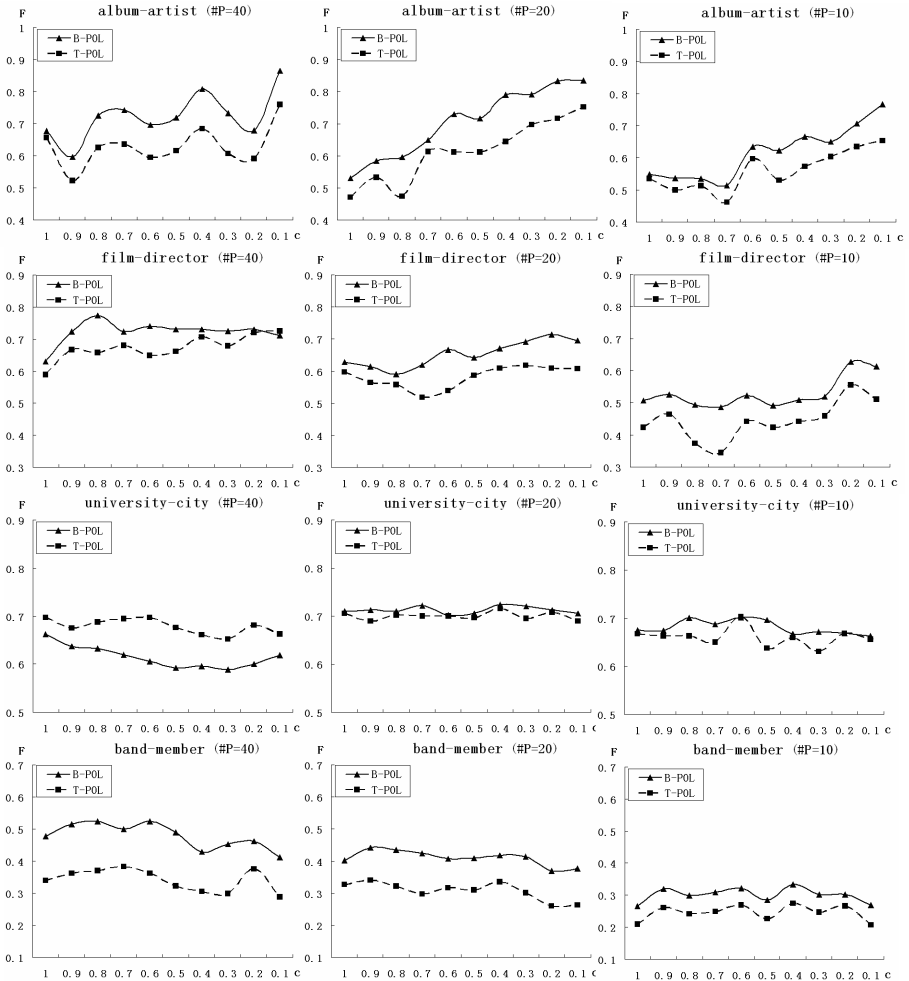


Fig. 5. F1-scores of *T-POL* and *B-POL* on the four relations with different settings

From Table 2, it can be observed that *B-POL* achieves significantly higher F1-scores than *T-POL* and *T-POL'* when the amount of the training data is less. This indicates that *B-POL* is more effective when dealing with fewer positive training data.

**POL vs M-SVM.** We also assess the performance of multi-class classification using LibSVM (*M-SVM*). We use the same training data and unlabeled data in *B-POL* for *M-SVM*. Our original setting for *M-SVM* is as follows. We treat each of the four relations as a class and add another the “others” class to indicate other relations or unrelated entity pairs. The examples for “others” class are sampled in the entire collection of entity pairs excluding the portion in the *gold standard* of the four relations. The sample size of the “others” class is equal to that of the four relations. However, this setting produces rather bad performance. Even when the same amount of training data is used for each class, the *album-artist* relation and “others” are always overwhelming and *M-SVM* just distributes the labels of the unlabeled data to the two classes. The other three relations obtain nearly zero F1-scores. Consequently, we actually conduct two-class classification by each time selecting only one of the four relations. The results are better than that of the original one. However, it can be seen in Table 2, the performance of *M-SVM* is still worse than *B-POL* and *T-POL*. It is even worse than *T-POL*’ when the training data is not much under-sampled in most cases.

**Table 2.** The extraction performance (Prec./Rec./F1) of *B-POL* and the other 3 baselines

#P	method	album-artist	film-director	university-city	band-member
		P/R/F1	P/R/F1	P/R/F1	P/R/F1
40	<i>T-POL</i> ’	96.7/36.5/47.8	82.8/50.6/60.6	65.4/74.4/ <b>68.6</b>	70.2/25.0/35.7
	<i>T-POL</i>	89.6/49.8/59.2	82.2/58.2/66.4	62.0/76.8/68.1	67.6/25.0/34.8
	<i>B-POL</i>	86.6/77.5/ <b>79.9</b>	69.4/81.2/ <b>73.2</b>	47.2/84.8/58.5	46.8/57.6/ <b>47.1</b>
	<i>M-SVM</i>	93.6/40.4/54.5	71.2/32.8/41.4	17.4/36.9/19.5	35.4/29.7/ 27.5
30	<i>T-POL</i> ’	97.4/45.8/58.8	85.5/51.1/62.2	75.1/67.7/70.5	74.3/24.5/35.9
	<i>T-POL</i>	93.2/56.7/68.2	83.7/51.0/61.8	70.7/72.6/ <b>70.6</b>	67.6/22.0/32.4
	<i>B-POL</i>	90.6/70.2/ <b>76.5</b>	73.4/69.6/ <b>68.6</b>	62.7/79.0/68.5	58.5/46.6/ <b>49.3</b>
	<i>M-SVM</i>	93.4/46.2/58.0	72.1/37.9/44.8	20.9/33.7/21.9	36.1/32.5/30.0
20	<i>T-POL</i> ’	97.1/34.6/48.0	84.6/37.7/49.9	80.3/63.6/70.5	77.7/21.7/33.5
	<i>T-POL</i>	93.5/52.8/63.7	81.3/47.0/56.5	79.8/64.0/70.2	72.3/21.0/31.5
	<i>B-POL</i>	90.0/69.2/ <b>76.4</b>	74.7/64.1/ <b>66.6</b>	75.3/70.1/ <b>71.6</b>	67.9/32.3/ <b>41.9</b>
	<i>M-SVM</i>	93.8/42.4/55.9	73.1/40.5/46.9	27.0/31.6/26.0	39.4/32.9/29.8
10	<i>T-POL</i> ’	99.1/35.3/50.7	89.1/32.1/45.7	82.5/57.7/66.7	81.4/12.5/21.2
	<i>T-POL</i>	96.7/40.5/53.8	86.2/30.5/42.5	84.1/54.1/64.8	76.7/15.2/24.6
	<i>B-POL</i>	95.0/48.6/ <b>61.3</b>	83.2/41.3/ <b>51.0</b>	82.7/58.1/ <b>67.5</b>	74.0/19.9/ <b>30.1</b>
	<i>M-SVM</i>	93.4/46.3/58.9	78.3/31.4/42.7	32.1/28.1/29.1	40.6/32.8/26.4

Note that we actually feed additional “negative” information to *M-SVM* by providing the “others” class with the sampled data that are known to be absent from the *gold standard* of the four relations. However, in most cases, the performance of *M-SVM* is still poorer than the “*POL*” methods. On one hand, since it is believed that unlabeled data can significantly help learning [14] [15], it is intuitive for one to expect that the “*POL*” methods that employ the unlabeled data in learning outperform *M-SVM* that does not. On the other hand, the “*POL*” methods are transductive, which is believed to be better than the inductive one, *M-SVM*, when dealing with sparse training data [22].

**Impact of  $c$ .** Looking at Fig. 5, we can observe that *B-POL* and *T-POL* obtain significantly higher F1-scores on *album-artist* relation when parameter  $c$  is smaller. This is because lower  $c$  settings conservatively identify smaller portion of negatives

that are strongest in *Roc-SN* (Sec. 3), which results in greatly improved recall. The results on *film-director* relation are similar but the changes of F1-scores are less significant along the different settings of  $c$ . The results on the other two relations, excluding *band-member* (#P=40), do not change much with different settings of  $c$ . In *band-member* (#P=40), the precision decreases too much when the smaller amount of negatives identified by *Roc-SN* cannot cover a sufficiently large region. From the investigations, we found that *album-artist* and *film-director* relations are described by strong co-occurrence contexts while those of the other two relations are somehow general. In the cases of strong contexts, the precision would not decrease much when strategies are made to increase recall. Nevertheless, for general contexts, to a certain degree, recall is already guaranteed by the contexts, so the decrease of precision dominates the F1-scores when lower  $c$  values are set.

**Efficiency.** As described in [15], the time complexity of the original *POL* is  $O(|U|^2 * \log|U|)$ , assuming the number of iterations is  $\log|U|$  (here  $U$  represents the set of unlabeled instance). For *B-POL*, although the number of iterations invoking *T-POL* cannot be pre-determined, in the experiments this number is usually around 5 and within 10. Consequently, *B-POL* runs fast and usually takes less than 10 seconds on a Pentium 3.2G Dual-Core CPU. Although the time cost of *B-POL* depends on the size of the unlabeled data, *B-POL* can usually run fast due to the fact that the entity features are first selected to filter out irrelevant data so that a much smaller set of unlabeled data is finally fed into *B-POL*.

**Discussion.** *PORE* aims to extract relationships between Wikipedia entities, where it can make use of the entity features in the data filtering process. Although *PORE* is not intended to extract attributes, it can be applied to extracting various relationships, which is, to some degree, reflected by the demonstrated relations selected based on the criteria mentioned previously. Moreover, the core part *B-POL* is a general learning algorithm since it is independent of the data filtering process. At present, we choose only four relations from the infoboxes for the experimentation. In the near future, we plan to apply *PORE* to many other relations which do not come from the infoboxes. We also plan to apply *B-POL* to extracting attributes from free text.

## 6 Conclusions and Future Work

In this paper, we described the *Positive-Only Relation Extraction (PORE)* framework for relation extraction from Wikipedia text. We proposed *B-POL*, the core algorithm in *PORE*, for relation classification. It makes some extensions to a state-of-the-art positive-only learning approach built upon SVMs. Experimental results demonstrated that *B-POL* achieved significant improvements on the performance, especially when the amount of the training data is small. We also empirically showed that *B-POL* significantly outperforms the multi-class classification approach. In addition, we demonstrated the feature engineering and data filtering components of *PORE*. Although *PORE* is applied in the context of Wikipedia, the core algorithm *B-POL* is a general approach for Ontology Population and can be adapted to other domains.

In the future, we would like to investigate an optimization technique to uncover the best value of parameter  $c$  of *B-POL* given the positive and unlabeled data. We also plan to improve the data filtering component of *PORE* in the near future.



## References

1. Ding, L., Finin, T.: Characterizing the Semantic Web on the Web. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
2. Ramakrishnan, C., Kochut, K.J., Sheth, A.P.: A Framework for Schema-Driven Relationship Discovery from Unstructured text. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
3. Sergey, B.: Extracting Patterns and Relations from the World Wide Web. In: Atzeni, P., Mendelzon, A.O., Mecca, G. (eds.) The World Wide Web and Databases. LNCS, vol. 1590, Springer, Heidelberg (1999)
4. Agichtein, E., Gravano, L.: Snowball: Extracting Relations from Large Plain-text Collections. In: ACM DL 2000 (2000)
5. Pantel, P., Pennacchiotti, M.: Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations. In: COLING 2006 (2006)
6. Ravichandran, D. and Hovy, E.H. 2002. Learning Surface Text Patterns for a Question Answering System. ACL'02.
7. Boer, V., Someren, M., Wielinga, B.J.: Extracting Instances of Relations from Web Documents using Redundancy. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)
8. Cimiano, P., Handschuh, S., Staab, S.: Towards the Self-Annotating Web. In: WWW 2004 (2004)
9. Mori, J., Tsujishita, T., Matsuo, Y., Ishizuka, M.: Extracting Relations in Social Networks from Web using Similarity between Collective Contexts. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
10. Tang, J., Hong, M., Li, J., Liang, B.: Tree-structured Conditional Random Fields for Semantic Annotation. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
11. Chang, C.-C., Lin, C.-J.: LIBSVM: A Library for Support Vector Machines, Software (2001), available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
12. Völkel, M., Kröttsch, M., Vrandečić, D., Haller, H., Suder, R.: Semantic Wikipedia. In: WWW 2006 (2006)
13. Auer, S., Lehmann, J.: What have Innsbruck and Leipzig in common? Extracting Semantics from Wiki Content. In: ESWC 2007 (2007)
14. Yu, H., Zhai, C.X., Han, J.: Text Classification from Positive and Unlabeled Documents. In: CIKM 2003 (2003)
15. Li, X., Liu, B.: Learning to Classify Texts Using Positive and Unlabeled Data. In: IJCAI 2003 (2003)
16. Rocchio, J.: Relevance Feedback in Information Retrieval. In: Salton, G. (ed.) The smart retrieval system: experiments in automatic document processing (1971)
17. Denoyer, L.: The Wikipedia XML Corpus. SIGIR Forum (2006)
18. Suchanek, F.M., Ifrim, G., Weikum, G.: Combining Linguistic and Statistical Analysis to Extract Relations from Web Documents. In: KDD 2006 (2006)
19. Chen, J., Ji, D., Tan, C.L., Niu, Z.: Relation Extraction Using Label Propagation Based Semi-supervised Learning. In: ACL 2006 (2006)

20. Zhang, Z.: Weakly-Supervised Relation Classification for Information Extraction. In: CIKM 2004 (2004)
21. Wang, T., Li, Y., Bontcheva, K., Cunningham, H., Wang, J.: Automatic Extraction of Hierarchical Relations from Text. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)
22. Mitchell, T.M.: Machine Learning. McGraw-Hill, New York (2005)
23. Schölkopf, B., et al.: New Support Vector Algorithms. *Neural Computation* (2000)
24. Wang, G., Zhang, H., Wang, H., Yu, Y.: Enhancing Relation Extraction by Eliciting Selectional Constraint Features from Wikipedia. In: NLDB 2007 (2007)
25. Ruiz-Casado, M., Alfonseca, E., Castells, P.: Automatic extraction of semantic relationships for WordNet by means of pattern learning from Wikipedia. In: Montoyo, A., Muñoz, R., Métais, E. (eds.) NLDB 2005. LNCS, vol. 3513, Springer, Heidelberg (2005)
26. Zhou, G.D., Su, J., Zhang, J., Zhang, M.: Exploring Various Knowledge in Relation Extraction. In: ACL 2005 (2005)
27. Schutz, A., Buielaar, P.: RelExt: A Tool for Relation Extraction from Text in Ontology Extension. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
28. Manevitz, L.M., Yousef, M.: One-Class SVMs for Document Classification. *Journal of Machine Learning Research* 2, 139–154 (2001)
29. Wang, G., Yu, Y., Zhu, H.: Tech. Report. Available at [http://apex.sjtu.edu.cn/apex\\_wiki/Papers?action=AttachFile&do=get&target=wang-iswc07-tr.pdf](http://apex.sjtu.edu.cn/apex_wiki/Papers?action=AttachFile&do=get&target=wang-iswc07-tr.pdf)
30. Zhu, X.: Semi-supervised Learning Literature Survey. TR 1530, Univ. of Wisconsin, Madison (December 2006)
31. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In: WWW 2007 (2007)

# Ontology Performance Profiling and Model Examination: First Steps

Taowei David Wang<sup>1</sup> and Bijan Parsia<sup>2</sup>

<sup>1</sup> Department of Computer Science, University of Maryland,  
College Park, MD 20742, USA

tw7@cs.umd.edu

<sup>2</sup> School of Computer Science, University of Manchester, UK  
bparsia@cs.man.ac.uk

**Abstract.** “[Reasoner] performance can be scary, so much so, that we cannot deploy the technology in our products.” – Michael Shepard [1]. What are typical OWL users to do when their favorite reasoner never seems to return? In this paper, we present our first steps considering this problem. We describe the challenges and our approach, and present a prototype tool to help users identify reasoner performance bottlenecks with respect to their ontologies. We then describe 4 case studies on synthetic and real-world ontologies. While the anecdotal evidence suggests that the service can be useful for both ontology developers and reasoner implementors, much more is desired.

## 1 Introduction

Ontology engineering is the discipline of building a certain class of computational artifacts — ontologies — which typically are a component of a larger software system. Ontologies are used as conceptual models, for data integration, or to directly represent information in a variety of domain areas. Today, most self-described ontology development has used logic-based representation languages to express ontologies, that is, ontologies are (in part) *theories in some logic*. Indeed, the Web Ontology Language (OWL) is based on a description logic and has spurred a large rise in the number of publicly available logic based ontologies [15].

However, as with most interesting logics, the reasoning services which are critical for the development and sometimes the deployment of ontologies have very bad worst case complexity. For example, consistency checking for *SHOIN*, the DL underlying OWL is NEXPTIME complete. While modern reasoners (such as FaCT++, KAON2, and Pellet) employ an increasingly sophisticated array of optimizations, it is still not particularly difficult to stymie them. When the reasoners seem, somewhat randomly, to never halt while performing reasoning services, users often feel lost, frustrated, and helpless. Most do not have the training or the expertise to delve into their reasoner to figure out what is going

---

<sup>1</sup> <http://lists.w3.org/Archives/Public/public-owl-dev/2007JanMar/0047.html>

on, even if the source code is available. Consequently, users try to (1) Remove portions of the ontology that are thought to be the causes of reasoning performance problems, or (2) contact reasoner implementors for a detailed explanation. However, precisely because of the lack of expertise, attempts to remove axioms in order to reduce the computational load often results in efforts a little better than blind guesses. Though experts such as reasoner implementors have general ideas of the type of axioms that can create performance problems, general guidelines may not help solving specific problems. In addition, it can be difficult to identify performance problems due to hard-to-handle logical axioms when there are many easy-to-handle axioms. The lack of tool support and theory in this area is the primary reason that even experts find it a significant challenge to explain the performance of their reasoner against certain ontologies.

The situation in ontology engineering with regard to performance tuning is similar to that with regard to debugging a few years ago. In the past two years or so the state of the tooling for finding and explaining semantic errors in ontology has gone from nothing to respectable [12][11]. The availability of robust services and tools for debugging ontologies allows ontology engineers to build larger, more correct, more interesting ontologies in less time and with less expertise and tedium. Today, performance analysis of ontologies against reasoners is a painful, tedious, manual affair even to identify a bottleneck as a test case for reasoner implementors.

In this paper we take some first steps toward supporting the performance analysis of description logic knowledge bases. In particular, we focus on understanding the effort in testing the *satisfiability* of a class using a tableau reasoner. In the following sections, we relate our research to similarities in other disciplines, describe our prototype tool and case studies on how the tool relays the internal states of a reasoner to help a user understand the performance bottleneck.

## 2 Background and Challenges

We draw the analogy between software engineering and ontology engineering. Both ontologies and software source code are human-written computational artifacts meant to be processed by other programs for a purpose. Source code is to be compiled and run in an environment, and ontologies are to be processed by reasoners for entailments. Software profiling involves collecting various performance statistics during program execution which are correlated with statements in the source code. Similarly, we envision an ontology profiler gathering performance related statistics during reasoning and correlating them with axioms (or terms) in the ontology. We emphasize that we are not interested in profiling the reasoners as programs themselves. We are treating the reasoners as fixed entities, and instrument their behaviors with regards to different parts of an ontology to identify sources of performance bottlenecks. Of course, just as sometimes one has to look to the behavior of the compiler, interpreter, query engine, or runtime libraries to actually solve a performance problem, sometimes the problem can only reasonably be solved by investigating and modifying the reasoner. However,

even in these cases, it can be very helpful if the users can effectively isolate the aspects of their ontology that is causing problems.

In [10], Jeffery identifies four challenges to building successful program monitors (e.g., for debugging or performance analysis), all of which apply to the ontology engineering case:

1. **Volume of Data:** The ontologies themselves can be large and complex enough to require significant tool support even in their asserted form. The search space for reasoning services for expressive description logics is very large (as indicated by their EXPTIME to NEXPTIME worst case complexity). Tableau reasoners build finite graph representations (completion graphs) of models of the given ontologies, and this graph and its construction trace can be large and unwieldy. The challenge here is first deciding what are the most useful data, and subsequently, how to allow users explore and correlated the two data sources to gain insight.
2. **Dimensionality of Data:** As the inference services proceed, data from tracing the reasoners' high-level behavior and the resultant completion graphs are generated. At each time point during the execution, the state of the reasoner can be described by the (incompleted) completion graph. The completion graph does not only contain structural information, but also sets of categorical information as labels. Additional flags (e.g. blocked, cached) designating techniques used by the reasoner also add dimensions to the data. Finally, statistics for raw performance measures should also be collected as performance overviews.
3. **Intrusion:** Software monitors typically must alter the program and the environment in order to gather useful data. Instead of providing execution-time monitoring, which may impose unrealistic slowdown, we employ less invasive "postmortem" methods to allow users to examine the performance data after the execution.
4. **Access:** Many aspects of the execution of inference services are not usually accessible. Users typically use reasoners as an oracle. They ask whether a certain concept is satisfiable, and the reasoner returns "yes" or "no". There is little information available for the user to review. Exposing the graphs and the internals of the reasoners' operations is not a straight-forward task, however. The non-deterministic nature of the tableau algorithms and its optimizations may cause the inference services to behave differently when given the same ontology. How to present these differences and still keep a coherent picture is a challenge. While reasoners can handle many ontologies within a reasonable time frame, sometimes the inference services never seem to end, or all available memory is consumed. When this happens, often no information can be given back to the user. To be able to handle such cases, even at a preliminary level, is key.

In program optimization, we can distinguish between two basic sorts: macro or "algorithmic" or "design" improvements, and micro or "code level" improvements. Changing data structures for one better suited to the problem is an

example of the former. Loop unrolling is an example of the latter. Obviously, there is a continuum between these poles, and some activities — such as tuning the garbage collector — do not obviously fit. It’s not clear, exactly, whether these categories provide a useful framework for thinking about ontologies. There are certainly analogous practices to micro-optimization: sometimes twiddling the axioms can have a large beneficial effect on performance. Many in-reasoner optimization involve transforming axioms into a better form for the reasoner. Similarly, the practice of approximating an ontology originally developed in a more expressive logic in a less expressive logic (see Dolce Lite<sup>2</sup> which is an OWL DL version of a full first order logic based ontology) or approximating difficult constructs (such as approximating nominals as new disjoint atomic classes) can be seen as something of a macro-optimization, at least, as an attempt to macro-optimize. Obviously, as with software engineering, it is important to understand what is lost with such changes. Not all optimizations preserve the exact behavior of the original program. Similarly, some changes to ontologies will not respect the intended representation of the domain and even lose significant entailments. Such are the compromises users face.

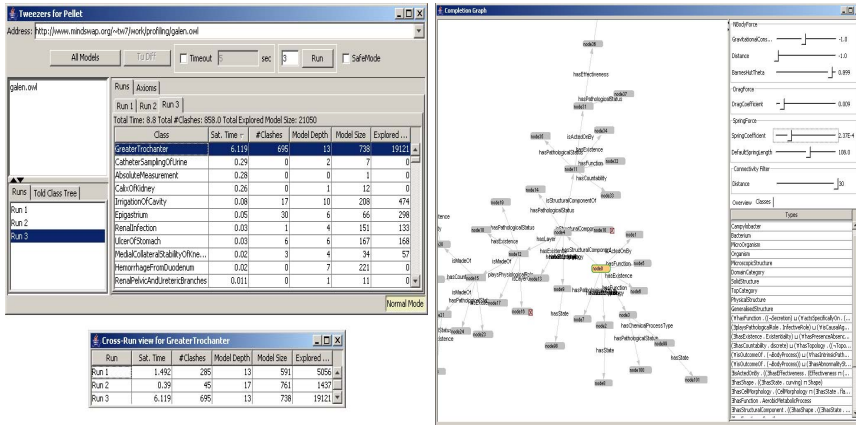
One significant point of potential disanalogy is that reasoning techniques vary widely. While tableau algorithms are still the dominant form of reasoning with OWL DL, other techniques such as reduction to disjunctive datalog<sup>3</sup> and reduction to first order logic, such as Hootlet<sup>3</sup> and MSPASS<sup>9</sup>, are significantly different in most details. Not only is there different information to extract, but the behavior model, thus how to fruitfully interpret the extracted data, is very different. The techniques we explore in this paper were developed for a specific reasoner, Pellet, and thus for a fairly bog-standard tableau reasoner. This obviously is limited, but is not unreasonably specific. Even if the techniques would not translate directly to other tableau reasoners, it is important to determine *whether such services* are useful to ontologists, and how.

### 3 Tool Design and Implementation

Tweezers is a prototype utility that instruments Pellet and allows users to gain access to the inference results and performance statistics. The main interface (See Figure 11) enables users to view and sort a set of performance statistics. We collect the following statistics: *Sat. Time*: The CPU time it takes to perform satisfiability check for a particular class. *#Clashes*: The number of clashes encountered when performing the satisfiability check. This measures how many dead ends reasoner run into before finding a completion. *Model Depth*: The depth of the completion graph. *Model Size*: The size of the completion graph. *Explored Size*: Number of nodes generated but were not in the final completion graph (due to clashes and backjumping). This is a rough measure of “wasted effort”. Upon loading an ontology from the Web, Tweezers automatically performs satisfiability checks on all

<sup>2</sup> <http://www.loa-cnr.it/DOLCE.html>

<sup>3</sup> Hootlet: <http://owl.man.ac.uk/hootlet/>



**Fig. 1.** On the left side is the Tweezers main interface. The statistics are sorted by Sat. Time. A cross-view of the statistics for a class is shown at the bottom. The right side shows the interface for completion graph inspection. The graph is visualized on the left. The right side contains the visualization control, and labels for the currently selected node (highlighted in orange).

classes. The default sat check behavior can be modified via several controls, described below.

For convenience, users can execute several runs in batch fashion with one click. Statistics from multiple runs let users better deal with nondeterminism via averages. Viewing of statistics from different runs is managed by tabs. Double-clicking a row gives users a view of the statistics of that class across all the runs. Double-clicking again here launches the completion graph viewer (See Figure 1). The completion graph is shown as a network in force-directed layout, using the open-source Prefuse library [4]. Each node represents an individual, and each edge represents a set of roles between the two individuals. The nodes are numbered in the order that they are generated. In a panel at the bottom right part of the interface, the labels of the currently selected individual are displayed in the order that they are added. Nodes with special attributes are additionally decorated: the first node is outlined in green, and blocked nodes are shown with a blocked symbol in red.

By clicking on “All Models”, users can perform a run where Pellet finds all possible representations of models for each class. Satisfiability checks typically stop when a first model is found. Here we force Pellet to find all models. These models are represented by a set of completion graphs, and these graphs are available for inspection using the same interactive visualization interface. Interested users can explore these completion graphs to look for qualitative differences.

With each run, users can optionally set a timeout limit (default is 5 seconds) for how much time Pellet is allowed to spend performing satisfiability check for each class. When the timeout is reached, Pellet stops processing that class, and moves onto the next, no statistics are kept for the aborted class.

The statistics and completion graphs are normally kept in-memory for faster access. The size of completion graphs for classes in some ontologies can be in the order of thousands, consuming all memory so that Tweezers fails. To better make use of main memory, Tweezers has a “Safe Mode” that allows the storage of collected data to be made on hard drives. In this mode, maximum amount of memory is ensured for the satisfiability check for every single class. When encountering an out-of-memory error in normal mode, Tweezer will attempt to recover and restart the process in Safe Mode.

Tweezers is built as part of pre-released version of Pellet 1.4 (no OWL 1.1 support). Currently it provides instrumentation for ontologies of expressivity  $SHIN(\mathcal{D})$  (OWL-DL without nominals). However, this is mostly a problem simplification decision, not a technical barrier. This Pellet is bundled with a version of Swoop, which contains useful utilities such as ontology modularization that can additionally help isolate performance problems. These tools can be found here<sup>4</sup>.

## 4 Case Studies

We describe a series of case studies on how our prototype tool help identifying the causes of performance bottleneck. There are four case studies, presented here roughly in the chronological order that they were studied.

### 4.1 Galen, Round 1

One common “difficult” OWL ontology is the version of the Galen<sup>5</sup> medical ontology translated from the original GRAIL formalism by Ian Horrocks, and has over 2700 classes, 400 ObjectProperties, and 300 GCIs. It is in the DL expressivity  $SHF$ . This is an interesting initial case because of the large number of GCIs. We expected to see a lot of variation in the performance statistics when we run satisfiability checks in Pellet. We were interested in finding out if there are consistently “difficult” classes, and if we could fine-tune it.

We initially ran Tweezers on Galen a few dozen times, and ascertained that the class *GreaterTrochanter* consistently has the worst statistics of all classes. The average completion graph size for *GreaterTrochanter* was well over 600, by far the largest in this ontology. We traced a few of the graphs manually, attempting to see if there are axioms that can be added or removed to reduce the complexity of the completion graph. In the graphs, we found that more than one individual is both a *Femur* and a *Tibia*, which seemed counterintuitive. We thought that by making *Femur* and *Tibia* disjoint, we may be able to reduce

<sup>4</sup> <http://www.mindswap.org/~tw7/work/profiling/code/index.html>

<sup>5</sup> Galen: <http://www.cs.man.ac.uk/~horrocks/OWL/Ontologies/galen.owl>. It is important to note that this is a translation of a very old version of Galen (sometimes called “not-Galen”). The current production version of Galen is orders of magnitude larger and is not classifiable by any existing reasoner, see: <http://www.co-ode.org/galen/index.php>



the completion graph size and make Pellet work less hard. After consulting the on-line version of Gray's Anatomy<sup>6</sup> to make sure our intuitions is not wrong (after all, we are not the domain experts!). The classes *Femur*, *Tibia*, *Fibula*, *Humerus*, *Ulna*, and *Radius* are direct subclasses of *LongBone*. We made an additional version of Galen, where these *LongBones* are all pairwise disjoint so we can compare the effects of disjoint axioms applied in a surgical setting as opposed to a more general setting.

We then ran Tweezers 5 times on each of the 3 versions of Galen: original, Femur-Tibia-Disjoint, and LongBones-Disjoint. The averages of the statistics are summarized in Table II.

**Table 1.** Averages of performance statistics over 5 runs each for the 3 versions of Galen. The left columns shows the statistics for performing satisfiability check for the class *GreaterTrochanter*, while the right columns show the same statistics for performing satisfiability for all classes in the ontology.

Scope	GreaterTrochanter					Ontology		
Statistics	Time	Clashes	Depth	Size	Explored	Time	Clashes	Explored
<b>Original</b>	1.3	118.6	12.8	689.2	5814.4	5.94	322.2	7892.2
<b>F+T Disjoint</b>	0.66	95	9.8	396.6	3184	3	297.4	5528.8
<b>LB Disjoint</b>	0.15	16	8	373.4	601.8	2.78	224.8	2756.8

Satisfiability check time for the *GreaterTrochanter* improved roughly 50%. The same statistic also improved roughly 50% for the all-class case. The number of clashes for *GreaterTrochanter* is reduced by 45%. The completion graph size is also reduced from roughly 700 to about 400. In *LongBones-Disjoint*, the satisfiability check time for *GreaterTrochanter* further improved so that it is an order of magnitude faster than the original. The number of explored nodes for *GreaterTrochanter* is greatly reduced (from 3184 in the *Femur-Tibia-Disjoint* to 601). However, the other improvements are too minor to be considered of any importance.

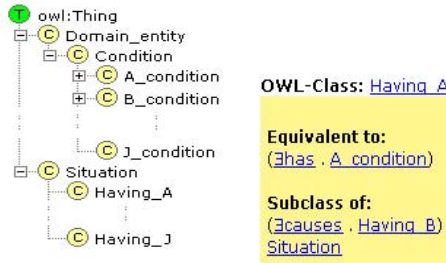
## 4.2 Causal Chains

Modeling causal relationships for diseases and diagnosis is common in biomedical ontologies. The Galen team in Manchester has devised a small test ontology to see if a particular way of modeling can capture the intended knowledge and allow reasonable performance in reasoners for real applications. This Causal-Chain ontology contains 43 classes, 4 object properties (2 pairs of inverses: *has*, *is\_had\_by*, *causes*, *is\_caused\_by*), and has the DL expressivity  $\mathcal{ALCI}$ . The class structure has two main branches. The first branch is a simple hierarchy of *Conditions*. The second branch contains the class *Situation* and its list of children (See Figure 2). Each child of *Situation* is named *Having\_X*, where *X* corresponds to a mirror class *X\_Condition* in the first branch. Generally, every *Having\_X* have the following axioms:

<sup>6</sup> Gray's Anatomy: <http://education.yahoo.com/reference/gray/>

1.  $Having\_X \equiv \exists has.X\_Condition$
2.  $Having\_X \sqsubseteq \exists causes.Having\_Y$
3.  $Having\_X \sqsubseteq \exists Situation$

where  $Y$  stands for the letter that occurs one after  $X$  (See Figure 2). The implication is that every  $Having\_X$  causes a  $Having\_Y$ , and this causal chain goes from  $Having\_A$  to  $Having\_J$ . We were told that FaCT++ was able to classify this ontology in seconds, but Pellet could not classify this ontology. We theorized that the causal chain was the problem – since the ontology is very small, and the structure is regular and simple. We would like to make causal chains of different length, and see which ones Pellet can process. Using Swoop, we modularized the ontology (as described in [3]) along the causal chain, once for every  $Having\_X$  class. We end up with 10 modules of increasing complexity, from  $Having\_J$  to  $Having\_A$ . Pellet can classify all module except the ones for  $Having\_B$  and  $Having\_A$ . Completion graphs revealed that there were many unexpected disjunctions in the nodes’ labels. Moreover, Pellet debugging message showed that the completion rules for domain axioms are being fired. The only problem was that there were no domain axioms in the ontology.



**Fig. 2.** On the left side, an abbreviated version of the class tree is shown. On the right hand side, detailed class definition for  $Having\_A$  as is shown in Swoop.

We later determined that the GCIs in this ontology are handled by a reasoner optimization called “role absorption”. The GCIs 4. and 5. are transformed into the domain axioms 6. and 7. by role absorption [14]:

4.  $\exists has.Condition\_X \sqsubseteq Situation$  (from 1. and 3.)
5.  $\exists has.Condition\_X \sqsubseteq \exists causes.Having\_Y$  (from 1. and 2.)
6.  $domain(has, Situation \sqcup \neg \exists has.Condition\_X)$  (from 4.)
7.  $domain(has, \exists causes.Having\_Y \sqcup \neg \exists has.Condition\_X)$  (from 5.)

If there are  $n$  such domain restrictions, every node that had a  $has$  successor would be labeled with at most  $n$  complex domain types, where each type contained a disjunct of 2 class expressions, for a total of  $2^n$  possible choices. From  $Having\_A$  to  $Having\_J$ , there were 19 such GCIs ( $Having\_J$  causes nothing), thus we had a search space of cardinality  $2^{19}$  for every node that generated

a *has* relation. To make matters worse, the presence of inverse properties means that more sophisticated blocking techniques needed to be used in the tableau to ensure the correctness of the inferences [7]. As a result, the size of completion graphs are usually much larger, and inference takes longer.

We found two possible ways to tune it for better performance. The first one is to remove the inverse assertions. The second one is to change the the  $\equiv$  assertions from the *Having\_X* classes to  $\sqsubseteq$ . Both of these techniques reduced the classification time for Pellet from impossible to within seconds. The first technique lowers the expressivity of the language used so that the reasoner could block more easily, reducing the amount of work. In the second case, removal of the equivalence axioms means we get rid of the GCIs, and no role absorption would occur. Though both solutions makes the ontology manageable, they also come with a price. In the first case, if inverses were the central focus of modeling, then it would be unwise to remove them. On the other hand, if the intended use of the ontology is for instantiation (determining what classes individuals are members of), then the second method would not be appropriate. This decision should be made by the ontology creators and application developers.

### 4.3 LKIF-Core Ontology

The LKIF is a suite of OWL ontologies that describe the legal domain [7]. A merged version of a snapshot of the ontology from late February 2007 is used in this discussion, and can be found here [8]. The ontology is not large, containing 206 classes, 1 data type property, 106 object properties, and no individuals. It has the DL expressivity  $SHIN(\mathcal{D})$ . There are 75 inverse properties. Neither FaCT++ (called from Protégé 4.27) nor Pellet was able to classify this ontology in reasonable amount of time. Desires to find out more about the behaviors of the reasoner with respect to the ontology was expressed in the Pellet user mailing list. In particular, the user would like to know whether there is a specific construct or pattern that has made the ontology unprocessable [9].

In the Causal-Chain case study, it was easy to isolate the potential problems by manually identifying the difficult part of the ontology because of its size and regularity in structure. We then modularized the ontology to examine the effects of the length of causal chain on performance. However, it is not so obvious here. Manually inspecting the asserted axioms showed that GCIs formed through equivalence axioms and subclass axioms in a hierarchy (as in the Causal-Chain case) were abundant. Class definitions also use the inverse properties frequently. We expected that much of the ontology will be problematic for the reasoner. However, blindly modularize the ontology did not seem a fruitful method. We used the timeout feature of Tweezers to restrict the time Pellet can use to perform satisfiability check for a particular class. The timeout was originally set to be 5 seconds. As expected, majority of the classes could not be checked for

<sup>7</sup> LKIF-Core ontology web page: <http://www.estrellaproject.org/lkif-core>

<sup>8</sup> <http://www.mindswap.org/~tw7/work/profiling/others/lkif-all-correct.owl>

<sup>9</sup> <http://lists.owlidl.com/pipermail/pellet-users/2007-February/001257.html>

satisfiability within the timeout(151 out of 206). Of the ones that could be, 17 were unsatisfiable. The satisfiable classes all had fairly small models, and took no more than 0.1 second to perform satisfiability check, and were without clashes. In this case, very little useful information was available to guide us to further isolate the problems. By extending the timeout limit to 10 seconds, we were able to see Tweezers process 90 classes (including the 17 unsatisfiable ones). Of these classes, many have completion graphs of the same size and depth, indicating that they may be mostly the same, sharing a common, large structure. However, the size of such graphs (over 3000+) made detailed manual inspections prohibitive. Instead, we suggested to the users to remove non-critical equivalence axioms and make less important properties not inverse, symmetric, or inverse functional. Pellet could process the ontology within seconds once these property attributes were removed.

There are a few reasons why this was an interesting case study. One, the situation that users get frustrated with reasoner's performance with respect to their own ontologies was consistent with our expectation, especially when the use of reasoners is not part of the ontology development cycle (as suggested by the existence of many unsatisfiable classes). Secondly, more sophisticated users of the OWL language desire to gain more understanding of their own ontology when this situation arises, and only finds (1) the current tools are not easily amendable to exposing the internal states and histories, and (2) when an ontology is unprocessable as a whole, the users get nearly no information about their ontology (even though part of it is processable). Our prototype tool attempted to give users more feedback, though in this particular case, only generic advice could be rendered without systematic examination and experimentation of large number of "difficult" classes.

#### 4.4 Galen, Round 2

The above case studies showed that having inverse properties in even small, but expressive ontologies can have far-reaching performance consequences. We returned to the ontology Galen and performed an experiment by adding an inverse property. There are two main object property branches in Galen. One is rooted at *DomainAttribute*, while the other is rooted at *InverseDomainAttribute*. These two branches mirror each other, and each branch has about 200 properties. Though the name suggests that the properties in one branch would be inverses of the corresponding ones in the other, no actual inverse assertions exist. For our experiment, we selected one pair of corresponding object properties to be inverses of each other: *ActsOn*, *isActedOnBy*. These two properties were used in definitions of 61 classes. When looking at the inferred hierarchy of Galen, there were a total of 385 classes effected by these definitions because they were descendants of the 61 classes. Pellet was not able to classify this ontology. Running Tweezers with a 5 second timeout showed that satisfiability of 900+ of the 2700+ concepts could not be checked. Our friend GreaterTrochanter belonged to that difficult group. In the original Galen, only GreaterTrochanter had any notable size in its completion graph. In this single-inverse version, many classes

have completion graphs of sizes over 10k. Even the simple classes that did not seem to have any connection to the inverse properties would have sizes in the order of thousands!

## 5 Discussion

These case studies exhibited all the facets of the research problem. First, there is a real need from the user community. Instrumenting the reasoner and allow users to view the performance statistics is one way to help them identify the problem which, aside from helping them cope with the particular problem, can increase their overall satisfaction with the process. We received very positive feedback from users about our analyses and they were very vocal about obtaining access to Tweezers. It seems that even if modifications to their ontologies would not be acceptable, simply *knowing* what the problem qualitatively improves their experience. It is shown in the first case study that when the ontology can be classified by a reasoner, it is possible to fine-tune the ontology via adding or removal of axioms to improve the reasoning performance. We showed that a strategically placed disjoint axiom can greatly reduce the performance time for both a specific concept and the entire ontology. The intuition is that the disjoint axioms restricts the reasoner from adding labels, thus limiting the size of the completion graph, and possibly pruning search space. However, it is possible that too many disjoint statements may cause performance problems, as this may restrict the reasoner too much, and the overhead for many backtracks catches up. We conjecture that there is a “right amount” of disjoints that can optimize the performance for many ontologies. We are currently investigating methods of determining such “right amount”.

In small ontologies with very regular structures such as the Causal-Chain ontology, manual inspection of the axioms was often enough to have an inkling of what might be causing the problem. Modularizing the ontology proved to be an effective method to simplify the problem and identify the bottleneck. In cases like LKIF with 5-second timeouts, the classes were either so easy that their performance statistics are of no insight, or that the classes were so difficult that no useful information can be collected, finding what classes with respect to which to modularize is a challenge. The LKIF case study also revealed that large completion graphs are prohibitive for both the tools to display and humans to digest. More automated methods should be used to look for points of interest in the completion graph.

The final 3 case studies showed that the presence of inverse properties can drastically change reasoner performance. They revealed how inefficiently inverses are handled. Currently, Pellet chooses a strategy for performing inference services by the expressivity of the ontology. For example, when an ontology contains inverse properties, a more sophisticated blocking technique is employed, and this technique is used for all inference tasks for the entire ontology. However, it may be the case that there is a large part of the ontology that does not use inverses (such as in our last case study), and can use a simpler blocking

technique without compromising the correctness of the inference. This may be achieved by investigating the possibility of performing modularization for each class prior to satisfiability check and use the expressivity of such module instead of the ontology, or having the reasoner dynamically change the strategy while reasoning.

## 6 Related Work

To our knowledge there is has been no previous work on providing *ontologists* with tools which attempt to explain the reasoning time characteristics of description logic reasoners in any detail. Reasoners like FaCT++ and Pellet can be configured to provide some feedback during the reasoning process, but this is generally limited to fairly coarse grain timing information (for example, such as would be presented in a progress bar). Sometimes, there is some correlation presented between time and certainly terms (e.g., a reasoner may print start and end points for testing the satisfiability of a particular class), but never to particular axioms or to particular internal behaviors (e.g., backtracking). Also, reasoners tend to be rather unforgiving if there is a memory or timeout problem with an ontology. For example, if a single subsumption test runs out of memory, FaCT++, Pellet, and Racer simply abort the entire classification process (sometimes aborting the the entire session and requiring a restart of the reasoner). As users and developers of reasoners and ontology development environments for several years, we were surprised that it had not occurred to us that this extremely user-hostile behavior was not a good idea. This all-or-nothing approach is deeply embedded in the culture even though, in principle, many of the subsumptions are easy to compute and are accessible, albeit in a clumsy way, from the various APIs. Just lifting this draconian behavior is useful, although raises interface challenges on how to present the resource failures.

Ontology development environments such as Swoop and Protégé provide a syntactic analysis of ontologies designed to help users gain some sense of the overall complexity of an ontology. These analysis include statistics on the number of classes, properties, and individuals and the number of GCIs, inverse properties and the like. They also attempt to classify more precisely the description logic the ontology falls into. This feature first appeared in Swoop and was inspired by the “species validation” service defined by the OWL specifications. The species of OWL — Lite, DL, Full — were intended to give a very course grained idea of the “difficulty” of reasoning with an ontology, all other things being equal. this difficulty is based on the worst case complexity and general experience with reasoning with the corresponding logics. Given that the species’ worst case complexity range from EXPTIME to undecidable, and that there is considerable expressive overlap between the species, it is clear that they are not a particularly helpful guide to expected performance, though they have been used as such. Recently, there has been a renaissance in the area of description logics with tractable (by some measuers) worst case complexity. These logics in

some sense promise better scalability (and their novel inference techniques have cracked ontologies that current tableau reasoners fail to handle), although this can vary significantly in particular cases. They also involve significant expressivity compromises. The idea of “light weight species” has considerable appeal to users as shown by the enthusiasm generated by the OWL 1.1 “tractable fragments” document<sup>[10]</sup>

There is a growing literature on description logic reasoner benchmarking, though the majority is embedded in discussions of optimizations or new reasoning techniques, e.g. [6]. There is a line which extends from the modal logic community’s attempt to generate “hard modal formulae” [5]. Unfortunately, relating these benchmarks to analysis of difficult ontologies (for certain reasoners) has not been systematically attempted.

Model and proof extraction and presentation has a number of uses from education [1] to debugging missing entailments [13]. One of our long term goals is to explore support for *model oriented* ontology development [2]. One reason to focus on the use of model/tableau visualization and exploration in a profiling context is that we do not need domain experts in order to perform useful experiments. Subjects can be set tasks such as “improve the performance of satisfiability testing while minimizing the loss of subsumptions” and model generation and exploration tools without them needing to understand the subject matter of the ontology. In this way, we believe that performance tuning has methodological value independently of its substantive value to users.

## 7 Future Work and Conclusions

Through our case studies, we have demonstrated the difficulty of ontology performance profiling, and how our prototype implementation can help. However, these case studies also revealed the limits of our current tool. Most glaringly, tools and users get easily overwhelmed by multiple, very large completion graphs. The development of more sensible user interfaces, coupled with a more complete coverage of reasoner behaviors (such as axiom-level profiling and completion graph construction history) for exploration and analysis of completion graphs is critical. We also need to study how experts use these tools so the process can be more automated, and more explanations can be given to the average OWL users. Indeed, if the performance is to be *scary*, it should not be scary because of the *unknown*.

## Acknowledgments

This work was supported in part by grants from Fujitsu, Lockheed Martin, NTT Corp., Kevric Corp., SAIC, the National Science Foundation, the National Geospatial Intelligence Agency, DARPA, US Army Research Laboratory, and NIST. Special thanks to Evren Sirin for his invaluable discussions and help.

<sup>10</sup> <http://www.w3.org/Submission/2006/SUBM-owl11-tractable-20061219/>



## References

1. D'Agostino, M., Endriss, U.: Winke: A proof assistant for teaching logic. Proceedings of the First International Workshop on Labelled Deduction, 1998 (1998)
2. Fermüller, C., Leitsch, A., Salzer, G.: Automated model building as future research topic
3. Grau, B.C., Horrocks, I., Kazakov, Y., Sattler, U.: A logical framework for modularity of ontologies. In: IJCAI 2007. Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (2007)
4. Heer, J., Card, S.K., Landay, J.A.: Prefuse: a toolkit for interactive information visualization. In: CHI 2005. Proceedings of the Conference On Human Factors In Computing Systems (2005)
5. Hladik, J.: A generator for description logic formulas. In: Proceedings of the International Workshop on Description Logics, 2005 (2005)
6. Horrocks, I., Patel-Schneider, P.F.: Comparing subsumption optimizations. In: DL 1998. Proc. of the 1998 Description Logic Workshop. CEUR, vol. 11, pp. 90–94 (1998), <http://ceur-ws.org/>
7. Horrocks, I., Sattler, U.: Optimised reasoning for shiq. In: ECAI 2002. Proceedings of the 15th European Conference on Artificial Intelligence (2002)
8. Hustadt, U., Motik, B., Sattler, U.: Reducing shiq description logic to disjunctive datalog programs. In: KR 2004. Proceedings of the 9th International Conference on Knowledge Representation and Reasoning (2004)
9. Hustadt, U., Schmidt, R.A.: Mspass: Modal reasoning by translation and first-order resolution. In: Automated Reasoning with analytic Tableaux and Related Methods (2000)
10. Jeffery, C.L.: Program Monitoring and Visualization. Springer, Heidelberg (1999)
11. Kalyanpur, A., Parsia, B., Sirin, E., Cuenca-Grau, B.: Repairing unsatisfiable concepts in owl ontologies. In: ESWC 2006 (2006)
12. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging unsatisfiable classes in owl ontologies. Journal of Web Semantics (2005)
13. McGuinness, D.: Explaining Reasoning in Description Logics. PhD thesis, New Brunswick, New Jersey (1996)
14. Tsarkov, D., Horrocks, I.: Efficient reasoning with range and domain constraints. In: Proceedings of the 2004 Description Logics Workshop (2004)
15. Wang, T.D., Parsia, B., Hendler, J.: A survey of the web ontology landscape. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)



# The Semantic Web and Human Inference: A Lesson from Cognitive Science

Takashi Yamauchi

Mail Stop 4235, Department of Psychology,  
Texas A&M University College Station, TX 77843 USA  
tya@psyc.tamu.edu

**Abstract.** For the development of Semantic Web technology, researchers and developers in the Semantic Web community need to focus on the areas in which human reasoning is particularly difficult. Two studies in this paper demonstrate that people are predisposed to use class-inclusion labels for inductive judgments. This tendency appears to stem from a general characteristic of human reasoning – using heuristics to solve problems. The inference engines and interface designs that incorporate human reasoning need to integrate this general characteristic underlying human induction.

## 1 Introduction

In order for the Semantic Web to remain technologically viable, it must either drastically expand human abilities or create totally new experiences in our lives. Many inventions have achieved this: the automobile expanded people’s mobility; TV elevated the scope of our experience; E-mail facilitated the ability to communicate. Unless Semantic Web can change human experience and ability in a fundamental manner, its technological innovation will be short-lived.

How can the Semantic Web change our lives? One goal of the Semantic Web is to aide complex inferential tasks such as medical diagnosis, business decisions, or investment predictions by making the vast amount of data now available on the World Wide Web useful for the machine [1]. However, to develop successful inference agents, semantic web researchers need to know exactly what is needed to supplant human inferential behavior – how people make inferential reasoning, when their reasoning becomes irrational and fallible, and how a computer program can help fulfill our shortcomings.

The question therefore boils down to the following: are the existing approaches, such as RDF, RDFS, OWL, and other inferential agents, sound enough in principle to support human reasoning abilities? If not, what is needed for the evolution of the Semantic Web? Study in human cognitive psychology can make an important contribution in this regard.

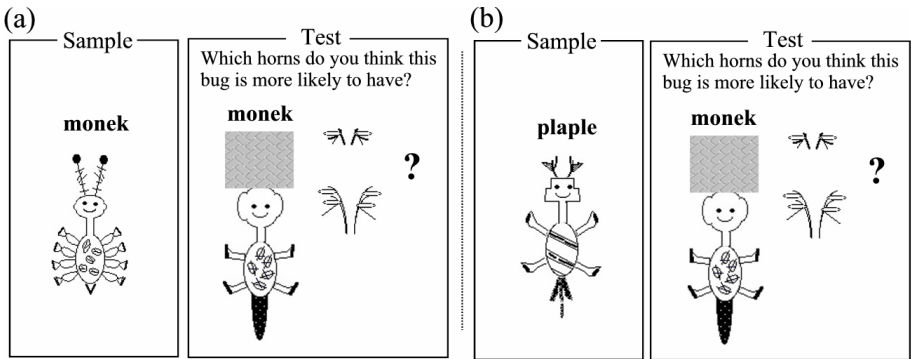
In this paper, I will describe two cognitive studies conducted in my laboratory and illustrate how people (college students) solve simple inferential questions using ontological “tags.” In so doing, I aim to provide insight into the development of

inferential agents in the Semantic Web environment from the perspective of human cognitive psychology.

In a nutshell, the present studies show two things: one, that people display a significant tendency to use class-inclusion labels to make inferential judgments, and two, that they separate category labels from property labels and make “rule-like” reasoning. This disproportional reliance on categorical class labels appears to stem from two sources: (a) human reasoning is intertwined with language, thus linguistic categories, such as count nouns and adjectives, influence the way people make inductive reasoning; (b) because of processing constraints, people often apply simple heuristics, such as using class-inclusion labels, to solve inferential problems [2]. The inference engines and interface designs that incorporate human reasoning need to take advantage of these fundamental characteristics of human reasoning in order to be successful. Specifically, it may be useful to separate ontological status of class-information (applicable for count nouns) and property information (applicable for adjectives). Furthermore, applications of Semantic Web technology should focus on the areas in which human reasoning is particularly vulnerable.

## 2 Study 1

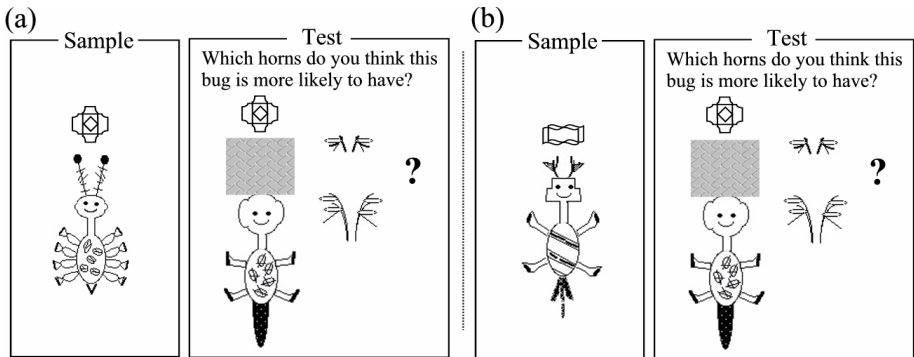
Consider a simple reasoning experiment, in which subjects are shown schematic pictures of cartoon insects side by side and are asked to predict a hidden value of an attribute of a test insect on the basis of a sample insect (Fig. 1). One selection is consistent with the attribute value shown in the sample insect (e.g., long horns in Fig. 1a). The other selection is inconsistent with the feature shown in the sample insect (e.g., short horns in Fig. 1a).



**Fig. 1.** Two examples of a stimulus frame in the inference task in Studies 1 and 2. In one group of trials, both sample and test stimuli had the same labels (“monek” and “monek”). In the other group of trials, sample and test stimuli had different labels (“plaple” and “monek”). Subjects were asked to predict the hidden value of body parts in test stimuli on the basis of sample stimuli shown right next to the test stimuli. In this manner, this experiment assessed the extent to which the attribute value of one stimulus (sample stimulus) is projected to other stimuli.

Above each insect, an arbitrary tag (“monek” or “plaple”) is placed. The main question is how the probability of selecting the consistent attribute (selecting the long horns in Fig. 1a) would change as the ontological status of the arbitrary tags is modified. As subjects rely on the labels to make judgments, the probability of making a consistent choice would go up or down sharply depending on whether sample and test stimuli have the same tags (Fig. 1a; “monek” vs. “monek”) or different tags (Fig. 1b; “plaple” vs. “monek”). For example, many subjects would select long horns in Fig. 1a (consistent choice), but few subjects would choose short horns in Fig. 1b. The question is how people’s inferential behavior would change when these arbitrary tags carry different ontological information, class inclusion information or property possession information.

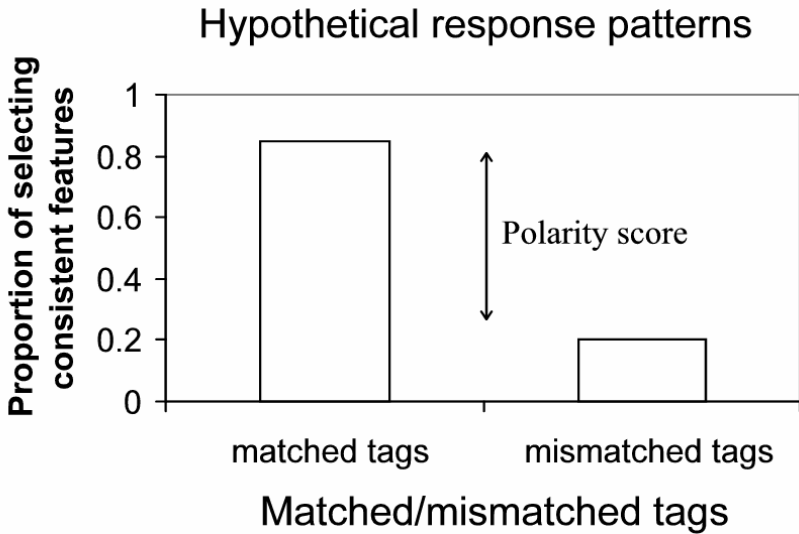
In four independent experiments, a group of undergraduate students received the same stimuli and answered the same inferential questions (60 trials in total). However, the instructions they received were modified slightly so that these arbitrary labels (“monek” and “plaple”) represented different ontological information. In a class-tag condition, the instructions characterized the two arbitrary labels (“monek” and “plaple”) as representing two *types* that these insects belong to. In a property-tag condition, the instructions characterized the same arbitrary labels as representing shapes of wings hidden underneath the insect’s body. In a pictorial class-tag condition and a pictorial property-tag condition, the labels were removed and replaced with pictorial symbols (Fig. 2).



**Fig. 2.** Two examples of the stimulus frames used in the pictorial class-tag and pictorial property-tag conditions in Study 1. Note that the verbal labels (“monek” and “plaple”) in Fig. 1 are replaced with pictorial signs in these conditions.

In this manner, the class-tag condition characterized these labels with class inclusion information, and the property-tag condition characterized these labels with “property” information. In both pictorial class-tag and pictorial property-tag conditions, the two verbal labels (“monek” and “plaple”) were replaced with pictorial signs. These pictorial signs were described as representing two “types” of insects (pictorial class-tag condition) or different shapes of insects (pictorial property-tag condition). Except for these points, all subjects received the same stimuli and answered the same inferential questions.

The probability of selecting attributes consistent with sample stimuli would go up or down drastically depending on whether sample and test stimuli have the same or different labels (Fig. 1 and Fig. 3). However, this shift would depend on the ontological information that these tags represent. For example, when the tags represent names of the “types” that these insects belong to, the probability of selecting attribute values consistent with the sample stimuli (i.e., “consistent choice”) would go up. On the other hand, when the same tags represent a property that these insects have, the probability of consistent choice would go down. In this manner, this study helped identify the extent to which different ontological tags (class information vs. property information; verbal labels vs. pictorial signs) would influence the inference of unknown attributes [3].



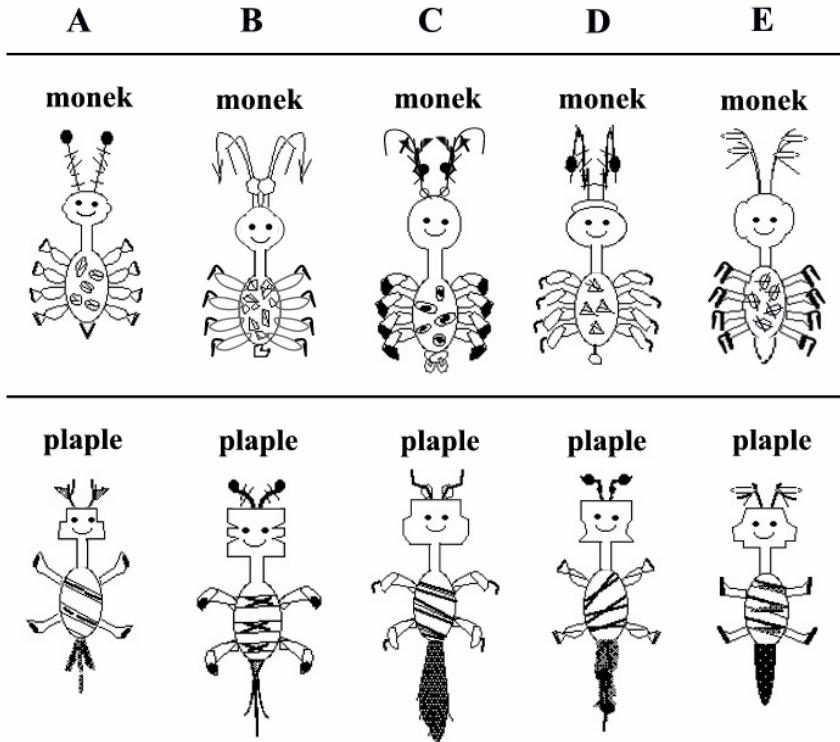
**Fig. 3.** A hypothetical response pattern. This response pattern shows a heavy reliance on tags. As the sample and test stimuli have the same tags (see Fig. 1a), the proportion of making consistent choices is high. However, when the sample and test stimuli have different labels, the proportion of making consistent choices goes down sharply. The difference between the two proportions is defined as a “polarity score.”

In Study 1, I present the results from four experiments, and show a clear indication that people tend to rely on verbal labels representing class-inclusion information to make inferential projections. Study 2 further shows that this bias is present even when “class” division does not have proper meaning, suggesting that using class-inclusion labels is likely to be a default strategy for human reasoning.

## 2.1 Method

### 2.1.1 Participants

A total of 175 undergraduate students participated in this study (the class-tag condition,  $n=49$ ; the pictorial class-tag condition,  $n=51$ ; the property-tag condition,



**Fig. 4.** Five sets of prototypes used to produce test stimuli. One hundred test stimuli were produced by swapping two feature dimensions of two corresponding categories, “monek” and “plaple.” See also Table 1.

$n=41$ ; the pictorial property-tag condition,  $n=34$ ). The participants were recruited from the Psychology Subject Pool administered at Texas A&M University.

### 2.1.2 Materials

The stimulus materials were schematic illustrations of cartoon insects produced from five sets (A, B, C, D, and E) of prototypes (Fig. 4). Each stimulus was composed of a combination of five feature dimensions with binary values (horns=long/short, head=round/angular, torso=dotted/striped, legs=8 legs/4 legs, tail=short/long), along with category labels (‘monek’/‘plaple’) (Table 1). The depiction of these features varied across the five stimulus sets, while they maintained abstract commonalities. For example, the monek prototypes in the five sets all had long horns, round heads, dotted torsos, 8 legs and short tails, while the exact appearance of these components was different across the sets.

Individual trials consisted of pairs of a sample stimulus and a test stimulus (Fig. 1). We created 100 test stimuli from the five sets, A, B, C, D, and E (20 stimuli from

each set – Fig. 4). These test stimuli were produced systematically by exchanging individual components of the two prototypes. All test stimuli had 2 features consistent with the prototype of one category and 2 features consistent with the prototype of the other category, and 1 feature was masked for an inference question (Table 1). One version of the stimulus materials was produced from prototype sets A, B, and C. The other version was produced from prototype sets A, D, and E (Fig. 4). The sample stimuli, which were shown right next to the test stimuli, were the two prototype stimuli of set A.

**Table 1.** The structure of the test stimuli used in Studies 1 and 2. (1, 0)=Horns(long horns, short horns), Head(round, angular), Body(dotted, striped), Legs(8 legs, 4 legs), Tail(short, long), Labels(monek, plaple). *?/I* refers to the dimension queried in each trial, and the “consistent response” for that question. For example, given test stimulus M1, the horns dimension was queried, and the selection with 1 (long horns) is defined as a “consistent response.”

	Horns	Head	Body	Legs	Tail	Labels
M1	<i>?/1</i>	1	1	0	0	1
M2	1	1	0	0	<i>?/1</i>	1
M3	1	0	0	<i>?/1</i>	1	1
M4	0	0	<i>?/1</i>	1	1	1
M5	0	<i>?/1</i>	1	1	0	1
<hr/>						
M0 (sample stimulus)	1	1	1	1	1	1
<hr/>						
P1	<i>?/0</i>	0	0	1	1	0
P2	0	0	1	1	<i>?/0</i>	0
P3	0	1	1	<i>?/0</i>	0	0
P4	1	1	<i>?/0</i>	0	0	0
P5	1	<i>?/0</i>	0	0	1	0
<hr/>						
P0 (sample stimulus)	0	0	0	0	0	0

### 2.1.3 Procedure

For each trial, participants were shown a pair of sample and test stimuli on a computer screen, and were instructed to select one of two feature values for the body part in question. Every participant received 30 test stimuli twice (60 trials in total). In one case, a test stimulus was paired with the prototype of the corresponding

category (i.e., *match* condition) (Fig. 1a). In the other case, the same test stimulus was paired with the prototype of the other category (i.e., *mismatch* condition) (Fig. 1b). For example, stimulus M1 in Table 1 was shown twice, once with the sample stimulus M0 (this is called a *matched* trial because M1 and M0 had the same label ‘monek’) and once with the sample stimulus P0 (this is called a *mismatched* trial because M1 and P0 had different labels, ‘monek’ and ‘plaple’). Each participant received a total of 60 trials (20 test stimuli each from sets A, B, and C or sets A, D, and E – Fig. 4).

Participants indicated their responses by clicking one of the two designated buttons. The order of presenting stimuli was determined randomly for each participant.

## 2.2 Results and Discussion

Each condition was treated as an independent study (no participants participated in these experiments more than once). Because these experiments were given in separate semesters, the four experiments were compared by a meta-analytic procedure [4]. Specifically, we compared the effect size  $r$  of the polarity score (Fig. 3) obtained in each experiment with the following equations:

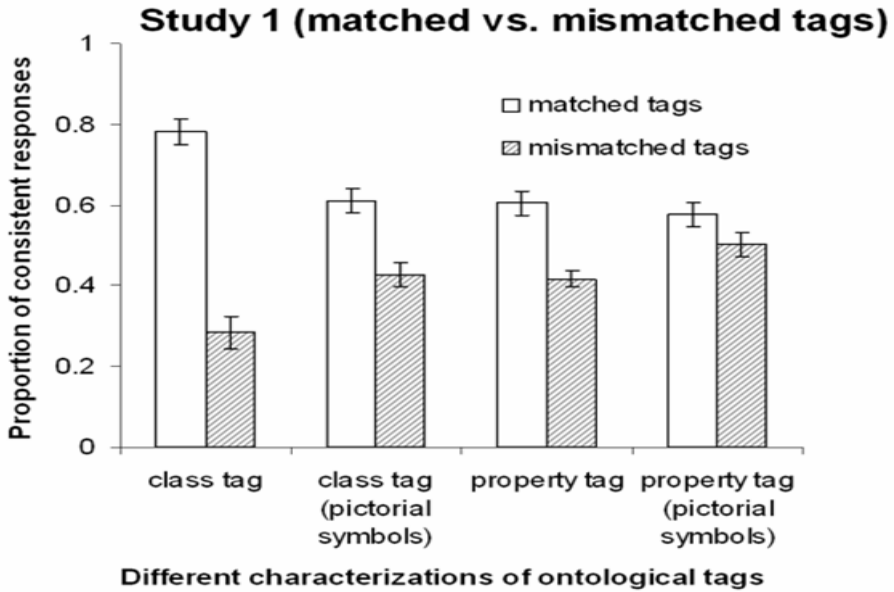
$$\text{Effect size } r = \sqrt{\frac{t^2}{t^2 + df}} \quad (1)$$

$$\text{Fisher } Z_r = \frac{1}{2} \log_e \left( \frac{1+r}{1-r} \right) \quad (2)$$

$$\frac{Z_{r1} - Z_{r2}}{\sqrt{\frac{1}{N_1 - 3} + \frac{1}{N_2 - 3}}} \quad (3)$$

where  $N_1$  and  $N_2$  represent the number of subjects in each of two experiments, and (3) is distributed as  $Z$ .

Fig. 5 and Table 2 summarize the major results from Study 1. As Fig. 5 shows, the impact of matched/mismatched labels was considerably larger when the labels were characterized with class inclusion information (class-tag condition). When sample and test stimuli had the same tags, the proportion of making consistent responses was high. When sample and test stimuli had different tags, the proportion of making consistent responses declined sharply. This tendency decreased substantially when the same labels were characterized with property information. It is also evident that verbal labels, but not pictorial signs, were more important for inferential projections. When the verbal tags were replaced with pictorial signs, the polarity score declined substantially.



**Fig. 5.** A summary of Study 1. The error bars represent two standard error units obtained in each condition.

**Table 2.** Main results from Study 1

	class tag	property tag	class tag (pictorial symbols)	property tag (pictorial symbols)
effect size ( <i>r</i> )	0.79	0.50	0.52	0.36
Fisher Z	1.08	0.55	0.58	0.38

Comparing Fisher Z's in all pairs (p-value)

	class tag	property tag	class tag (pictorial symbols)	property tag (pictorial symbols)
class tag	-			
property tag	0.01	-		
class tag (pictorial symbols)	0.01	0.14	-	
property tag (pictorial symbols)	0.001	0.71	0.89	-



Overall, the effect size comparing the impact of matched and mismatched tags was particularly large in the class-tag conditions as compared to the other conditions;  $Z's > 2.37$ ,  $p's < 0.01$  (Table 2), suggesting a clear inclination to use category labels for the prediction of body parts.

### 3 Study 2

Study 1 indicates a strong preference for using class-inclusion tags for inferential predictions. It is likely that using these categorical labels is a default strategy for inferential reasoning, and Study 2 tested this idea. In Study 2, the instructions in one condition (a random-class tag condition) explicitly stated that the two “classes” (“monek” and “plaple”) of these cartoon insects were determined randomly by a coin toss, and there was no particular meaning attached to these types. This condition was compared to other property-tag conditions, in which the same arbitrary tags were characterized as representing names of diseases that these insects carry, or names of the islands on which these insects live. If using class-inclusion tags is a default reasoning strategy, the polarity score obtained in the random-class tag condition should be still larger than those obtained in the other property-tag conditions.

#### 3.1 Method

##### 3.1.1 Participants

A total of 164 undergraduate students participated in the experiment for course credit (random-class tag,  $n=57$ , property-tag (disease),  $n=49$ ; property-tag (island),  $n=58$ ).

##### 3.1.2 Materials and Procedure

One version of the stimulus materials was produced from prototype sets A and B. The other version was produced from prototype sets A and C (Fig. 4). In total, each participant received 40 test trials. In the random-class tag condition, the instructions specified the arbitrary tags representing two types of the insects that were determined randomly by a coin toss. In the other property-tag conditions (disease and island), the instructions stated that these tags represent names of diseases or islands that these imaginary insects carry or live on. Except for this single point, the three conditions were identical in their procedures, materials, and designs. Individual participants participated in one of the 3 experiments, and no participants participated in the experiment more than once.

#### 3.2 Results and Discussion

As in Study 1, each experiment was treated as an independent study (no participants participated in these experiments more than once), and these studies were examined with a meta-analytic procedure by comparing the effect sizes of the polarity scores obtained in each experiment.

Fig. 6 and Table 3 summarize the major results from Study 2. The most important result in Study 2 is that the polarity score obtained in the random-class tag condition is still substantially higher than those obtained in the other property-tag conditions, suggesting that using class-inclusion labels is somewhat automatic for many participants.

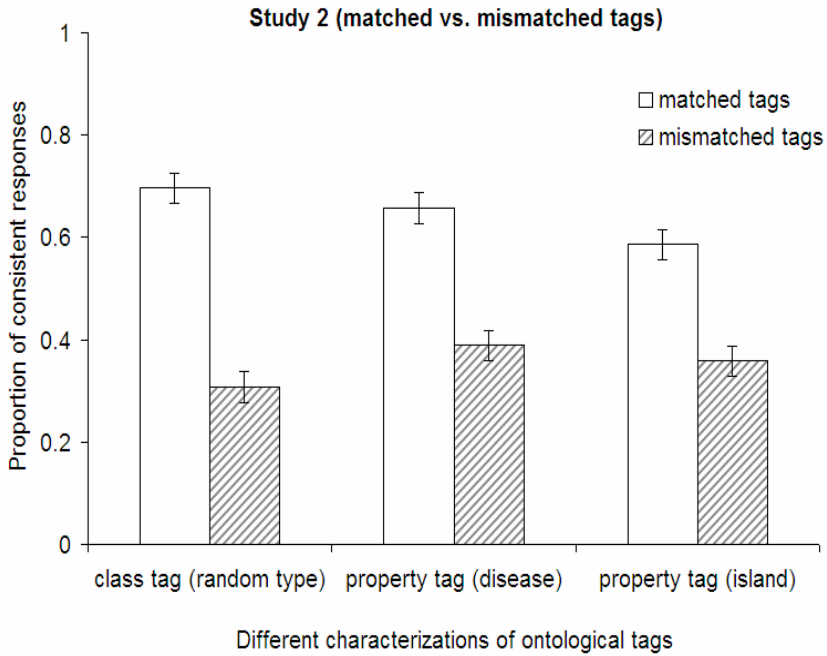


Fig. 6. A summary of Study 2. The error bars represent standard errors obtained in each condition.

Table 3. Main results from Study 2

	class tag (random type)	property tag (disease)	property tag (island)
effect size (r)	0.74	0.55	0.54
Fisher Z	0.95	0.62	0.61

Comparing Fisher Z in every pair (p-value)

	class tag (random type)	property tag (disease)	property tag (island)
class tag (random type)	-		
property tag (disease)	0.05	-	
property tag (island)	0.04	0.48	

As Fig. 6 shows, the impact of matched/mismatched labels was considerably larger when the labels were characterized with class inclusion information. When sample and test stimuli had the same tags, the proportion of making consistent responses was high. When sample and test stimuli had different tags, the proportion of making consistent responses declined sharply. This tendency decreased substantially when the same labels were characterized with property information.

Overall, the effect size obtained in the random-class tag condition was significantly larger than that in the property tag (island) condition;  $Z=1.70$ ,  $p<0.05$ . The effect size difference between the random-class tag condition and the property-tag (disease) condition was also substantial;  $Z=1.61$ ,  $p=0.053$  (Table 3), suggesting a sizable advantage for a random-class tag over the two property tags. Taken together, these results confirm that there is a strong tendency to use class-labels to make predictions.

## 4 General Discussion

We manipulated the meanings associated with arbitrary labels attached to imaginary insects, and examined how the manipulation would influence the way the subjects predict the attributes of these insects. In both Study 1 and Study 2, the college students displayed a strong tendency to use class inclusion labels to make their inferential judgments. This tendency was reduced significantly when the verbal tags were replaced with pictorial signs, or when the tags represented various kinds of property information, such as the disease, location, or the shape of wings that these imaginary insects carry, live, or possess.

Why does human reasoning heavily rely on verbal labels representing class inclusion information? One source of this bias seems to come from the need to maintain *cognitive economy* [5]. One of the most compelling findings in the cognitive psychology of inductive inference is that people make a predictive inference on the basis of the categorical information that is immediately recognizable [6]-[10]. By arranging concepts categorically, we treat individual objects as a group and deal with the characteristics of the group as a whole, rather than individual objects separately. "Grouped" representation can expedite many cognitive tasks, and help overcome the processing limitations of the human brain [11]. As a result, we often gloss over individual differences and draw erroneous conclusions about individuals (e.g., stereotyping).

Another important source of the over-reliance on class inclusion labels is likely to stem from the influence of language. Category labels generally correspond to count nouns (e.g., animals, dogs, cats, or apples). They are subject to linguistic constraints to a larger degree than adjectives, because count nouns vastly outnumber adjectives in linguistic communications. This may be an important reason why noun labels are fundamentally different from attribute labels and affect our inductive inference in a significant manner [12]. For example, noun labels can be used as a metaphor ("my job is a *jail*"), or to help activate a particular aspect of a concept ("the *pen* is mightier than the *sword*"). When noun labels are used metaphorically ("my job is a *jail*"), the label *jail* does not represent a collection of individual instances of "jail." Rather, the label accentuates one aspect of the concept (e.g., being captive). In this manner, the meaning associated with noun labels is determined in the context of communication, and categorical noun labels can flexibly influence our inferential behavior.

*Implications for Semantic Web research.* What does the current finding tell us in terms of the development of the Semantic Web? There are several important implications. First, the developers and researchers in the Semantic Web community may be better off by focusing on the areas in which human reasoning is particularly vulnerable. Our strong inclination to use class inclusion labels comes from the need to process information quickly in light of our processing constraints. This bias occurs particularly when processing information is mentally cumbersome. Unlike computers, the human brain is far limited in its processing capacity. When reasoning involves many options and conditions to weigh, we rely on simple heuristics, such as the recognizability and representativeness of a selection.

The application of the Semantic Web can be especially important in fields such as product selection, medical diagnosis, and human resources management. For example, in selecting a home insurance policy, we need to process a large amount of regulations, conditions, criteria and so on. Insurance companies often deliberately fill their policies with many unrecognizable options, making it difficult for lay people to grasp necessary information. Selecting appropriate drugs also require extensive processing of the information related to their ingredients, possible side effects, and effectiveness. The Semantic Web can help people make rational decisions by helping reduce processing overload. A similar application can be developed in many areas of product selection, such as the selection of infant foods, automobiles, schools, and houses. Clearly, machines appear to do much better than humans in these areas.

Another potentially promising area of Semantic Web application can be medical diagnosis. According to recent statistics, 1 in every 47 diabetic patients (5.7 million US citizens) were misdiagnosed.<sup>1</sup> Misdiagnosis occurs primarily due to the failure to take a variety of patient information into account, such as his/her family history, lifestyle, age and racial marker. With the advent of genome research, individuals' genomic information will also become available to clinicians in the future, and doctors will be increasingly fraught with individuated information. By developing inference agents tailored for specific patents, medical diagnosis can be facilitated significantly by using the Semantic Web technology [13] [14].

Human resources management can be another promising area. Selecting appropriate employees, and providing sound services to individual employees (e.g., health insurance and retirement benefits, and employee training) requires a vast amount of individuated information. This process can be facilitated by utilizing semantic web agents that are appropriately geared to analyze the information about individual employees.

Finally, the present study indicates that OWL researchers and developers may need to consider vocabularies that are aligned with linguistic classes, such as count nouns, adjectives and verbs. Likewise, algorithms for ontology matching may also need to integrate these linguistic classes. Many agents in ontology matching measure the semantic similarity between ontologies by a weighted sum of individuated "similarity factors" [15] [16] [17]. These linguistic classes can be a dominant factor in determining the degree of matching between ontologies. Because human knowledge

---

<sup>1</sup> This statistics is taken from The National Women's Health Information Center, U. S. Department of Health and Human Services and reported at <http://www.wrongdiagnosis.com/intro/notdiagcommon.htm>.

is primarily expressed by language, our conceptual activities (e.g., inferential reasoning) are also subject to the structure of language [18] [19]. Researchers in the Semantic Web community may need to pay attention to this relationship.

**Acknowledgements.** This research was supported by the Glasscock Center Faculty Fellow Award, and a Developmental Grant by the Mexican American and U.S. Latino Research Center, Texas A&M University. I would like to thank Art Markman, Na-Yung Yu and Wookyoung Jung for their valuable comments.

## References

1. Walton, C.D.: Agency and the Semantic Web. Oxford University Press, Oxford (2007)
2. Murphy, G.L.: The Big Book of Concepts. MIT Press, Cambridge (2002)
3. Yamauchi, T., Kohn, N., Yu, N.: Tracking mouse movement in feature inference: Category labels are different from feature labels. *Mem Cognition* 35, 852–863 (2007)
4. Rosenthal, R.: *Meta-analytic Procedures for Social Research*. Sage Publications, London (1984)
5. Rosch, E.: Principles of Categorization. In: Rosch, E., Lloyd, B.B. (eds.) *Cognition and Categorization*, pp. 27–48. Lawrence Erlbaum, Hillsdale, NJ (1978)
6. Lagnado, D.A., Shanks, D.R.: The Influence of Hierarchy on Probability Judgment. *Cognition* 89, 157–178 (2003)
7. Murphy, G.L., Ross, B.H.: Predictions from Uncertain Categorizations. *Cognitive Psychol.* 27, 148–193 (1994)
8. Ross, B.H., Murphy, G.L.: Category-based Predictions: Influence of Uncertainty and Feature Associations. *J. Exp. Psychol. Learn* 22, 753–763 (1996)
9. Yamauchi, T., Markman, A.B.: Inference Using Categories. *J. Exp. Psychol. Learn* 26, 776–795 (2000)
10. Yamauchi, T.: Labeling Bias and Categorical Induction: Generative Aspects of Category Information. *J. Exp. Psychol. Learn* 31, 553–568 (2005)
11. Gigerenzer, G., Todd, P.M.: *Simple Heuristics That Make Us Smart*. Oxford University Press, New York (1999)
12. Langacker, R.W.: Conceptualization, Symbolization, and Grammar. In: Tomasello, M. (ed.) *The New Psychology of Language: Cognitive and Functional Approach to Language Structure*, pp. 1–39. Lawrence Erlbaum, Mahwah (1998)
13. Shankar, R.D., Martin, S.B., O'Connor, M.J., Parish, D.B., Das, A.K.: Towards Semantic Interoperability in a Clinical Trials Management System. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006. LNCS*, vol. 4273, pp. 901–912. Springer, Heidelberg (2006)
14. Sheth, A., Agrawal, S., Lathem, J., Oldham, N., Wingate, H., Yadav, P., Gallagher, K.: Active Semantic Electronic Medical Record. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006. LNCS*, vol. 4273, pp. 913–926. Springer, Heidelberg (2006)
15. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. In: Spaccapietra, S. (ed.) *Journal on Data Semantics IV. LNCS*, vol. 3730, pp. 146–171. Springer, Heidelberg (2005)
16. Euzenat, J., Loup, D., Touzani, M., Valtchev, P.: Ontology alignment with OLA. In: *Proceedings of the 3rd International Workshop on the Evaluation of Ontology-based Tools 2004* (2004)

17. Doan, A., Madhavan, J., Domingos, P., Halevy, A.: Ontology matching: A machine learning approach. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*, pp. 385–403. Springer, Heidelberg (2004)
18. Genter, D., Goldin-Meadow, S.: Whiter Whorf. In: Genter, D., Goldin-Meadow, S. (eds.) *Language in Mind: Advances in the Study of Language and Thought*, pp. 3–14. MIT Press, Cambridge (2003)
19. Yamauchi, T.: *Linking Syntax and Inductive Reasoning: Categorical Labeling and Generic Noun Phrases*. *Psychologia* (in press)

# From Web Directories to Ontologies: Natural Language Processing Challenges\*

Ilya Zaihrayeu<sup>1</sup>, Lei Sun<sup>2</sup>, Fausto Giunchiglia<sup>1</sup>, Wei Pan<sup>2</sup>, Qi Ju<sup>3,1</sup>,  
Mingmin Chi<sup>2</sup>, and Xuanjing Huang<sup>2</sup>

<sup>1</sup> University of Trento, Italy

{ilya,fausto}@dit.unitn.it

<sup>2</sup> Fudan University, China

{leisun,panwei,mmchi,xjhuang}@fudan.edu.cn

<sup>3</sup> Jilin University, China

qi@jlu.edu.cn

**Abstract.** Hierarchical classifications are used pervasively by humans as a means to organize their data and knowledge about the world. One of their main advantages is that natural language labels, used to describe their contents, are easily understood by human users. However, at the same time, this is also one of their main disadvantages as these same labels are ambiguous and very hard to be reasoned about by software agents. This fact creates an insuperable hindrance for classifications to being embedded in the Semantic Web infrastructure. This paper presents an approach to converting classifications into lightweight ontologies, and it makes the following contributions: (i) it identifies the main NLP problems related to the conversion process and shows how they are different from the classical problems of NLP; (ii) it proposes heuristic solutions to these problems, which are especially effective in this domain; and (iii) it evaluates the proposed solutions by testing them on DMOZ data.

## 1 Introduction

The success of the Web was particularly conditioned by the ease with which its users could publish and interlink their data. However, as the Web has grown larger, it has become essential to categorize the huge amounts of documents on the web. Hierarchical classifications, whose nodes are assigned natural language labels, perfectly serve this purpose. In fact, there are plenty of classifications on the web: web directories like DMOZ<sup>1</sup>, business catalogues like Amazon<sup>2</sup>, topic categories like Wikipedia<sup>3</sup>, site maps in web portals and in personal pages are examples that demonstrate the pervasive presence of classifications on the web.

---

\* This work has been partly supported by the OpenKnowledge project (FP6-027253, see <http://www.openk.org>) and by Natural Science Foundation of China under grant No. 60673038.

<sup>1</sup> See <http://www.dmoz.org>

<sup>2</sup> See <http://www.amazon.com>

<sup>3</sup> See <http://www.wikipedia.org>

The underlying idea of the Semantic Web is that web content should be expressed not only in natural language, but also in a language that can be unambiguously understood, interpreted and used by software agents, thus permitting them to find, share and integrate information more easily [3]. The central notion to this idea is *ontology*, which defines a taxonomy of classes of objects and relations among them [3]. Differently from classifications, ontologies should be written in a formal language such as OWL [16], which is unambiguous and suitable for being reasoned about by software agents.

Ontologies are very hard to be designed by an ordinary user of the Web, and designing an OWL-DL [16] ontology is a difficult and error-prone task even for experienced users [22]. This fact further complicates a classic chicken-and-egg problem which prevents the Semantic Web from scaling in the large: users will not mark up their data unless they perceive an added value from doing so, and tools to demonstrate this value will not be developed unless a “critical mass” of annotated data is achieved [12]. As Hendler further remarks in [12], “*Lowering the cost of markup isn’t enough – for many users it needs to be free. That is, semantic markup should be a by-product of normal computer use*”.

On the other hand, classifications are very easy to be created and maintained by an ordinary user. They represent a very natural way for (natural language) markup of the data classified in them. Moreover, classifications are used pervasively on the web thus creating the necessary “critical mass” of annotated data. These facts seem to resolve the chicken-and-egg problem. However, because they are described in natural language, classifications cannot be easily embedded in the infrastructure of the Semantic Web. To address this problem, [9] discusses how classifications can be scaled up to the Semantic Web by converting them into lightweight ontologies, and [11] demonstrates the practical applicability of the approach in its application to automatic ontology-based document classification.

The current paper extends the work presented in [9,11] by analyzing in detail the principle step of conversion from natural language to formal language. The main natural language processing (NLP) problems related to the conversion process are: named entity (NE) locating, part-of-speech (POS) tagging, word sense disambiguation (WSD), and parsing. We show how these problems, when applied to the classification domain, are different from their classical application on full-fledged sentences. We propose heuristic solutions to the NE locating, POS tagging, and WSD problems, and evaluate their performance by testing them on DMoz data. As we show in the paper, NE locating is a much easier problem in the DMoz data set, where we reach 93.45% of precision; in a POS tagging task we reach 96.00% of precision which is 11.52% higher than in the application of the POS tagger trained on full-fledged sentences; and, in the WSD task we reach 66.51% of accuracy which is an acceptable performance result according to the state-of-the-art in this field of NLP.

The paper is organized as follows. In Section 2 we discuss how we convert classifications into lightweight ontologies and show how the above mentioned NLP problems are relevant to this conversion process. Sections 3, 4, and 5 discuss particular problems of, proposed solutions and evaluation results for NE locating,



POS tagging, and WSD respectively. In Section 6 we discuss the related work. Section 7 summarizes the results and concludes the paper.

## 2 From Classifications to Lightweight Ontologies

Classification labels are expressed in natural language, which is ambiguous and very hard to be reasoned about. In order to address this problem, we encode classification labels into formulas in propositional Description Logic language  $L^C$ , following the approach described in [9]. Note that even if  $L^C$  is propositional in nature, it has a set-theoretic semantics. Namely, the interpretation of a (lexically expressed) concept is the set of documents, which are about this concept [9]. For instance, the interpretation of concept **Capital** (defined as “a seat of government”) is the set of documents about capitals, and *not* the set of capitals which exist in the world. Below we briefly describe how we encode classification labels into formulas in  $L^C$ . Interested readers are referred to [9] for a complete account. Here, we discuss the conversion process in a limited extent while focusing on the related NLP problems.

WordNet [17] senses of adjectives and common nouns become atomic concepts. The extension of a common noun concept is the set of documents about objects of the class, denoted by the noun; and, the extension of an adjective concept is the set of documents about objects, which possess the qualities, denoted by the adjective. Proper names (also recognized as named entities) become atomic concepts as well, whose extension is the set of documents about the individual referenced by the proper name. Notationally, we construct adjective and common noun atomic concepts using the following syntax: *lemma-pos-sn*, where *lemma* is the lemma of the word, *pos* is its part of speech, and *sn* is the sense number in WordNet [17]. We use  $_{NNP}$  to mark proper name atomic concepts.

Atomic concepts are then connected to form complex concepts as follows: syntactic relations between words are translated to logical connectives of  $L^C$ . For example, a set of adjectives followed by a noun group is translated into the logical conjunction ( $\sqcap$ ) of the concepts corresponding to the adjectives and to the nouns; prepositions like “of” and “in” are translated into the conjunction; coordinating conjunctions “and” and “or” are translated into the logical disjunction ( $\sqcup$ ). The final formula for a label is built following these rules and taking into account how words are coordinated in the label. The final formulas are then assigned to classification nodes, thus converting the classification into a lightweight ontology. These ontologies can be used for automating various tasks on classifications, such as semantic search [9], semantic matching [10], and document classification [11].

Let us consider a relatively complex label: “*Bank and personal details of George Bush*”. Its correct translation to  $L^C$  will produce the following concept:

$$(\text{bank-noun-1} \sqcup \text{personal-adj-1}) \sqcap \text{detail-noun-1} \sqcap \text{george\_bush}_{NNP}$$

The extension of the concept above is the intersection of three sets of documents: (i) documents about the President George W. Bush, (ii) documents containing isolated facts about something (i.e., details), and (iii) the union of documents

about bank institutions and documents concerning a particular person or his/her private life. As it can be seen, the extension includes documents one would classify under a node with the above given natural language label.

Despite its seeming simplicity, the translation process is subject to various mistakes originating from incorrect NLP. For instance, due to a mistake in POS tagging, the word *personal* might be recognized as a noun, which has only one sense in WordNet defined as “a short newspaper article about a particular person or group”; due to a mistake in WSD, the word *bank* might be identified as “sloping land (especially the slope beside a body of water)”; due to a mistake in NE locating, the proper name *George Bush* might not be recognized and might then be considered as two distinct nouns, where the noun *bush* means “a low woody perennial plant usually having several major branches”; finally, due to a mistake in (syntax) parsing, the input label might be translated into:

bank-noun-1  $\sqcup$  personal-adj-1  $\sqcap$  detail-noun-1  $\sqcap$  george\_bush<sub>NP</sub>

a concept, whose extension is the union of documents about bank institutions and documents discussing personal details of the President George W. Bush.

The NLP framework, which enables the conversion of classification labels into formulas in  $L^C$ , is depicted in Fig. 1. It follows the standard NLP pipeline: tokenization, NE locating, POS tagging, WSD, and parsing. In our framework we assume that a label is either an NE or a non-NE. Therefore, NE labels skip the last three steps since they already represent atomic concepts.

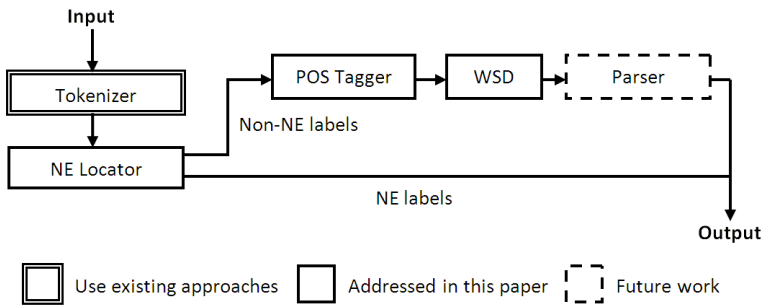


Fig. 1. Framework of NLP on a Web directory

Tools developed for general NLP tasks cannot be directly used in our framework. One main reason is that the corpus we use, namely, labels in Web directories, is significantly different from those on which most NLP tools are developed. These differences are briefly described as follows:

- Web directory labels are short phrases, while general NLP tools are developed on full-fledged sentences;
- Most of the words in a Web directory are nouns, adjectives, articles, conjunctions and prepositions. The verbs and pronouns are very rare in a Web directory while being common in full-fledged sentences;

- NEs occur densely in a Web directory. This is not surprising, as a Web directory is a knowledge base, which unavoidably has many proper nouns that describe entities in the world;
- The capital rule is different in a Web directory. In full-fledged sentences, the first words of sentences and the words in proper names are initialized with capital letters. In a Web directory, however, most often every word begins with a capital letter except for prepositions and conjunctions;
- The proper sense of a word may depend on the meaning of a word appearing in a label located higher in the classification tree. For instance, noun “Java” means an island if it appears under a node with label “Geography”.

In this paper, we focus on NE locating, POS tagging and WSD on a Web directory. We perform tokenization following the standard approach from Penn Treebank [18], and we leave parsing to the future work as how to do it strongly depends on the results presented in this paper.

### 3 Named Entity Locating

The data set we used for our analysis and evaluation (of the NE locator and the POS taggers) is built on DMoz. According to a dump created in 2004, the DMoz directory has 698,057 labels (nodes). There are many non-English labels, which can be excluded from consideration by discarding the DMoz subtrees rooted at the following nodes: `Top/World`, `Top/Kids_and_Teens/International`, and `Top/Adult/World`. As the result, we have 474,389 English labels. For these English labels, the average length (i.e., the number of tokens) is 1.91 and the average depth (i.e., the number of hops from the root) is 7.01.

Out of 474,389 labels we randomly selected 12,365 labels (2.61%) for analysis and manual annotation. Each label in this data set has been annotated with POS and NE information. As the result, we have totally 8177 non-NE labels (66.13%) and 4188 NE labels (33.87%). We observed that nearly all NEs take *entire* labels. We manually examined the data set and found only 7 exceptional labels (0.06%). Therefore, the assumption made in our NLP framework is valid. In Table 1 we report statistics of POS occurrences in the non-NE labels in our data set.

**Table 1.** Statistics of POS occurrences in the data set

POS	NP	NN	NNS	CC	JJ	IN	,	TO	other
Occurrence	7952	7784	3623	2893	1020	235	72	18	39
Percentage	33.64	32.93	15.33	12.24	4.32	1.00	0.30	0.07	0.17

#### 3.1 The Approach

By analyzing the data set, we noticed the following characteristics of NEs:

- Rare labels tend to be NEs. A general label (such as `Arts and Entertainment`) can occur thousands of times in a Web directory, while NE labels

occur much more rarely. Most of NE labels, such as “Schindler’s List” (a movie name) occur only once;

- Labels in which most of the tokens are rare words tend to be NEs, e.g., *Agios Dometios* is a geography name and each of its tokens occurs only once in the whole directory;
- There are so-called letter bars in Web directories, such as single letter “A”, “B”, ..., “Z” and also double letters “Aa”, “Ab”, ..., “Zz”. These labels are created only for the convenience of navigation. Besides, they are good indicators of NEs, as nearly all children of these labels are NEs;
- In an NE label, initial articles, such as “the”, “a” and “an”, are usually put at the end after a comma. For example, “The Magic School Bus” is written as “Magic School Bus, The”. This is another good indicator of NEs;
- The NE and non-NE labels distribute differently on their lengths. This difference is illustrated in Fig. 2(a), which is the statistical result of label length in our data set;
- The NE and non-NE labels distribute differently on their depths. This difference is illustrated in Fig. 2(b), which is the statistical result of label depth in our data set.

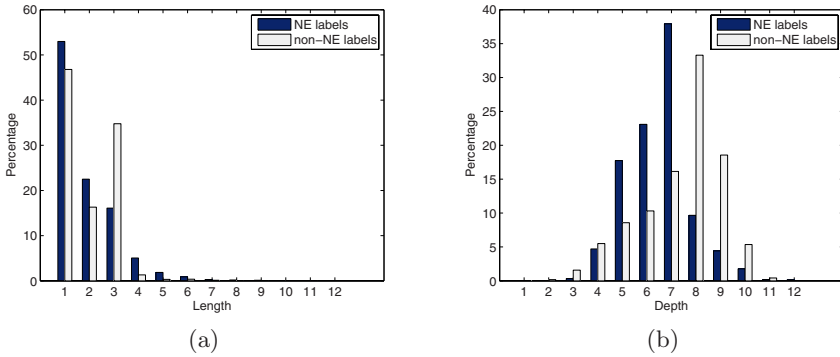


Fig. 2. (a) Label length distribution; (b) Label depth distribution

Taking these characteristics into account, we implemented the NE locator using Conditional Maximum Entropy Model (CMEM) [2] with Gaussian smoothing [6]. The features for CMEM have been chosen according to the characteristics described above. Particularly, we consider the following feature classes (i.e., sets of features) in the implementation:

- *WordsInLabel*: The first two and the last two tokens in the label;
- *WordsInPath*: The first and the last tokens in the label’s parent, grandparent, the farthest ancestor (excluding the root “Top”) and the second farthest ancestor;
- *LengthOfLabel*: The number of tokens in the label;

- *DepthOfLabel*: Depth of the label (distance from the root node);
- *FrequencyOfLabel*: Count how many times the label occurs in the whole directory;
- *AveFrequencyOfTokens*: Count how many times each token in the label occurs in the whole directory, and calculate the average.

### 3.2 Evaluation

This experiment is performed in two steps. First, we train the NE locator by using each feature class to compare their contributions. Then, we train the NE locator again with some combinations of feature classes to see the best performance we can reach. To make our experimental results more reliable, we perform 6-fold cross validation. We use the following 3 measures to evaluate the performance of the NE locator:

- **Precision of NE locating (PNE)**. We count how many labels picked out by the NE locator are real NE labels (those annotated as NEs in the data set), and calculate the percentage;
- **Recall of NE locating (RNE)**. We count how many real NE labels are picked out by the NE locator, and calculate the percentage;
- **F-score of NE locating (FNE)**. An overall measure of performance of the NE locator, which combines PNE and RNE as:

$$\text{FNE} = \frac{2 \cdot \text{PNE} \cdot \text{RNE}}{\text{PNE} + \text{RNE}}$$

We report the performance results of the NE locator in Table 2. As it can be observed, feature classes *WordsInLabel* and *WordsInPath* provide the most important contributions to the precision. By combining these two feature classes we can get the performance which is close to that provided by combining all the feature classes (compare the figures reported in row “1+2” with ones in row “1+2+3+4+5+6”).

**Table 2.** Performance results of the NE Locator

Feature Class	PNE	RNE	FNE
1. <i>WordsInLabel</i>	81.49	94.33	87.45
2. <i>WordsInPath</i>	89.48	79.36	84.12
3. <i>FrequencyOfLabel</i>	75.04	91.30	82.37
4. <i>AveFrequencyOfTokens</i>	76.05	82.95	79.35
5. <i>DepthOfLabel</i>	53.13	78.76	63.45
6. <i>LengthOfLabel</i>	64.64	8.05	14.32
1+2	92.08	94.20	93.13
1+2+3+4+5+6	93.45	94.04	93.75

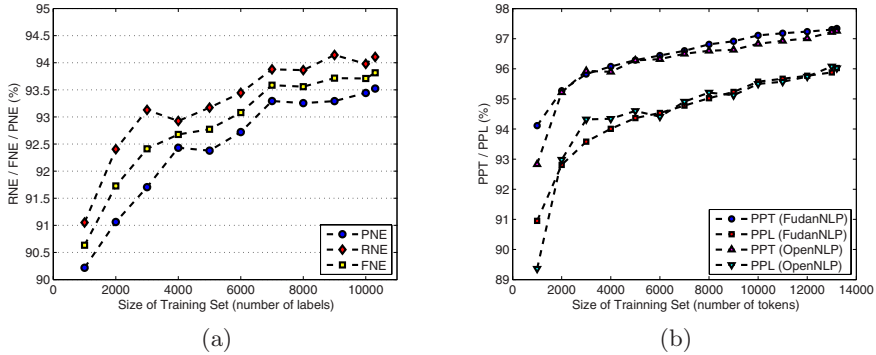


Fig. 3. Incremental training of: (a) NE Locator; (b) POS Taggers

One state-of-the-art system [8] of NE locating in the Web environment on full-fledged sentences has the performance of 59% in precision, 66% in recall and 38% in F-score. Similar to our task, NE locating in the Web environment share the difficulty of large amount of undefined entity classes. The reason our approach outperforms theirs is that NE locating on Web directories is a relatively easy task, as we only need to tell whether a label is an NE or not.

To check whether our data set is properly sized, we performed incremental training, namely, keeping the testing set unchanged, we checked how performance varied with the growing size of the training set. In Fig. 3(a) we show the achieved results. As it can be observed, PNE, RNE, and FNE increase significantly when the size of the training set grows from 1000 to 7000 samples. When the number of samples becomes greater than 7000, the performance measures change slightly. Empirically, we conclude that our NE locating model is effective and stable enough to be used on web directories such as DMoz.

## 4 Part of Speech Tagging

### 4.1 The Approach

Nearly all state-of-the-art POS taggers are based on supervised learning approaches. In these approaches, first, a properly sized manually annotated corpus is created. Then, a model is trained on this corpus to allow for further POS tagging of unseen data. Popular models for POS tagging include Hidden Markov Model (HMM) [21], Conditional Maximum Entropy Model (CMEM) [2], and Conditional Random Field (CRF) [15]. Below we briefly describe CMEM and CRF, as they are used by the POS taggers we employ in our experiments.

To tag a token, CMEM considers the context of the token by introducing the notion of feature. In the task of POS tagging, the context of a token is usually the token itself and its surroundings. A feature is a function which maps the context of the token to a 0-1 value. Namely, it answers a yes/no question about the

context. CMEM learns how POS is conditioned by contexts as a set of probability distributions from a manually annotated corpus. The learning process applies a max-entropy style parameter optimization. CMEM tags the tokens sequentially (starting from the left-most token in the sentence) by assigning the POS with the highest conditional probability to each token given the token's context.

Differently from CMEM, in CRF, the POS of a token is conditioned by contexts of *all* the tokens in the given sentence. This allows for a global coordination among local taggings. This property makes CRF a more advanced model for the POS tagging task.

In our experiments, we employed two POS taggers: the CRF-based FudanNLP POS tagger [20] and the CMEM-based OpenNLP POS tagger [19]. We retrained these tools on our data set and checked if we gain an improvement in accuracy w.r.t. the case when the tools are trained on full-fledged sentences. To avoid a negative influence of NE labels on the training of a POS tagger, both POS taggers were trained and tested only on the non-NE labels in the data set.

## 4.2 Evaluation

To make our experimental results more reliable, we perform 6-fold cross validation. The following 2 measures are used to evaluate the performance of the POS taggers:

- **Precision of POS tagger by Tokens (PPT)**. The granularity of this precision measure is a token, namely, we count tokens which are tagged with the correct tag, and calculate the percentage;
- **Precision of POS tagger by Labels (PPL)**. The granularity of this precision measure is a label, namely, we count labels whose tokens are all correctly tagged, and calculate the percentage.

The evaluation results are shown in Table 3, where the following notations are used:  $PPT_0$  and  $PPL_0$  refer to the PPT and PPL before retraining, while  $PPT_1$  and  $PPL_1$  refer to the PPT and PPL after retraining. Note that there is no significant difference in the performance between the CMEM approach (OpenNLP) and the CRF approach (FudanNLP). It has been proven that CRF outperforms CMEM when tagging full-fledged sentences, since CRF considers global coordination in a sentence while CMEM only considers local context. However, in DMOZ, labels (symmetric to sentences) are too short (1.91 tokens on the average). In most cases, CMEM features of a single token are able to consider information of the whole label. In other words, CMEM is able to do something like global coordination as CRF on these short labels. This property of our data set makes CRF similar to CMEM in performance.

The state-of-the-art performance of a POS tagger on full-fledged sentences is 97.24% in token precision (PPT) according to [24], which is very close to ours. However, precision by sentences should be lower than our PPL, as our labels are much shorter.

We performed incremental training of the POS taggers, too. The result is shown in Fig. 3(b), which demonstrates a trend similar to that in Fig. 3(a).

**Table 3.** Performance results of the OpenNLP and FudanNLP POS taggers before and after retraining

	PPT <sub>0</sub>	PPT <sub>1</sub>	Gain	PPL <sub>0</sub>	PPL <sub>1</sub>	Gain
OpenNLP	91.27	97.23	+6.16	84.68	96.00	+11.52
FudanNLP	96.12	97.33	+1.21	92.72	96.02	+3.30

Empirically, we conclude that our POS tagging model is effective and stable enough to be used on web directories such as DMoz.

## 5 Word Sense Disambiguation

### 5.1 The Approach

The proposed WSD algorithm traverses the nodes of the classification tree in the BFS or DFS order. Then, at each node, it first finds *concept tokens*, i.e., tokens which are present in WordNet as adjectives and/or as nouns. Next, it identifies *ambiguous concept tokens*, i.e., concept tokens which have more than one sense. Ambiguous concept tokens of each node are processed by the algorithm following the steps reported below. If a token is not disambiguated at step  $n$ , then it is processed at step  $n + 1$ . The ultimate goal of the algorithm is to select only one sense for each ambiguous concept token. Below we say that a token sense is *active* if it has not been discarded.

1. Identify the POS of the token and, if the token has senses of this POS, then preserve these senses and discard senses belonging to the other POS, if any;
2. Preserve noun token senses if they are hypernyms or hyponyms of active noun senses of other concept tokens in the label, and discard the other senses. Hypernymy and hyponymy relations amongs noun token senses are checked using the WordNet hypernymy hierarchy [17];
3. Preserve noun token senses if they are located within a certain distance in the WordNet hypernymy hierarchy from active noun senses of other concept tokens in the label. If there are several matching senses with different distances, then preserve those with the shortest distance and discard the others;
4. Preserve noun token senses if they are hyponyms of active noun senses of concept tokens appearing in the label of an ancestor node, and discard the other senses. Note that we do not consider hypernyms since, as reported in [11], higher level nodes usually represent more general categories than lower level nodes;
5. Preserve noun token senses if they are located within a certain distance in the WordNet hypernymy hierarchy from active noun senses of concept tokens appearing in the labels of ancestor nodes, and discard the other senses. If there are several matching senses with different distances, then preserve those with the shortest distance and discard the others;



6. Preserve the first active noun sense (in WordNet) and discard the other active senses. If there is no active noun sense, then preserve the first active adjective sense and discard the other active senses. Noun senses prevail over adjective senses since, according to the results reported in Table 1, nouns are much more frequent in a web directory than adjectives. Note that senses in WordNet are sorted in the descendant order of the number of times they were tagged in the semantic concordance texts [17]. Therefore, picking up a sense that appears higher in the list of senses increases the probability of that the sense will be the correct meaning of the token. After this step, the token is disambiguated.

## 5.2 Evaluation

To evaluate the performance of our WSD algorithm, we have selected a DMoz subtree rooted at `Top/Business/Consumer_Goods_and_Services`. The subtree has 781 nodes, its maximal depth is 6, and the average branching factor is 4.22. Its nodes have 1368 tokens in total. There are 1107 concept tokens, out of which 845 are ambiguous. The average polysemy of an ambiguous concept token is 4.05. Note that this data set is different from that used in NE locating and POS tagging because the WSD algorithm requires a POS tagged subtree and not a set of randomly selected labels. The two data sets do not have nodes in common, which ensures unbiased performance of the POS tagger at step 1 of the algorithm.

In Table 4 we report the results of the algorithm measured in 8 experiments. For each step we provide the number of disambiguated tokens and the accuracy of the step. For steps 3 and 5 we provide the similarity distance threshold as the number of edges in the WordNet hypernymy hierarchy. We write “na” (not applicable) as a parameter and an output value of a step when the step was skipped in the experiment. In the right-most column we report the overall accuracy. At step 1 of the algorithm, we used the FudanNLP POS tagger, which was beforehand trained on the whole data set described in Section 3. At steps 2-5 we used WordNet version 2.1.

**Table 4.** Performance results of the WSD algorithm

#	Step 1		Step 2		Step 3			Step 4		Step 5			Step 6		Accur.
	tok.	acc.	tok.	acc.	thr.	tok.	acc.	tok.	acc.	thr.	tok.	acc.	tok.	acc.	
1	na	na	na	na	na	na	na	na	na	na	na	na	845	63.90	63.90
2	84	98.81	na	na	na	na	na	na	na	na	na	na	761	60.84	64.62
3	84	98.81	11	100	na	na	na	na	na	na	na	na	750	61.60	65.80
4	84	98.81	11	100	10	250	50.80	na	na	na	na	na	500	61.20	62.37
5	84	98.81	11	100	2	24	87.50	na	na	na	na	na	726	60.88	65.92
6	84	98.81	11	100	2	24	87.50	8	87.50	na	na	na	718	61.28	66.51
7	84	98.81	11	100	2	24	87.50	8	87.50	10	379	33.24	339	42.77	46.51
8	84	98.81	11	100	2	24	87.50	8	87.50	2	43	41.86	675	60.00	64.49

The baseline solution, i.e., when only step 6 is executed, gives 63.90% accuracy. Step 1 performed reasonably well, correctly disambiguating 98.81% of about 10% of tokens. Step 2 disambiguated a small number of tokens (11) but all of them were disambiguated correctly. Step 3 performed reasonably well on small thresholds producing the best result when the threshold value was 2 (compare step 3 in experiments 4 and 5). A similar trend can be observed for step 5. However, even with the threshold value of 2, its accuracy is lower than the baseline accuracy, thus making a negative effect on the overall accuracy. Note that when the threshold value is 2, steps 3 and 5 preserve senses which are siblings in the WordNet hypernymy hierarchy. The best performance of the algorithm was recorded in experiment 6 with the accuracy value of 66.51%, which is 2.61% higher than the baseline.

The performance of a state-of-the-art WordNet-based WSD algorithm on full-fledged sentences varies according to the underlying approach. For instance, the best accuracy of the WSD algorithm presented in [11] is 47.3% for polysemous nouns. Similar to our case, in [11] the best accuracy is only slightly higher than the baseline. A more recent work, [25], uses a web search engine (together with WordNet) for WSD and reaches 76.55% accuracy for polysemous nouns in the best case. While the average polysemy of nouns is close to ours (4.08), the size of the context window varied from 3 to 7 words that are known to WordNet, what is not possible to have in our case. Empirically, we conclude that the result of our WSD algorithm is comparable to the state-of-the-art in this field of NLP.

## 6 Related Work

It is now quite clear that there is an insuperable gap between the logic-based Semantic Web and its real-world users because it is nearly impossible for an ordinary user to learn (how to use) an unfamiliar formal language [5,14,12,23]. To address this problem, ontology authoring (e.g., see [4,23]), interfacing (e.g., see [7]), and querying (e.g., see [5]) tools that use natural language as an interface with the human user have been proposed. The approach to converting classifications into lightweight ontologies, described in this paper, shares the spirit of [12,14,5,4,7,23] and makes a step exactly in the direction of helping users to benefit from the Semantic Web without requiring them to go through the burdensome learning curve.

We differ from the above cited approaches in several respects. For instance, the approaches reported in [4,5,7] require an ontology at the backend against which the user could formulate her queries in natural language. To reduce both ambiguity and complexity of natural language, [4,5,23] use a controlled language for user queries, therefore requiring the user to know about the used subset of English, its semantics and grammar rules. In order to provide meaningful input, in [23] the user still needs to understand the general principles of first order logic. None of these requirements are made in our approach, in which the user does not need to have preloaded ontologies at the backend. Instead, the user creates an ontology seamlessly as a *by-product of normal computer*

*use* – by creating a classification. The complexity of language is lower and no controlled language is required – the user naturally uses noun phrases to describe classification nodes. Noteworthy, even if our approach is more lightweight, it still allows for automating various tasks on classifications, such as semantic search [9], semantic matching [10], and document classification [11].

The approach described in [13] allows it to convert a hierarchical classification into an OWL ontology by deriving OWL classes from classification labels and by arranging these classes into a hierarchy (based on the `rdfs:subClassOf` relation) following the classification structure. The approach is based on some application-dependent assumptions such as that one label represents one atomic concept, and that relations between labels can be defined as is-a relations in some context (e.g., concept “ice” is more specific than concept “non-alcoholic beverages” when considered in the context of procurement [13]). These assumptions do not hold in a general case and are not made in our approach. More importantly, in the current paper we provide a complete account of the NLP problems which need to be dealt with when converting classifications into ontologies. This problem is not addressed in [13] and can be seen as a preliminary step to their work.

## 7 Conclusions

The paper presents an approach to converting classifications into lightweight ontologies and discusses in detail the NLP problems related to this conversion process on the example of the DMoz web directory. The NLP analysis reported in this paper, to the best of our knowledge, is the first investigation of how NLP technology can be applied on classification labels and, more generally, on short natural language (noun) phrases. Noteworthy, even if the application domain we consider is different from the one on which NLP technology is usually applied, the results reported in this paper are comparable with (and, sometimes, exceeding) those reached by the state-of-the-art NLP tools.

## References

1. Agirre, E., Rigau, G.: A proposal for word sense disambiguation using conceptual distance. In: The First International Conference on Recent Advances in NLP, Tzigrav Chark, Bulgaria, September 1995 (1995)
2. Berger, A.L., Pietra, S.A.D., Pietra, V.J.D.: Maximum entropy approach to natural language processing. *Computational Linguistic* 22(1), 39–71 (1996)
3. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Scientific American* 284(5), 34–43 (2001)
4. Bernstein, A., Kaufmann, E.: GINO - a guided input natural language ontology editor. In: International Semantic Web Conference, pp. 144–157 (2006)
5. Bernstein, A., Kaufmann, E., Göhring, A., Kiefer, C.: Querying ontologies: A controlled English interface for end-users. In: International Semantic Web Conference, pp. 112–126 (2005)
6. Chen, S., Rosenfeld, R.: A Gaussian prior for smoothing maximum entropy models. Technical Report CMUCS -99-108, Carnegie Mellon University (1999)

7. Chong, W., Miao, X., Qi, Z., Yong, Y.: PANTO: A Portable Natural Language Interface to Ontologies. In: Proceedings of the European Semantic Web Conference, July 2007. LNCS, vol. 4519, Springer, Heidelberg (2007)
8. Downey, D., Broadhead, M., Etzioni, O.: Locating complex named entities in web text. In: Proc. of IJCAI, 2007 (2007)
9. Giunchiglia, F., Marchese, M., Zaihrayeu, I.: Encoding classifications into lightweight ontologies. In: JoDS VIII: Special Issue on Extended Papers from 2005 Conferences, A shorter version of the paper appeared in ESWC-2006 (2006)
10. Giunchiglia, F., Yatskevich, M., Shvaiko, P.: Semantic matching: Algorithms and implementation. Journal on Data Semantics IX (2007)
11. Giunchiglia, F., Zaihrayeu, I., Kharkevich, U.: Formalizing the get-specific document classification algorithm. In: 11th European Conference on Research and Advanced Technology for Digital Libraries, Budapest, Hungary, September 2007 (2007)
12. Hendler, J.: Agents and the semantic web. IEEE Intelligent Systems (2) (2001)
13. Hepp, M., de Bruijn, J.: GenTax: A generic methodology for deriving owl and RDF-S ontologies from hierarchical classifications, thesauri, and inconsistent taxonomies. In: 4th European Semantic Web Conference, Innsbruck, Austria, June 3-7 2007. LNCS 4519, pp. 129–144. Springer, Heidelberg (2007)
14. Katz, B., Lin, J.: Annotating the semantic web using natural language. In: NLPXML '02: Proceedings of the 2nd workshop on NLP and XML. Association for Computational Linguistics, Morristown, NJ, USA, pp. 1–8 (2002)
15. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. of 18th International Conf. on Machine Learning (2001)
16. McGuinness, D.L., van Harmelen, F.: OWL Web Ontology Language Overview. Technical report, World Wide Web Consortium (W3C) (February 10, 2004)
17. Miller, G.: WordNet: An electronic Lexical Database. MIT Press, Cambridge (1998)
18. Penn Treebank Project, See <http://www.cis.upenn.edu/~treebank/>
19. The OpenNLP project, See <http://opennlp.sourceforge.net/>
20. Qian, X.: A CRF-based pos tagger. Technical Report FDUCSE 07302, Fudan University (2007)
21. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. In: Proc. of the IEEE, vol. 77(2), pp. 257–285. IEEE Computer Society Press, Los Alamitos (1989)
22. Rector, A.L., Drummond, N., Horridge, M., Rogers, J., Knublauch, H., Stevens, R., Wang, H., Wroe, C.: OWL pizzas: Practical experience of teaching OWL-DL: Common errors & common patterns. In: EKAW, pp. 63–81 (2004)
23. Schwitter, R., Tilbrook, M.: Let's talk in Description Logic via controlled natural language. In: LENLS 2006. Logic and Engineering of Natural Language Semantics, Tokyo, Japan (2006)
24. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proc. of Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, vol. 1, pp. 173–180 (2003)
25. Yang, C., Hung, J.C.: Word sense determination using wordnet and sense co-occurrence. In: Proceedings of the 20th International Conference on Advanced Information Networking and Applications, Washington, DC, USA, vol. 1, pp. 779–784. IEEE Computer Society Press, Los Alamitos (2006)

# On the Foundations of Computing Deltas Between RDF Models

Dimitris Zeginis, Yannis Tzitzikas, and Vassilis Christophides

Computer Science Department, University of Crete, GREECE, and  
Institute of Computer Science, FORTH-ICS, Greece  
{zeginis,tzitzik,christop}@ics.forth.gr

**Abstract.** The ability to compute the differences that exist between two RDF models is an important step to cope with the evolving nature of the Semantic Web (SW). In particular, RDF Deltas can be employed to reduce the amount of data that need to be exchanged and managed over the network and hence build advanced SW synchronization and versioning services. By considering Deltas as sets of change operations, in this paper we study various RDF comparison functions in conjunction with the semantics of the underlying change operations and formally analyze their possible combinations in terms of correctness, minimality, semantic identity and redundancy properties.

## 1 Introduction

In order to cope with the evolving nature of the Semantic Web (SW) we need effective and efficient support for building advanced SW synchronization and versioning services. RDF Deltas, reporting the differences that exist between two RDF models have been proven to be crucial in order to reduce the amount of data that need to be exchanged and managed over the network in this respect [17][18][3][8].

Although RDF models can be serialized in various text formats (e.g., XML [1], N-Triples [2], Trix [3]), a straightforward application of existing version control systems for software code, such as RCS [29] and CVS [4], is not a viable solution for computing RDF Deltas. This is mainly due to the fact that RDF models, essentially represent graphs which (a) may feature several possible serializations (since there is no notion of edge ordering in [4]) and (b) are enriched with the semantics of RDFS specification (also including inferred edges according to [5]). For these reasons, several non text-based tools have been recently developed for comparing RDF graphs produced autonomously on the SW, as for example, SemVersion [31], PromptDiff [23], Ontoview [18], [10] and [3]. In most cases, the output of these tools is exploited by humans, and thus an intuitive presentation of the comparison results (and other related issues) has received considerable attention. SemVersion [31] proposes two Diff algorithms: (a) one *structure-based* which returns a set-based difference of the triples explicitly forming the two graphs, and (b) one *semantic-aware* which also takes into account the triples inferred by the associated RDFS schemas. PromptDiff [23][24][22] is an ontology-versioning environment,

<sup>1</sup> <http://www.w3.org/TR/rdf-syntax-grammar/>

<sup>2</sup> <http://www.w3.org/2001/sw/RDFCore/ntriples/>

<sup>3</sup> <http://www.w3.org/2004/03/trix/>

that includes a version-comparison algorithm (based on heuristic matchers [23,24]), while the visualization of the computed difference between two ontologies is discussed in [22]. Ontoview [18] is an ontology management system, able to compare two ontology versions and highlight their differences. Notably, it allows users to specify the conceptual relations (i.e. equivalence, subsumption) between the different versions of an ontology concept. Moreover, [10,13] introduce the notion of RDF molecules as the finest components to be used when comparing RDF graphs (in the absence of blank nodes each triple constitutes a molecule). Finally, tracking the evolution of ontologies when changes are preformed in more controlled environments (e.g. collaborative authoring tools) has been addressed in [19,25,32].

However, existing RDF comparison tools have not yet focused on the size of the produced Deltas, a very important aspect for building versioning services over SW repositories [28]. In this paper we are interested in computing RDF Deltas as sets of change operations (i.e. SW update programs) that enable us to transform one RDF model into the other. Consider, for example, the two RDF models  $K$  and  $K'$  of Figure 1 and their standard representation as sets of explicitly defined triples [5]: what set of change operations could transform  $K$  to  $K'$  ( $\Delta(K \rightarrow K')$ ) or vice versa ( $\Delta(K' \rightarrow K)$ )?

To answer this question we need to consider the semantics of the update primitives such as  $Add(t)$  and  $Del(t)$  where  $t$  is triple involving any RDF predicate. By assuming a side-effect free semantics for these primitives, i.e.  $Add(t)$  (resp.  $Del(t)$ ) is a straightforward addition (resp. deletion) of  $t$  from the set  $Triples(K)$ ,  $K'$  can be obtained by executing the following set  $\Delta_e$  ( $e$  stands for explicit) of change operations:

$$\Delta_e = \{Del(TA \text{ subclassOf } Person), Del(Address \text{ domain } Student), \\ Del(Jim \text{ type } Student), Add(TA \text{ subclassOf } Student), \\ Add(Address \text{ domain } Person), Add(Jim \text{ type } Person)\}$$

$\Delta_e$  is actually composed of update operations over the explicit triples of  $K$  and  $K'$ , and it is provided by the majority of existing RDF comparison tools [3,31,10]. However, by assuming side-effects (on the inferred triples not represented in Figure 1) during the execution of the above update primitives, we can reach  $K'$  by applying on  $K$  the following set  $\Delta_d$  ( $d$  stands for dense) of change operations:

$$\Delta_d = \{Del(Jim \text{ type } Student), Add(TA \text{ subclassOf } Student), \\ Add(Address \text{ domain } Person)\}$$

As we can easily observe,  $\Delta_d$  has only three change operations in contrast to  $\Delta_e$  that has six, given that inferred triples are also taken into account for the Delta computation. For example,  $Del(TA \text{ subclassOf } Person)$  is not included in  $\Delta_d$  because it can be inferred from  $K'$ . As we can see in Figure 1, this comparison function yields even smaller in size operation sets than the  $\Delta_c$  ( $c$  stands for closure) semantics-aware Delta of [31]. However,  $\Delta_d$  cannot always successfully transform one RDF model to another. Returning to our example of Figure 1,  $\Delta_d$  cannot be used to migrate backwards from  $K'$  to  $K$  since  $Del(Address \text{ domain } TA)$  is an operation not included in  $\Delta_d$ . For this reason, we need to consider additional RDF comparison functions involving inferred triples such as  $\Delta_{dc}$  ( $dc$  stands for dense & closure) illustrated in Figure 1. Still the resulting sets of operations have at most the same size as those returned by  $\Delta_c$ .

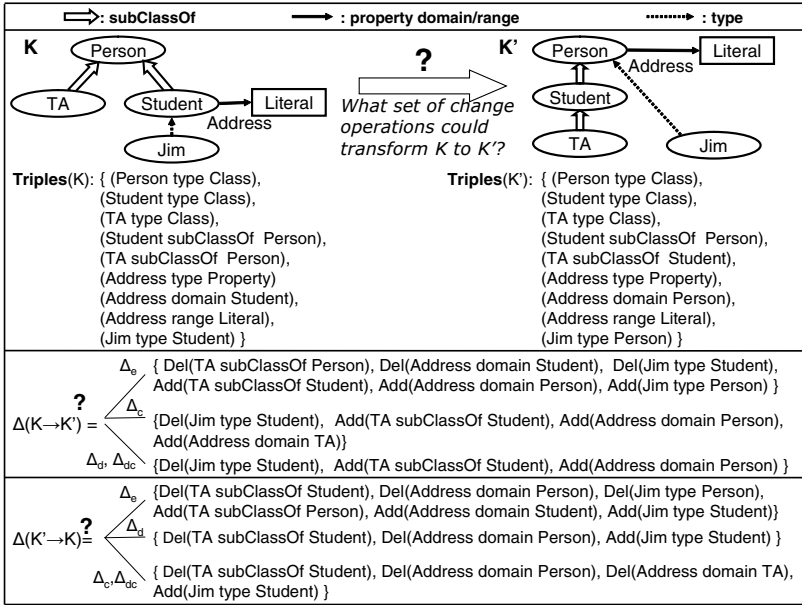


Fig. 1. Transforming K to K' and vice versa

RDF comparison functions that yield as less as possible change operations are quite beneficial for building SW versioning services. In particular, by advocating a change-based versioning framework [9] we can store in a SW repository only the update programs required to migrate (forward or backward) from one version to another rather than the entire set of triples for each version. In a nutshell, storing (or exchanging) as less as possible change operations is more space (or time) efficient. In this context, the main questions addressed by our work are: (a) what semantics of update primitives would make the above scenario possible (i.e. with what side-effects), and (b) how could we compute the corresponding set of change operations (i.e. with what comparison functions)? In response to these questions, the main contributions of this paper are:

- (a) We introduce two change operations semantics: one *plain* set-theoretic (considers only updates of the explicit triples) denoted by  $\mathcal{U}_p$ , and the other involves *inference and redundancy elimination* of updated Knowledge Bases, denoted by  $\mathcal{U}_{ir}$ .
- (b) We analyze four different comparison functions returning sets of changes operations, namely, *explicit* ( $\Delta_e$ ), *closure* ( $\Delta_c$ ), *dense* ( $\Delta_d$ ), and *dense & closure* ( $\Delta_{dc}$ ).
- (c) We study which combinations of change operation semantics and comparison functions are correct and satisfy properties such as semantic identity and non redundancy. It should be stressed that the combination ( $\Delta_{dc}, \mathcal{U}_{ir}$ ) is quite promising: (i) it returns an empty result if K and K' are semantically equivalent (ii) the knowledge base obtained when applying  $\Delta_{dc}(K \rightarrow K')$  on K is redundancy free, and (iii) if K' is an extension of K then it is guaranteed that the Delta that we get is smaller than all comparison functions already proposed in the literature [3,31,10].



The rest of this paper is organized as follows. Section 2 provides background information regarding RDF Knowledge Bases (KB). Section 3 introduces four RDF comparison functions, Section 4 elaborates on the change operations and their semantics, while Section 5 shows the interplay between the two. Finally, Section 6 concludes the paper and identifies issues for further research.

## 2 Background: RDF KBs

In general, an RDF Knowledge Base (KB) is defined by a set of triples of the form (subject, predicate, object). Let  $\mathcal{T}$  be the set of all possible triples that can be constructed from an infinite set of URIs (for resources, classes and properties) as well as literals [15]. Then a KB can be seen as a finite subset  $K$  of  $\mathcal{T}$ , i.e.  $K \subseteq \mathcal{T}$ . Apart from the explicitly specified triples of a  $K$ , other triples can be inferred based on the semantics of RDF/S [16]. For this reason, we introduce the notion of closure and reduction of RDF KBs.

The *closure* of a  $K$ , denoted by  $C(K)$ , contains all the triples that either are explicitly specified or can be inferred from  $K$  by taking in account the semantics of the associated RDFS schemas. As we can view an RDF model as a graph, we could consider that  $C(K)$  is defined (and computed) by taking the reflexive and transitive closures of binary relations (subsumption, type [1]). If it holds  $C(K) = K$ , then we will call  $K$  *completed*. The elements of  $K$  will be called *explicit* triples, while the elements of  $C(K) - K$  will be called *inferred*. We can now define an equivalence relation between two knowledge bases.

**Def. 1.** Two knowledge bases  $K$  and  $K'$  are *equivalent*, denoted by  $K \sim K'$ , iff  $C(K) = C(K')$ .

The *reduction* of a  $K$ , denoted by  $R(K)$ , is the smallest in size set of triples such that  $C(R(K)) = C(K)$ . In general, the reduction of a  $K$  is not necessarily unique (when cycles occur in the subsumption relations). Let  $\Psi$  denote the set of all knowledge bases that have a unique reduction. Independently of whether the reduction of a  $K$  is unique or not, we can characterize a  $K$  as (semantically) *redundancy free*, and we can write  $RF(K) = True$  (or just  $RF(K)$ ), if it does not contain explicit triples which can be inferred from  $K$ . Formally,  $K$  is redundancy free if there is not any proper subset  $K'$  of  $K$  (i.e.  $K' \subset K$ ) such that  $K \sim K'$ .

## 3 RDF KBs Deltas

In this section we formally define the four comparison functions of RDF KBs introduced in Figure 1, namely,  $\Delta_e$ ,  $\Delta_c$ ,  $\Delta_d$  and  $\Delta_{dc}$ .

$$\begin{aligned} \Delta_e(K \rightarrow K') &= \{Add(t) \mid t \in K' - K\} && \cup \{Del(t) \mid t \in K - K'\} \\ \Delta_c(K \rightarrow K') &= \{Add(t) \mid t \in C(K') - C(K)\} \cup \{Del(t) \mid t \in C(K) - C(K')\} \\ \Delta_d(K \rightarrow K') &= \{Add(t) \mid t \in K' - C(K)\} && \cup \{Del(t) \mid t \in K - C(K')\} \\ \Delta_{dc}(K \rightarrow K') &= \{Add(t) \mid t \in K' - C(K)\} && \cup \{Del(t) \mid t \in C(K) - C(K')\} \end{aligned}$$

<sup>4</sup> The consequence operator of logic theories (e.g. see [11]) is out of the scope of this paper.



$\Delta_e$  (where  $e$  stands for *explicit*) actually returns the triple-set difference over the explicitly specified triples, while  $\Delta_c$  (where  $c$  stands for *closure*) returns the triple-set difference by also taking into account the inferred triples. As we mentioned in Section 1 existing approaches (e.g. [31]) are based on  $\Delta_e$  and  $\Delta_c$ . However, as we are especially interested in comparison functions that yield smaller in size Deltas, we introduce two novel comparison functions namely  $\Delta_d$  (where  $d$  comes from *dense*) and  $\Delta_{dc}$  ( $dc$  comes from *dense & closure*). It is not hard to see that  $\Delta_d$  yields smaller in size outputs (in comparison with the previous two). Unfortunately, and as we will see at Section 5,  $\Delta_d$  cannot be used in general since only for specific cases returns correct results. For this reason we additionally consider  $\Delta_{dc}$  which yields smaller in size outputs than  $\Delta_c$ . This function resembles  $\Delta_d$  regarding additions and  $\Delta_c$  regarding deletions.

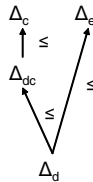
**Prop. 1.** Let  $|\Delta(K \rightarrow K')|$  to denote the *number* of change operations in  $\Delta(K \rightarrow K')$ . Then for any pair of valid knowledge bases  $K$  and  $K'$  it holds:

$$|\Delta_d(K \rightarrow K')| \leq |\Delta_c(K \rightarrow K')|$$

$$|\Delta_d(K \rightarrow K')| \leq |\Delta_{dc}(K \rightarrow K')| \leq |\Delta_c(K \rightarrow K')|$$

We have  $K \subseteq C(K) \Leftrightarrow K' - C(K) \subseteq K' - K$  (1) and  $K' \subseteq C(K') \Leftrightarrow K - C(K') \subseteq K - K'$  (2). From (1) and (2) it follows that  $|\Delta_d| \leq |\Delta_e|$ . The formula for additions is the same for both  $\Delta_d$  and  $\Delta_{dc}$ . If we consider deletions we have  $K \subseteq C(K) \Leftrightarrow K - C(K') \subseteq C(K) - C(K') \Leftrightarrow |\Delta_d| \leq |\Delta_{dc}|$ . Furthermore, we have  $K' \subseteq C(K') \Leftrightarrow K' - C(K) \subseteq C(K') - C(K)$  (3) and  $K \subseteq C(K) \Leftrightarrow K - C(K') \subseteq C(K) - C(K')$  (4). From (3) and (4) it follows that  $|\Delta_d| \leq |\Delta_c|$ . Finally, the formula for deletions is the same for both  $\Delta_c$  and  $\Delta_{dc}$ . If we consider additions we have  $K' \subseteq C(K') \Leftrightarrow K' - C(K) \subseteq C(K') - C(K) \Leftrightarrow |\Delta_{dc}| \leq |\Delta_c|$ .

In a nutshell  $\Delta_d$  gives always smaller in size Deltas while  $\Delta_{dc}$  is incomparable to  $\Delta_e$  (Figure 2 shows the Hasse diagram of the ordering relation).



**Fig. 2.** Ordering of Comparison Functions with respect to the size of their output

In the next section we will investigate what happens if we "execute" the Deltas produced by the above comparison functions under different semantics of the change update primitives  $Add(t)$  and  $Del(t)$ .

## 4 RDF KB Change Operations Semantics

A *change operation semantics* defines precisely the pre and post-conditions of the operations  $Add(t)$ ,  $Del(t)$  where  $t$  is a triple involving any RDF predicate. In Table 1

we define two alternative semantics, namely,  $\mathcal{U}_p$  ( $p$  comes from *plain*), and  $\mathcal{U}_{ir}$  ( $ir$  comes from *inference & reduction*). Under  $\mathcal{U}_p$ -semantics, the execution of the operations consists of plain set theoretic additions and deletions of triples. This implies that only the explicit triples are taken into account while inferred ones are ignored. Under  $\mathcal{U}_{ir}$ -semantics the execution of update primitives incurs also interesting side-effects such as redundancy elimination and knowledge preservation. This implies that the updated KB will not contain any explicit triple which can be inferred, while preserves as much of the knowledge expressed in  $K$  as possible (reminiscent to the postulates of the AGM theory [2] regarding contraction, and compliant with the semantics of the RUL update language [21]).

We first explain  $\mathcal{U}_{ir}$  using the example of Figure 1. If we apply on  $K$  the set  $\Delta_d$  under  $\mathcal{U}_{ir}$ -semantics, then we will indeed get  $K'$ . The insertion of  $(TA \text{ subClassOf Student})$  makes the triple  $(TA \text{ subClassOf Person})$  redundant, so the execution of  $Add(TA \text{ subClassOf Student})$  will remove  $(TA \text{ subClassOf Person})$  from the KB. Analogously, the insertion of  $(Address \text{ domain Person})$  makes the triple  $(Address \text{ domain Student})$  redundant, while the deletion of the triple  $(Jim \text{ type Student})$  will add the triple  $(Jim \text{ type Person})$ .

Returning to Table 1 for every operation  $u$  (of the form  $Add(t)$  or  $Del(t)$ ) three different, and mutually exclusive, pre-conditions are examined, namely  $t \in K$ ,  $t \in C(K) - K$  and  $t \notin C(K)$ . The post-conditions of each case are specified.  $K$  ( $K'$ ) denotes the knowledge base before (after) the execution of an operation  $u$ . Notice that post-conditions define exactly what  $K'$  will be<sup>5</sup>, unless the reduction is not unique.

**Table 1.** Two change operation semantics  $\mathcal{U}_p$  and  $\mathcal{U}_{ir}$

Change Operation Semantics $\mathcal{U}_p$				
Operation		Pre-condition	Post-condition	Comment <sup>5</sup>
$Add(t)$	1	$t \in K$	$K' = K$	void
	2	$t \in C(K) - K$	$K' = K \cup \{t\}$	addition (although already inferred)
	3	$t \notin C(K)$	$K' = K \cup \{t\}$	addition
$Del(t)$	4	$t \in K$	$K' = K - \{t\}$	deletion
	5	$t \in C(K) - K$	$K' = K$	an inferred triple cannot be deleted
	6	$t \notin C(K)$	$K' = K$	void
Change Operation Semantics $\mathcal{U}_{ir}$				
$Add(t)$	7	$t \in K$	$K' = K$	void
	8	$t \in C(K) - K$	$K' = K$	it is already inferred so it is ignored
	9	$t \notin C(K)$	$K' = R(K \cup \{t\})$	addition and then reduction
$Del(t)$	10	$t \in K$	$K' = R(C(K) - \{t\})$	deletion from closure and then reduction
	11	$t \in C(K) - K$	$K' = K$	an inferred triple cannot be deleted
	12	$t \notin C(K)$	$K' = K$	void

In particular, let  $t$  be the triple whose addition is requested. If  $t \in K$ , then under both  $\mathcal{U}_p$  and  $\mathcal{U}_{ir}$  semantics no change will be made i.e.  $K' = K$  (recall that  $K$  is a *set* of triples). If  $t \in C(K) - K$ , then under  $\mathcal{U}_p$ -semantics,  $K'$  will indeed contain that triple however, under  $\mathcal{U}_{ir}$ -semantics we will have  $K' = K$  because every triple that

<sup>5</sup> One could consider the rows of Table 1 as ECA rules where the Events correspond to column “Operation”, the Conditions correspond to column “Pre-Condition” and the Actions correspond to column “Post-condition”.

exists at  $C(K) - K$  can be inferred (and  $\mathcal{U}_{ir}$  aims at redundancy-free KBs). Finally, when requesting the addition of a triple  $t \notin C(K)$  under  $\mathcal{U}_p$ ,  $K'$  will contain that triple. Under  $\mathcal{U}_{ir}$ ,  $K'$  will contain the triples that remain after adding  $t$  to  $K$  and eliminating the redundant triples (i.e. those that can be inferred).

Let us now consider that the deletion of a triple  $t$  is requested. If  $t$  belongs to  $K$ , then  $K'$  will not contain  $t$  under  $\mathcal{U}_p$ -semantics. Under  $\mathcal{U}_{ir}$ ,  $K'$  will contain the triples that remain after deleting  $t$  from  $C(K)$  and eliminating the redundant triples (note that  $C(K)$  is used in order to preserve as much knowledge as possible). Now if  $t \in C(K) - K$ , then this request is ignored under both semantics. This means that in both semantics, *only explicit triples can be deleted*. This relieves us from having to decide which of the (possibly several) policies to adopt for reaching a  $K'$  whose closure does not contain  $t$ . Finally, if  $t \notin C(K)$ , then nothing happens as  $t$  is already out of  $K$ .

Let  $\mathcal{S}$  be the set of all possible operations of the form  $Add(t)$ ,  $Del(t)$  where  $t \in \mathcal{T}$ . Let  $S$  be a finite subset of  $\mathcal{S}$  (i.e.  $S \subset \mathcal{S}$ ). If  $\mathcal{U}$  is a symbol that denotes the semantics of a particular change operation (i.e.  $\mathcal{U}_p, \mathcal{U}_{ir}$ ), then we will use  $S^{\mathcal{U}}(K)$  to denote the result of applying  $S$  to  $K$  under  $\mathcal{U}$  semantics. Notice that the result of applying an operation is unique under  $\mathcal{U}_p$ -semantics. This is true also for  $\mathcal{U}_{ir}$  if we are in  $\Psi$  (KBs with unique reduction).

Now we introduce some notions regarding *sets* of change operations (based on [30]). Two sets of change operations  $S$  and  $S'$  are *universally equivalent* under  $\mathcal{U}$ , denoted by  $S \equiv^{\mathcal{U}} S'$ , iff  $S^{\mathcal{U}}(K) \sim S'^{\mathcal{U}}(K)$  for every possible knowledge base  $K$ .

For computing change-based Deltas we need a less strong notion of equivalence (analogously to transaction equivalence [1]).

**Def. 2.**  $S$  and  $S'$  are *equivalent over a given  $K$*  under  $\mathcal{U}$ , denoted by  $S \equiv_K^{\mathcal{U}} S'$ , iff  $S^{\mathcal{U}}(K) \sim S'^{\mathcal{U}}(K)$ .

In order to elaborate on cases where the order of execution of the update operations affects the final result, we introduce the following notion of satisfaction.

**Def. 3.** We will say that  $K$  *satisfies*: (a) an operation  $Add(t)$ , iff  $t \in C(K)$ , (b) an operation  $Del(t)$ , iff  $t \notin C(K)$  and (c) a set of change operations  $S$  (where  $S \subseteq \mathcal{S}$ ) if  $K$  satisfies *every* element of  $S$ .

If the resulting KB does not satisfy  $S$ , then we will write  $S(K) = \mathcal{E}$  where  $\mathcal{E}$  is a special symbol indicating that an error occurred. In the sequel, and for reasons of brevity, whenever we write  $S(K)$  we will also mean that  $S(K) \neq \mathcal{E}$ .

## 5 Comparison Functions and Change Operation Semantics

In this section we investigate which of the four comparison functions (introduced in Section 3) and under what semantics of update primitives (presented in Section 4) could be used for building a change-based versioning system. To this end, we have to define formally the notions of *correctness*, *semantic identity* and *redundancy*, and then elaborate on the execution of the update programmes. Finally, we will identify these pairs that are correct and the properties that they satisfy.

## 5.1 Correctness, Semantic Identity and Non Redundancy of RDF Deltas

Let  $\Delta_x$  be a comparison function, and  $\mathcal{U}_y$  be a change operation semantics.

**Def. 4.** A pair  $(\Delta_x, \mathcal{U}_y)$  is *correct* if for any pair of knowledge bases  $K$  and  $K'$ , it holds  $\Delta_x(K \rightarrow K')^{\mathcal{U}_y}(K) \sim K'$ .

Obviously, a pair  $(\Delta_x, \mathcal{U}_y)$  can be used for versioning services only if it is correct. Apart from correctness, a pair  $(\Delta_x, \mathcal{U}_y)$  may also satisfy the following properties.

(P1) If  $K \sim K'$  then  $\Delta_x(K \rightarrow K') = \emptyset$  (semantic identity)

It is desirable to have a comparison function that reports an empty result if its operands are equivalent.

(P2)  $RF(\Delta_x(K \rightarrow K')^{\mathcal{U}_y}(K))$  (non redundancy)

The resulting KB is always redundancy free (i.e. for any  $K$  and  $K'$ ).

(P2.1) If  $RF(K')$  then  $RF(\Delta_x(K \rightarrow K')^{\mathcal{U}_y}(K))$

If  $K'$  is *RF* then the resulting KB is also *RF*. Note that (P2.1) is weaker than (P2): if (P2) holds then (P2.1) holds too.

## 5.2 Executing (or Satisfying) RDF Deltas

Def. 4 presupposes that we have at our disposal an appropriate "execution mode" such that when we apply  $\Delta_x(K \rightarrow K')$  in  $K$ , and according to the selected semantics, the resulting KB will *satisfy* every element of  $\Delta_x(K \rightarrow K')$ . Of course, the above premise requires that the set  $S$  does not contain contradictions i.e. it does not contain both  $Add(t)$  and  $Del(t)$  for a given  $t$ . This is true for the comparison functions  $\Delta_e, \Delta_c, \Delta_d, \Delta_{dc}$ . However, this is not the only technical problem we have to address.

The order of execution of the change operations may affect the resulting KB, in particular the resulting KB may not satisfy all change operations returned by a comparison function (see Def. 3). For instance, for the KBs of Table 2 (d) we get  $\Delta_{dc}(K' \rightarrow K) = \{Del(A \text{ subClassOf } D), Del(B \text{ subClassOf } D), Del(C \text{ subClassOf } D)\}$ . If the operations are executed in the order  $\langle Del(A \text{ subClassOf } D), Del(B \text{ subClassOf } D), Del(C \text{ subClassOf } D) \rangle$  under  $\mathcal{U}_{ir}$  semantics, then all of them will be satisfied and the result will be equivalent to  $K$ . Now consider the following execution order  $\langle Del(B \text{ subClassOf } D), Del(C \text{ subClassOf } D), Del(A \text{ subClassOf } D) \rangle$ . In this case the operation  $Del(B \text{ subClassOf } D)$  does not change the  $K$  as it requests the deletion of an inferred triple and according to  $\mathcal{U}_{ir}$  semantics an inferred triple can't be deleted. The same will happen with the operation  $Del(C \text{ subClassOf } D)$ . Finally, the operation  $Del(A \text{ subClassOf } D)$  will be executed and will cause the addition of the triple  $(B \text{ subClassOf } D)$ . It is obvious that the operation  $Del(B \text{ subClassOf } D)$  is not satisfied by the resulting KB because it contains the triple  $(B \text{ subClassOf } D)$  i.e.  $\Delta_{dc}(K \rightarrow K')^{\mathcal{U}_{ir}} = \mathcal{E}$ . We have just seen an example where the order of execution matters. The same problem occurs when  $K'$  contains a redundant triple e.g.  $(B \text{ subClassOf } D)$ . A similar situation is encountered with  $\Delta_c$  and with  $\Delta_d$  when  $K$  and  $K'$  are not redundancy free.

To avoid nondeterminism and to ensure correctness, we need an execution semantics of change operations (comprised in Deltas) that guarantees their satisfaction (if this is

possible). This can be achieved by: (a) defining comparison functions that return *sequences* (not sets) of change operations which guarantee satisfaction of their elements, or by (b) using a multi-pass execution mode that guarantees that all change operations will eventually be satisfied. Below we elaborate on the (b) approach. We could use a loop-based algorithm which terminates when every operation returned by a comparison function is satisfied.

Alg 1. **Execute**( $K, M, sem$ ) where  $M \subseteq \mathcal{S}$ ,  $sem \in \{\mathcal{U}_p, \mathcal{U}_{ir}\}$

- (1) repeat
- (2) get an element  $u \in M$  that is not satisfied by  $K$
- (3)  $K_t = u^{sem}(K)$  // i.e. apply on  $K$  the appropriate post-conditions of  $u$  wrt  $sem$
- (4)  $K = K_t$
- (5) until  $\{u \in M | u \text{ not satisfied by } K\} = \emptyset$

In this context, we have to prove that the execution algorithm always terminates (if  $M$  has been derived from one of  $\Delta_e, \Delta_c, \Delta_d, \Delta_{dc}$ ). It is clear that the loop always terminates for the case of  $(\Delta_e, \mathcal{U}_p)$  because it yields operations that are always satisfiable. So we only have to study  $\Delta_d, \Delta_c$  and  $\Delta_{dc}$  under  $\mathcal{U}_{ir}$ -semantics.

Let  $Y$  be the satisfiable deletions and  $Z$  the unsatisfiable deletions at any point during the execution of the algorithm. If we prove that whenever  $|Y| = 0$  we also have  $|Z| = 0$  then we prove that our algorithm always terminates since all elements of  $M$  are satisfied.

Both  $\Delta_c$  and  $\Delta_{dc}$  produce the following set of delete statements:  $X = \{ Del(t') \mid t' \in C(K) - C(K') \}$ . An element  $Del(t)$  will be satisfied if  $t \in R(K)$ . So the set  $Y$ , i.e. the satisfiable deletions of  $X$ , is defined as  $Y = R(K) \cap (C(K) - C(K')) = R(K) - C(K')$ . Let's now define  $Z$ . Recall that a  $Del(t')$ , may not be satisfied (when applied to  $K$ ) only if  $t' \in C(K) - R(K)$ . So the set  $Z$ , i.e. the unsatisfiable deletions of  $X$ , is defined as  $Z = (C(K) - R(K)) \cap (C(K) - C(K')) = C(K) - (R(K) \cup C(K'))$ .

Let's now investigate whether  $|Y| = 0 \Rightarrow |Z| = 0$  holds. At first, notice that  $Y = \emptyset \Leftrightarrow R(K) - C(K') = \emptyset \Leftrightarrow R(K) \subseteq C(K')$ . Also note that  $R(K) \subseteq C(K') \Rightarrow C(K) \subseteq C(K')$ . This is based on the properties of the closure operator: if we have two sets  $A$  and  $B$  such that  $A \subseteq B$  and  $B$  is closed with respect to the closure operator  $C$  (i.e.  $C(B) = B$ ), then  $C(A) \subseteq B$ .

Returning to our problem, if  $Y = \emptyset$  (that is if  $R(K) \subseteq C(K')$ ), then the formula  $Z = C(K) - (R(K) \cup C(K'))$  is equivalent to  $Z = C(K) - C(K')$ . But above we have seen that  $Y = \emptyset \Rightarrow C(K) \subseteq C(K')$  too. It follows that  $Z = \emptyset$ . So the algorithm always terminates.

The above is actually the proof of the proposition: If  $|R(K) - C(K')| = 0$  then  $|C(K) - (R(K) \cup C(K'))| = 0$ .

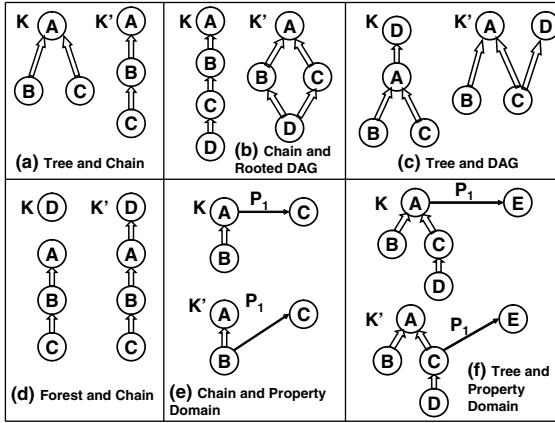
The proof for  $\Delta_d$  is similar.

### 5.3 Identifying the Correct $(\Delta_x, \mathcal{U}_y)$ -pairs

For identifying the pairs that are correct, Table 2 depicts 6 examples. For each example, it shows the result of applying  $\Delta_d, \Delta_c, \Delta_e$  and  $\Delta_{dc}$  for both  $K \rightarrow K'$  and  $K' \rightarrow K$ , and contains the following columns:

- $\mathcal{U}_p$  Co: If Y then this means that  $\Delta_x(K \rightarrow K')^{\mathcal{U}_p}(K) \sim K'$ , i.e. the approach is *correct*. Otherwise the cell is marked with N.

Table 2. Examples



<b>(a) Tree and Chain</b>								
Delta	$K \rightarrow K'$	$U_p$ Co	$U_p$ RF	$U_{irr}$ Co	$K' \rightarrow K$	$U_p$ Co	$U_p$ RF	$U_{irr}$ Co
$\Delta_e$	$\{Add(C \text{ subClassOf } B), Del(C \text{ subClassOf } A)\}$	Y	Y	Y	$\{Add(C \text{ subClassOf } A), Del(C \text{ subClassOf } B)\}$	Y	Y	Y
$\Delta_c$	$\{Add(C \text{ subClassOf } B)\}$	Y	N	Y	$\{Del(C \text{ subClassOf } B)\}$	N	Y	Y
$\Delta_d$	$\{Add(C \text{ subClassOf } B)\}$	Y	N	Y	$\{Del(C \text{ subClassOf } B)\}$	N	Y	Y
$\Delta_{dc}$	$\{Add(C \text{ subClassOf } B)\}$	Y	N	Y	$\{Del(C \text{ subClassOf } B)\}$	N	Y	Y
<b>(b) Chain and Rooted DAG</b>								
$\Delta_e$	$\{Add(C \text{ subClassOf } A), Add(D \text{ subClassOf } B), Del(C \text{ subClassOf } B)\}$	Y	Y	Y	$\{Add(C \text{ subClassOf } B), Del(C \text{ subClassOf } A), Del(D \text{ subClassOf } B)\}$	Y	Y	Y
$\Delta_c$	$\{Del(C \text{ subClassOf } B)\}$	N	Y	Y	$\{Add(C \text{ subClassOf } B)\}$	Y	N	Y
$\Delta_d$	$\{Del(C \text{ subClassOf } B)\}$	N	Y	Y	$\{Add(C \text{ subClassOf } B)\}$	Y	N	Y
$\Delta_{dc}$	$\{Del(C \text{ subClassOf } B)\}$	N	Y	Y	$\{Add(C \text{ subClassOf } B)\}$	Y	N	Y
<b>(c) Tree and DAG</b>								
$\Delta_e$	$\{Add(C \text{ subClassOf } D), Del(A \text{ subClassOf } D)\}$	Y	Y	N	$\{Add(A \text{ subClassOf } D), Del(C \text{ subClassOf } D)\}$	Y	Y	Y
$\Delta_c$	$\{Del(A \text{ subClassOf } D), Del(B \text{ subClassOf } D)\}$	N	Y	Y	$\{Add(A \text{ subClassOf } D), Add(B \text{ subClassOf } D)\}$	Y	N	Y
$\Delta_d$	$\{Del(A \text{ subClassOf } D)\}$	N	Y	N	$\{Add(A \text{ subClassOf } D)\}$	Y	N	Y
$\Delta_{dc}$	$\{Del(A \text{ subClassOf } D), Del(B \text{ subClassOf } D)\}$	N	Y	Y	$\{Add(A \text{ subClassOf } D)\}$	Y	N	Y
<b>(d) Forest and Chain</b>								
$\Delta_e$	$\{Add(A \text{ subClassOf } D)\}$	Y	Y	Y	$\{Del(A \text{ subClassOf } D)\}$	Y	Y	N
$\Delta_c$	$\{Add(A \text{ subClassOf } D), Add(B \text{ subClassOf } D), Add(C \text{ subClassOf } D)\}$	Y	N	Y	$\{Del(A \text{ subClassOf } D), Del(B \text{ subClassOf } D), Del(C \text{ subClassOf } D)\}$	Y	Y	Y
$\Delta_d$	$\{Add(A \text{ subClassOf } D)\}$	Y	Y	Y	$\{Del(A \text{ subClassOf } D)\}$	Y	Y	N
$\Delta_{dc}$	$\{Add(A \text{ subClassOf } D)\}$	Y	Y	Y	$\{Del(A \text{ subClassOf } D), Del(B \text{ subClassOf } D), Del(C \text{ subClassOf } D)\}$	Y	Y	Y
<b>(e) Chain and Property Domain</b>								
$\Delta_e$	$\{Add(B, P_1, C), Del(A, P_1, C)\}$	Y	Y	Y	$\{Add(A, P_1, C), Del(B, P_1, C)\}$	Y	Y	Y
$\Delta_c$	$\{Del(A, P_1, C)\}$	N	Y	Y	$\{Add(A, P_1, C)\}$	Y	N	Y
$\Delta_d$	$\{Del(A, P_1, C)\}$	N	Y	Y	$\{Add(A, P_1, C)\}$	Y	N	Y
$\Delta_{dc}$	$\{Del(A, P_1, C)\}$	N	Y	Y	$\{Add(A, P_1, C)\}$	Y	N	Y
<b>(f) Tree and Property Domain</b>								
$\Delta_e$	$\{Add(C, P_1, E), Del(A, P_1, E)\}$	Y	Y	N	$\{Add(A, P_1, E), Del(C, P_1, E)\}$	Y	Y	Y
$\Delta_c$	$\{Del(A, P_1, E), Del(B, P_1, E)\}$	N	Y	Y	$\{Add(A, P_1, E), Add(B, P_1, E)\}$	Y	N	Y
$\Delta_d$	$\{Del(A, P_1, E)\}$	N	Y	N	$\{Add(A, P_1, E)\}$	Y	N	Y
$\Delta_{dc}$	$\{Del(A, P_1, E), Del(B, P_1, E)\}$	N	Y	Y	$\{Add(A, P_1, E)\}$	Y	N	Y

- $\mathcal{U}_p$  RF: If Y then this means that the application of these changes results in a *redundancy free*  $K$ . Formally Y iff  $RF(\Delta_x(K \rightarrow K')^{\mathcal{U}_p}(K))$ . Otherwise the cell is marked with N.
- $\mathcal{U}_{ir}$  Co: If Y then this means that  $\Delta_x(K \rightarrow K')^{\mathcal{U}_{ir}}(K) \sim K'$  i.e the approach is correct. Otherwise the cell is marked with N.

In all cases we assume that the KBs are redundancy free. We do not have a column " $\mathcal{U}_{ir}$  RF" because by definition the execution of a  $\mathcal{U}_{ir}$ -operation leaves the knowledge base in a redundancy free state. Those pairs that have a N in the cells that concern correctness, constitute a proof (by counterexample) that they are *not correct*. For the rest pairs (those with a Y) we have to prove that they are *always* correct.

**Theorem 1.** For any pair of valid knowledge bases  $\{K, K'\} \subseteq \Psi$  it holds:

$$\Delta_c(K \rightarrow K')^{\mathcal{U}_{ir}}(K) \sim \Delta_e(K \rightarrow K')^{\mathcal{U}_p}(K) \sim \Delta_{dc}(K \rightarrow K')^{\mathcal{U}_{ir}}(K) \sim K'$$

**Theorem 2.**  $\Delta_d(K \rightarrow K')^{\mathcal{U}_{ir}}(K) \sim K'$  iff  $\{K, K'\} \subseteq \Psi$  and either: (a)  $K$  is complete, or (b)  $C(K) - K \subseteq C(K')$ .

Due to space limitations the proof of the above theorems is omitted. An interesting remark regarding Th. 2 is that if  $C(K') \supseteq C(K)$ , then condition (b) holds. This means that we could use the pair  $(\Delta_d, \mathcal{U}_{ir})$  in cases we know that  $C(K') \supseteq C(K)$ . For example if  $K$  is an ontology  $O$  and  $K'$  is an additional ontology  $O'$  that specializes  $O$ , then we are sure that  $C(K') \supseteq C(K)$ . In such cases we can use  $\Delta_d$  (or alternatively  $\Delta_{dc}$ ) which give the smallest in size Deltas ( $\Delta_{dc}$  returns the same Deltas).

### 5.4 Semantic Identify and Non Redundancy Properties of $(\Delta_x, \mathcal{U}_y)$ -Pairs

**Prop. 2.** If  $K \sim K'$  then  $\Delta_d(K \rightarrow K') = \Delta_c(K \rightarrow K') = \Delta_{dc}(K \rightarrow K') = \emptyset$ .

This is property (P1) and its proof is trivial. Note that  $\Delta_e$  is not included in Prop. 2 because even if  $K \sim K'$ , it may be  $K = K'$ ,  $K \subset K'$ ,  $K' \subset K$ , or  $K \not\subseteq K'$  and  $K' \not\subseteq K$ . In the example of Figure 3 (a) we get  $\Delta_e(K \rightarrow K') = \{Add(C \text{ subclassOf } A)\}$  although  $K \sim K'$ . It should be stressed that most of the existing comparison functions [3][31][10] actually employ  $\Delta_e$ , so they do not satisfy (P1).

However, one can easily prove that: If  $K$  and  $K'$  are both redundancy free, and the knowledge bases considered have always a unique reduction, then  $K \sim K' \Rightarrow \Delta_e(K \rightarrow K') = \emptyset$ . In general, if the transitive closure of a binary relation  $R$  is antisymmetric and finite, then the transitive reduction of  $R$  is unique. In the problem at

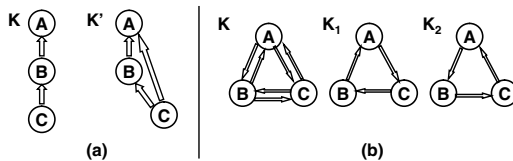


Fig. 3.  $K, K'$

hand, if an RDF knowledge base allows forming cycles with subsumption relationships, then the transitive reduction is not unique. For example, in Figure 3(b) we have  $K \sim K_1 \sim K_2$ , moreover  $RF(K_1), RF(K_2)$ , but  $K_1 \neq K_2$ .

**Prop. 3.** If  $K \sim K'$ ,  $\{K, K'\} \subseteq \Psi$  and  $RF(K), RF(K')$  then  $\Delta_e(K \rightarrow K') = \emptyset$

## 5.5 Summarizing the Results

The pairs that are always correct are:  $(\Delta_c, \mathcal{U}_{ir})$ ,  $(\Delta_e, \mathcal{U}_p)$  and  $(\Delta_{dc}, \mathcal{U}_{ir})$ . The pair  $(\Delta_c, \mathcal{U}_p)$  is correct if  $K$  is complete. The pair  $(\Delta_d, \mathcal{U}_{ir})$  is correct in the cases specified in Theorem 2. The set of change operations derived from either  $\Delta_c$  or  $\Delta_{dc}$  need the multi-pass execution mode while  $\Delta_e$  requires a single pass execution mode. Concerning the size criterion,  $\Delta_d$  produces the smallest in size result.  $\Delta_{dc}$  produces smaller results than  $\Delta_c$ . Concluding, we can say that the pairs  $(\Delta_{dc}, \mathcal{U}_{ir})$  and  $(\Delta_e, \mathcal{U}_p)$  are the *most appropriate for implementing change-based versioning services*: they are always correct and the size of  $\Delta_{dc}$  is less than  $\Delta_c$ . We cannot however compare the size of  $\Delta_e$  with that of  $\Delta_{dc}$  (in some cases the first is smaller, in others the second). Table 3 synthesizes the results. Concerning the column labeled "Execution Mode", S is used to denote single pass, and M to denote multi pass.

It is worth mentioning that  $(\Delta_e, \mathcal{U}_p)$  is correct even if we are not in  $\Psi$ . Moreover, Theorem 1 holds even if we are not in  $\Psi$  but we adopt a "batch" execution mode for  $\mathcal{U}_{ir}$ , where each change operation is not executed independently but all change operations of the produced Delta are executed as one "transaction", i.e. we compute the closure and the reduction only once.

**Table 3.** Synopsis

Comp.	Sem.	Always Correct	Exec Mode	(P1)	(P2)	(P2.1)
$\Delta_e$	$\mathcal{U}_p$	Y	S	see (Prop. 3)	N	Y
$\Delta_c$	$\mathcal{U}_p$	Y if $K$ complete	S	Y	N	
$\Delta_d$	$\mathcal{U}_p$	N		Y	N	
$\Delta_{dc}$	$\mathcal{U}_p$	N		Y	N	
$\Delta_e$	$\mathcal{U}_{ir}$	N		see (Prop. 3)	Y	Y
$\Delta_c$	$\mathcal{U}_{ir}$	Y in $\Psi$	M	Y	Y	Y
$\Delta_d$	$\mathcal{U}_{ir}$	(see Theorem 2)	M	Y	Y	Y
$\Delta_{dc}$	$\mathcal{U}_{ir}$	Y in $\Psi$	M	Y	Y	Y

## 6 Concluding Remarks

One approach for computing the difference between two RDF models is to take the difference between the sets of triples forming the two models (along with some refinements such as taking into account blank nodes). Another approach (useful for



versioning) is to identify a set of change operations that will transform one model into the other. In this paper we investigated the second approach and studied different semantics for this computation as well as properties like minimality and correctness of the produced Deltas. Most of the existing RDF comparison tools [33][10] rely on the  $(\Delta_e, \mathcal{U}_p)$  pair. Semversion [31] offers also  $(\Delta_c, \mathcal{U}_p)$  for the case where the  $K$  is complete (we have proved that in such cases this approach yields correct results). None of the works (theoretical or practical) has used  $\Delta_d$  or  $\Delta_{dc}$ . Recall that we have shown that  $(\Delta_{dc}, \mathcal{U}_{ir})$  is better than  $(\Delta_c, \mathcal{U}_p)$  not only because  $(\Delta_{dc}, \mathcal{U}_{ir})$  does not require the KBs to be complete, but also because it returns smaller in size Deltas. We have identified the cases where  $(\Delta_d, \mathcal{U}_{ir})$  is beneficial (recall that  $\Delta_d$  gives the minimum in size Deltas). An issue for further research is to identify the conditions under which  $\Delta_e$  yields smaller Deltas than  $\Delta_{dc}$  and vice versa.

In comparison with belief contraction-revision (e.g. [14][20][11]), these theories consider KBs as logic theories and focus on what the result of applying a contraction/revision operation on a KB should be. In our setting, the destination KB is known, i.e. it is  $K'$ , so the focus is given on the transition from  $K$  to  $K'$ .

We plan to exploit the properties of the various Delta functions presented in this paper for building versioning services on top of SW repositories [28]. For reasons of space, technical details, as well as issues regarding the peculiarities of RDF including blank nodes identification, and containers (Bag, Sequence, Alternative) are omitted. An important implementation issue that is worth mentioning is that the algorithm implementing the Delta functions never computes the closure of a KB. Instead, it constructs the explicit graph of each KB, and then it checks whether  $t \in C(K)$  which can be decided efficiently (in  $O(1)$ ), thanks to a labeling scheme [7] for subsumption relationships that is supported by RDFSuite [12]. Concerning the execution of  $\mathcal{U}_{ir}$  operations, related algorithms include [26], while a similar in spirit approach for RDF has already been implemented for the RUL language [21].

**Acknowledgements.** This work was partially supported by the EU projects CASPAR (FP6-2005-IST-033572) and KP-Lab (FP6-2004-IST-4).

## References

1. Abiteboul, S., Vianu, V.: Equivalence and optimization of relational transactions. *Journal of the ACM (JACM)* 35(1), 70–120 (1988)
2. Alchourrón, C.E., Gärdenfors, P., Makinson, D.: On the Logic of Theory Change: Partial Meet Contraction and Revision Functions. *J. Symb. Log.* 50(2), 510–530 (1985)
3. Beners-Lee, T., Connolly, D.: Delta: An Ontology for the Distribution of Differences Between RDF Graphs, (version: 2006-05-12) (2004), <http://www.w3.org/DesignIssues/Diff>

<sup>6</sup> Note that  $\mathcal{U}_p, \mathcal{U}_{ir}$  are not proposed as general purpose change operations, but only for executing the results of the comparison functions we have defined in this paper.

<sup>7</sup> One approach to tackle the blank node identification problem is to consider that a KB is not a set of RDF triples but a set of RDF molecules [10][327]. Alternative techniques include [6].

4. Berliner, B.: CVS II: Parallelizing Software Development. In: Proc of the USENIX Winter 1990 Technical Conference, Berkeley, CA, pp. 341–352 (1990), [citeseer.ist.psu.edu/berliner90cvs.html](http://citeseer.ist.psu.edu/berliner90cvs.html)
5. Brickley, D. and Guha, R.V. RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation (February 2004) <http://www.w3.org/TR/rdf-schema/>
6. Carroll, J.J.: Matching RDF graphs. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 5–15. Springer, Heidelberg (2002)
7. Christophides, V., Plexousakis, D., Scholl, M., Tourtounis, S.: On Labeling Schemes for the Semantic Web. In: Proc. of WWW 2003, Budapest, Hungary, pp. 544–555 (May 2003)
8. Cloran, R., Irwin, B.: Transmitting RDF graph deltas for a Cheaper Semantic Web. In: Proc. of SATNAC 2005. South Africa (September 2005)
9. Conradi, R., Westfechtel, B.: Version models for software configuration management. ACM Comput. Surv. 30(2), 232–282 (1998)
10. Ding, L., Finin, T., Joshi, A., Peng, Y., Silva, P.d., McGuinness, D.: Tracking RDF Graph Provenance using RDF Molecules. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
11. Flouris, G.: On Belief Change and Ontology Evolution. PhD thesis, Computer Science Department, University of Crete, Greece (2006)
12. FORTH-ICS. The ICS-FORTH RDFSuite: High-level Scalable Tools for the Semantic Web (2005), <http://139.91.183.30:9090/RDF/>
13. Petersson, J., Piazza, F., Puliti, P., Tummarello, G., Morbidoni, C.: RDFGrowth, a P2P annotation exchange algorithm for scalable Semantic Web applications. In: 1st Annual International Conference on Mobile and Ubiquitous Systems MobiQuitous, Boston, MA, August 2004 (2004)
14. Gärdenfors, P.: Belief Revision: An Introduction. In: Belief Revision, pp. 1–20. Cambridge University Press, Cambridge (1992)
15. Gutierrez, C., Hurtado, C., Mendelzon, A.: Foundations of Semantic Web Databases. In: PODS 2004. 23 ACM Symposium on Principles of Database Systems, ACM Press, New York (2004)
16. Hayes, P. RDF Semantics, W3C Recommendation (February 2004) <http://www.w3.org/TR/rdf-mt/>
17. Heflin, J., Hendler, J., Luke, S.: Coping with Changing Ontologies in a Distributed Environment. In: Proc. of AAAI 1999 Workshop on Ontology Management, Florida (July 1999)
18. Klein, M., Fensel, D., Kiryakov, A., Ognyanov, D.: Ontology versioning and change detection on the web. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS (LNAI), vol. 2473, pp. 197–212. Springer, Heidelberg (2002)
19. Klein, M., Noy, N.: A component-based framework for ontology evolution. In: IJCAI 2003. Workshop on Ontologies and Distributed Systems, Acapulco, Mexico (2003)
20. Konieczny, S., Perez, R.P.: Propositional Belief Base Merging or How to Merge Beliefs/Goals Coming from Several Sources and Some Links With Social Choice Theory. European Journal of Operational Research 160(3), 785–802 (2005)
21. Magiridou, M., Sahtouris, S., Christophides, V., Koubarakis, M.: RUL:A Declarative Update Language for RDF. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 506–521. Springer, Heidelberg (2005)
22. Klein, M., Noy, N.F., Kunnatur, S., Musen, M.A.: Tracking Changes During Ontology Evolution. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 259–273. Springer, Heidelberg (2004)
23. Noy, N.F., Musen, M.A.: Prompt Diff: A Fixed-point Algorithm for Comparing Ontology Versions. In: Proc of AAAI 2002, Edmonton, Alberta, pp. 744–750 ( July 2002)
24. Noy, N.F., Musen, M.A.: Ontology versioning in an ontology management framework. IEEE Intelligent Systems 19(4), 6–13 (2004)

25. Plessers, P., Troyer, O.D.: Ontology Change Detection Using a Version Log. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 578–592. Springer, Heidelberg (2005)
26. La Poutre, J.A., van Leeuwen, J.: Maintenance of Transitive Closures and Transitive Reductions of Graphs. In: *Procs. of the Intern. Workshop on Graph-Theoretic Concepts in Computer Science*, pp. 106–120 (July 1987)
27. Stickler, P.: CBD - Concise Bounded Description. W3C Member Submission June 2005), <http://www.w3.org/Submission/CBD/>
28. Theoharis, Y., Christophides, V., Karvounarakis, G.: Benchmarking Database Representations of RDF/S Stores. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 685–701. Springer, Heidelberg (2005)
29. Tichy, W.F.: RCS-a system for version control. *Software Practice & Experience* 15(7), 637–654 (1985)
30. Tzitzikas, Y., Kotzinos, D. (Semantic Web) Evolution through Change Logs: Problems and Solutions. In: *Procs of AIA 2007, Innsbruck, Austria (February 2007)*
31. Volkel, M., Winkler, W., Sure, Y., Kruk, S.R., Synak, M.: SemVersion: A Versioning System for RDF and Ontologies. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, Springer, Heidelberg (2005)
32. Zhang, Z., Zhang, L., Lin, C., Zhao, Y., Yu, Y.: Data Migration for Ontology Evolution. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, Springer, Heidelberg (2003)

# Semplore: An IR Approach to Scalable Hybrid Query of Semantic Web Data

Lei Zhang<sup>1</sup>, QiaoLing Liu<sup>2</sup>, Jie Zhang<sup>2</sup>, HaoFen Wang<sup>2</sup>,  
Yue Pan<sup>1</sup>, and Yong Yu<sup>2</sup>

<sup>1</sup> IBM China Research Lab  
Beijing, 100094, China

lzhang1,panyue@cn.ibm.com

<sup>2</sup> Department of Computer Science & Engineering  
Shanghai Jiao Tong University, Shanghai, 200240, China

{lql,zhangjie,whfcarter,yyu}@apex.sjtu.edu.cn

**Abstract.** As an extension to the current Web, Semantic Web will not only contain structured data with machine understandable semantics but also textual information. While structured queries can be used to find information more precisely on the Semantic Web, keyword searches are still needed to help exploit textual information. It thus becomes very important that we can combine precise structured queries with imprecise keyword searches to have a hybrid query capability. In addition, due to the huge volume of information on the Semantic Web, the hybrid query must be processed in a very scalable way. In this paper, we define such a hybrid query capability that combines unary tree-shaped structured queries with keyword searches. We show how existing information retrieval (IR) index structures and functions can be reused to index semantic web data and its textual information, and how the hybrid query is evaluated on the index structure using IR engines in an efficient and scalable manner. We implemented this IR approach in an engine called Semplore. Comprehensive experiments on its performance show that it is a promising approach. It leads us to believe that it may be possible to evolve current web search engines to query and search the Semantic Web. Finally, we briefly describe how Semplore is used for searching Wikipedia and an IBM customer's product information.

## 1 Introduction

With more and more structured and semantic information made available on the Semantic Web, structured queries such as SPARQL can be used to find information more precisely. At the same time, current web search is dominated by the form of keyword searches. Although precise structured queries generally produce far better results than imprecise keyword searches, keyword search capability is still needed in Semantic Web because: (1) The huge amount of textual information in the (Semantic) Web will remain to be a valuable source of information that need be exploited using keyword searches; (2) Users often have vague information needs that can hardly be expressed as formal queries; and (3) keyword

searches are extremely simple to use. It thus becomes very important that we can combine precise structured queries with imprecise keyword searches to have a hybrid query capability. On the other hand, as an extension to the current Web, Semantic Web will have an even larger volume of data and textual information. Hybrid queries against such large volume of web-scale data must be evaluated in a very scalable way.

Current research on searching or querying Semantic Web uses either an IR-based (e.g. [1,2,3]) or a DB-based (e.g. [4,5,6,7]) approach. The IR-based work does not provide structured query capability and the DB-based work lacks support to keyword searches. Few work [8] tries to combine them to achieve hybrid query capability. DB-based work pays more attention to the scalability of query-answering and it relies on database's various indices and query optimization algorithms to support efficient evaluation of complex queries. However, taking DB engines to support complex queries on web-scale data is still a big challenge. In contrast, IR engine is a special-purpose engine supporting only keyword searches but has proven to be able to scale to the size of the Web. Current web search engines have developed scalable method to process keyword searches using classic IR techniques with distributed and parallel backend infrastructure [9]. This inspires us to trade query capability for scalability and to try an IR approach of indexing, querying and searching of semantic web data. The problem then breaks down to the following points:

- What is the hybrid query capability ?
- How to index semantic web data using existing IR index structures which are designed for textual information ?
- How to use IR engines to answer the hybrid queries and maintain the efficiency and scalability ?

In this paper, we show how we tried to solve these problems and how we implemented the solution in Semplere using a popular open-source IR engine – Lucene. Our experiments show that the IR approach is promising and it leads us to believe that it may be possible to evolve current web search engine's powerful backend IR infrastructure for querying and searching the Semantic Web and ultimately evolve them to web-scale semantic web search engines.

The paper is organized as follows. Section 2 defines the hybrid query capability. Section 3 describes in detail the methods of indexing and querying of semantic web data using IR index structure and engine. Section 4 then reports the experiment results of Semplere and briefly describes its two applications. We discuss related work in Section 5 and conclude the paper in Section 6.

## 2 Hybrid Query Capability

[10] introduced a DL-based formal conjunctive query language for Semantic Web. Conjunctive queries are also the formal core of SPARQL query language [11]. We thus use conjunctive queries as the basis of the hybrid query. We then make an extension to ordinary conjunctive queries to combine keyword search for

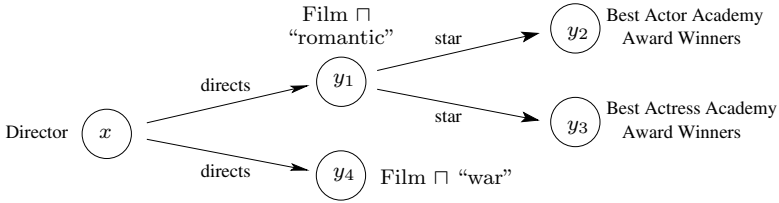


Fig. 1. An Example Query Graph

semantic web data as well. The idea is that we view the searched keywords as a “virtual” concept called keyword concept  $W$ . An individual will be regarded as an instance of a keyword concept  $W$  if the textual content of any of its datatype properties contains the searched keywords in  $W$  (i.e., we adopt a boolean IR model). This idea derives from our previous work in [8].

Based on that, we can formally define the hybrid query capability of Semplore. A *unary hybrid conjunctive query*  $q$  over a knowledge base  $K$  is a query expression of the form

$$q(x) \leftarrow \exists \vec{y}. conj(x, \vec{y})$$

where  $x$  is called the target variable,  $\vec{y}$  are existentially quantified variables called non-distinguished variables, and  $conj(x, \vec{y})$  is a conjunction of terms of the form  $C(z)$ ,  $R(z_1, z_2)$ , or  $R^-(z_1, z_2)$ .  $z, z_1, z_2$  are individuals in  $K$  or variables in  $x$  or  $\vec{y}$ .  $R$  is a role/relation name and  $C$  (or  $D$ ) is a concept expression that is a boolean combination of concept name  $A$  and keyword concept  $W$ :

$$C, D := \top \mid \perp \mid A \mid W \mid \{i\} \mid C \sqcap D \mid C \sqcup D \mid \neg C$$

The answer to the query w.r.t  $K$  is the set defined by  $\{a \in O \mid K \models q[x/a]\}$ , where  $O$  denotes the set of all individual names in  $K$ , and  $q[x/a]$  denotes the query  $q$  with all occurrences of variable  $x$  substituted by the individual name  $a$ .

The above query can be depicted as a directed graph, where the nodes are variables or individual names and the edges are relations connecting them. Concept expressions and relation names provide labels for nodes and edges respectively. Fig. 1 shows an example of such a query graph and [10] has details about the definition of a query graph.

In this paper, we restrict to queries whose graph patterns are trees, as in [10]. We also restrict the query result to be unary (i.e., a single target variable  $x$  in the query). These restrictions lead to a much more simplified and hence efficient query evaluation procedure, while a large portion of information needs can still be expressed. This is the major place where we trade query capability for scalability.

The query capability of Semplore can then be defined primarily as *unary tree-shaped hybrid query*. It’s not hard to see that the unary tree-shaped hybrid

<sup>1</sup> If the node is an individual  $i$ , the concept expression is  $\{i\}$ . If a node have no label on it, we add the concept expression  $\top$  as its label.

**Table 1.** Translating Semantic Web Data to “Documents” in IR

Document	Field	Term
concept $C$	subConOf	super-concepts of $C$
	superConOf	sub-concepts of $C$
	text	tokens in textual properties of $C$
relation $R$	subRelOf	super-relations of $R$
	superRelOf	sub-relations of $R$
	text	tokens in textual properties of $R$
individual $i$	type	concepts that $i$ belongs to
	subjOf	all relations $R$ that $(i, R, ?)$ is a triple in data
	objOf	all relations $R$ that $(?, R, i)$ is a triple in data
	text	tokens in textual properties of $i$

queries without keyword concepts is a strict subset of SPARQL queries. In the next section, we show that the hybrid query capability can be achieved using IR engines on semantic web data.

### 3 Semplore Engine

#### 3.1 Index Structure

IR indexing is based on the concepts of documents, fields(e.g. title, abstract, etc.), and terms. Based on the classic inverted index structure, IR engines can efficiently retrieve documents given a boolean combination of pairs of (field, term) as a query. Current web search engine has implemented this technique on the web scale.

Our intuition is that if we treat individuals as documents and their associated concept names as terms, we can then retrieve all individuals of a given concept by inputting the concept name as a query term into the IR engine. Extending this intuition, we realize that we can answer many kinds of semantic queries using IR engines if we translate semantic web data into documents, fields and terms in a proper way as shown in Table 1. After the translation, IR engine can index semantic web data in its inverted index structure and provide retrieval functions over the data. In addition, in order to return inferred data in query result, we require that the semantic web data be preprocessed by a reasoning engine to contain all inferred data.

The idea of treating concepts as terms to index individuals is not new. Previous work [12] proposed and analyzed more complex labeling schemes for indexing semantic web data. What’s new in our work, however, is on how to index relation instances  $(s, R, o)$ . The index should enable us to find all the objects of a relation with a given set of subjects (e.g. find all the films directed by some Chinese director:  $\{f \mid \text{directs}(d, f) \wedge \text{ChineseDirector}(d)\}$ ) and vice versa.

We propose an approach called PosIdx to index a relation instance  $(s, R, o)$ . Recall that in the classic inverted index structure, for each term there is a posting list of documents that contain the term. In addition, for each document in the

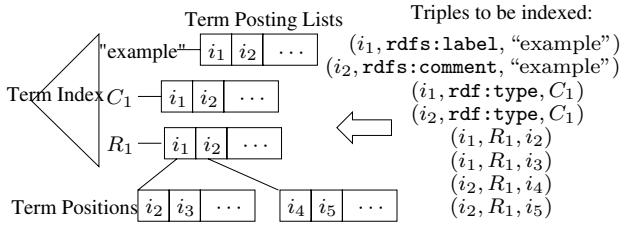


Fig. 2. PosIdx Index Structure Example

posting list, there is also a list of positions showing where the term appears in the document. In the PosIdx method, subject  $s$  is treated as a document with a field named `subjOf` and a term  $R$  in the field. Object  $o$  is stored as a “position” of the term  $R$  in the document  $s$ . We use the position list to store the objects of relation  $R$  under subject  $s$ . An example index structure is depicted in Fig. 2 (field information is omitted for brevity). The `objOf` field is a symmetric case of `subjOf` and is also not shown in Fig. 2. If we see `subjOf` as a field for indexing instances of relation  $R$ , the `objOf` field can then be seen as indexing instances of relation  $R^-$ .

As to the physical storage and access of the logical inverted index structure, it has been thoroughly studied in the IR field, which results in many optimized methods, such as byte-aligned index compression [13] and self-indexing [14]. Furthermore, in the proposed PosIdx method, relation objects enjoy the benefit of spatial locality for fast access, because positions of a term are usually physically stored together and continuously in modern IR engines. In fact, in our Semplore implementation, we enjoy all the above-mentioned optimizations and benefits because they are already built into the underlying IR engine Lucene.

The use of IR approach also enables us to leverage the heavy optimizations on the architectures and algorithms for building index in IR field (e.g. [15]). In a real (Semantic) Web search engine, advanced techniques such as MapReduce [16] can be applied on a cluster of machines to speed up the index building process, thanks to the simplicity of the IR index structure.

### 3.2 Query Evaluation

Based on the above index, Semplore reuses IR engine’s merge-sort based boolean query evaluation method and extends it to answer the unary tree-shaped hybrid query defined in Section 2. In the following, we first introduce and explain some basic operations and their corresponding notations, then we describe the query evaluation algorithm of Semplore.

**Basic Operations.** We generalize the notion of a posting list to an Ascending Integer Stream (AIS) which can be accessed from the smallest integer to the largest one. By adding additional indexing structures to the inverted index (e.g., self-indexing [14]), modern IR engines can supply a very efficient stream reader for a posting list AIS.



(1) Basic retrieval:  $(f, t)$

Given a field  $f$  and a term  $t$ ,  $(f, t)$  retrieves the corresponding posting list from inverted index. This is a standard IR operation. The output of this operation is an AIS. For example, under Table 1's structure,  $(\text{type}, \text{ChineseDirector})$  will retrieve all individuals of the `ChineseDirector` concept as an AIS.

(2) Merge-sort:  $m(S_1, op, S_2)$

$S_1$  and  $S_2$  are two AISs and  $op$  is a binary operator which can be  $\cap$ ,  $\cup$  or  $-$ . Merge-sort computes  $S_1 op S_2$  and returns a new AIS. Merge-sort can be nested to compute boolean combinations of multiple AISs. IR research has developed efficient algorithms to do nested merge-sort on AISs.

(3) Concept expression evaluation:  $\lambda(C)$

The input of this operation is a concept expression  $C$  as defined in Section 2 which is a boolean combination of concept name  $A$  and keyword concept  $W$ . The output of this operation is an AIS containing all the IDs of the individuals of  $C$ . This operation can be implemented using basic retrievals and nested merge-sort operations, thus is also readily available in modern IR engines. For example, if  $C$  is  $\text{Film} \sqcap \text{"war"}$ , the  $\lambda(C)$  operation can then be achieved through two basic retrievals and one merge-sort:  $m((\text{type}, \text{Film}), \cap, (\text{text}, \text{"war"}))$ .

(4) Relation expansion:  $\bowtie(S_1, R, S_2)$

The input of the operation is a relation  $R$  and two AISs  $S_1$  and  $S_2$  that contain individual IDs. The operation computes the set  $\{y \mid \exists x : x \in S_1 \wedge (x, R, y) \wedge y \in S_2\}$  and returns it as an AIS. For example,  $\bowtie(\lambda(\text{ChineseDirector}), \text{directs}, \lambda(\text{DocumentaryFilm}))$  can be used to find all documentary films directed by some Chinese director. This operation is not available in a classic IR engine. We will show later how it can be computed on the index structure we proposed.

**Evaluation Algorithm.** Based on the above four basic operations, Algorithm 1 shows how a *unary tree-shaped hybrid query* defined in Section 2 can be evaluated. The algorithm can be visually imagined as traversing the query tree in the depth-first order. It evaluates the concept expression of each vertex when moving forward and uses results of children to constrain the results of parent when moving backward. It will terminate in  $2 * E$  steps each of which is either a  $\lambda(C)$  operation or a  $\bowtie(S_1, R, S_2)$  operation.

Taking Fig 1 as an example with target variable  $x$ , we first reach to leaf vertex  $y_2$  via  $x$  and  $y_1$ , and compute their results  $S[x], S[y_1], S[y_2]$ . Then  $S[y_1] = \bowtie(S[y_2], \text{star}^-, S[y_1])$  is computed when we move backward from  $y_2$  to  $y_1$ . Similarly,  $y_3$  is traversed and its result also adds constraint to result of its parent  $y_1$ . Next, result of root vertex  $x$  is updated by  $S[x] = \bowtie(S[y_1], \text{directs}^-, S[x])$ . Finally,  $y_4$  is traversed and the result of root  $x$  is updated again, which is the final answer.

**Algorithm 1.** Query Evaluation Algorithm

**Input** : A unary tree-shaped hybrid query  $Q(t)$  with graph  $G = (V, E)$  and target variable  $t$  on the vertex  $v_t \in V$ ; Each vertex  $u \in V$  has a concept expression  $C_u$  as its label and each edge  $(u, v) \in E$  has a relation  $R_{(u,v)}$  as its label. (Note that  $R_{(u,v)}$  is equal to  $R_{(v,u)}^-$ .)

**Output:** An AIS containing the IDs of individuals in the answer set of  $Q(t)$

```

1 foreach vertex  $u \in V$  do
2    $checked[u] = false; S[u] = null;$ 
3  $DFS(v_t);$ 
4 return  $S[v_t]$ 

```

**Procedure**  $DFS(u)$ 

```

1  $checked[u] = true;$ 
2  $S[u] = \lambda(C_u);$ 
3 foreach vertex  $v$  such that  $(u, v) \in E$  or  $(v, u) \in E$  do
4   if  $(checked[v] == true)$  then continue  $v;$ 
5    $DFS(v);$ 
6   if  $((v, u) \in E)$  then  $S[u] = \bowtie(S[v], R_{(v,u)}, S[u]);$ 
7   else  $S[u] = \bowtie(S[v], R_{(u,v)}^-, S[u]);$ 

```

**Relation Expansion.** Among the four basic operations used for query evaluation, relation expansion  $\bowtie(S_1, R, S_2)$  is the one that is not directly supported by current IR engines. However, it can be evaluated in four steps using additional operations. Fig. 3 shows how this is done on the PosIdx index structure.

First, we compute the valid subjects that have  $R$  relations:  $S = m(S_1, \cap, (\text{subjOf}, R))$ .<sup>2</sup> Second, we find the objects for each valid subject  $s \in S$  using an operation called  $\text{getObjects}(s, R)$  which returns the object set  $\{o \mid (s, R, o)\}$  as an AIS, given  $s$  and  $R$ . Third, we union all these object sets and sort the result set to obtain a new AIS  $S_O$ . This step is encapsulated in an operation called  $\text{massUnion}(S, R) = S_O = \bigcup_{s \in S} \text{getObjects}(s, R)$ . Finally, we do a merge-sort  $m(S_O, \cap, S_2)$  to obtain the final result. This final step can be integrated into the  $\text{massUnion}$  operation as we will show later. In short, relation expansion can be completed using basic retrieval, merge-sort and two additional operations:  $\text{getObjects}$  and  $\text{massUnion}$ .

When the number of valid subjects in  $S$  is large, the  $\text{massUnion}(S, R)$  operation becomes expensive because it has to union and sort a large number of sets stored on disk, which is an I/O bound operation. In Fig. 3, these sets are the shaded segments in term  $R$ 's position list. If we still do a streaming merge-sort on all the sets to obtain all the objects, it would incur a prohibitive I/O cost because it will lead to a large number of back-and-forth disk seeks. One way to save I/O cost is to selectively union a subset of all the sets that can still cover all the objects. However, selecting such a subset is the Set-Cover problem which is NP hard [17].

<sup>2</sup> If the relation is  $R^-$ , replace  $\text{subjOf}$  with  $\text{objOf}$ . Similarly, in the following steps  $\text{getSubjects}(o, R)$  is used instead of  $\text{getObjects}(s, R)$ .

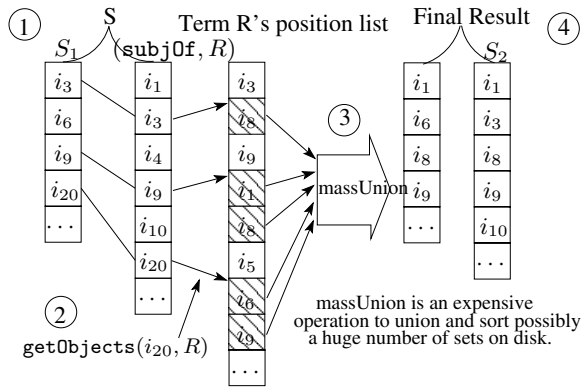


Fig. 3. Relation Expansion on PosIdx

---

**Algorithm 3.** BVI for  $\bowtie (S_1, R, S_2)$

---

- 1 Do a basic retrieval  $O_R = (\text{objOf}, R)$ ; Let  $N = |O_R|$  and  $T = 0$ ;
  - 2 Allocate two bit vectors  $B_1$  and  $B_2$  of size  $N$  and initialize them to all 0s;
  - 3 Do  $m(O_R, \cap, S_2)$ . During the merge-sort, set  $B_2[i] = 1$  for the  $i$ th element in  $O_R$  that is also in  $S_2$ . Let  $M$  be the number of 1 bits in  $B_2$ ;
  - 4 **foreach**  $s \in m(S_1, (\text{subjOf}, R))$  **do**
  - 5      $O_s = \text{getObjects}(s, R)$  ;
  - 6     **foreach** *sequence number*  $i \in O_s$  **do**
  - 7         **if**  $(B_2[i] == 1)$  **then**
  - 8              $B_2[i] = 0$ ;  $T++$ ;  $B_1[i] = 1$ ;
  - 9             **if**  $(T == M)$  **then** goto 10;
  - 10 Construct the result AIS  $G$  from the  $O_R$  stream using the filtering condition: the  $i$ th element of  $O_R$  is in  $G$  if  $B_1[i]$  is 1;
  - 11 **return**  $G$  as an AIS
- 

We use a simple yet effective approach for the massUnion operation: read in all the sets one by one using the  $\text{getObjects}(s, R)$  operation and use a bit vector to track the union result. For convenience, we use sequence numbers to identify the objects in the position list. Suppose the set of all distinct objects of relation  $R$  is  $O_R = \{o \mid \exists s : (s, R, o)\}$  and  $N = |O_R|$ . Sorting the set  $O_R$  on object IDs ascendingly gets a list of objects  $o_1, o_2, \dots, o_N$ . The sequence number of object  $o_i$  is  $i$  under relation  $R$ . We thus can allocate a limited size bit vector to track which object is in the result of the relation expansion. The algorithm is called Bit Vector Intersection (BVI) and shown in Algorithm 3.

Note that  $N = |O_R|$  in line 1 can be directly obtained from inverted index as the document frequency of the term  $R$  without any computation at run time. Line 4 and 5 can be implemented together on the PosIdx index very efficiently using a sequential scan on the position list during merge-sort, with the help of self-indexing [14].

The worst case time complexity of the algorithm is linear to the number of all objects of valid subjects but it will stop when all the possible results have been found in line 9. Meanwhile, because of the stop condition, the execution time of the algorithm won't increase definitely with the size of the valid subjects  $S = m(S_1, \cap, (\text{subjOf}, R))$ . In our experiments, we are surprised to find that the time even decreases when the size of valid subjects  $|S|$  exceeds an threshold. This is because in one disk I/O more  $O_s$  sets (i.e, the shaded blocks in Fig. 3) can be read in due to the increased locality of these sets when  $|S|$  becomes large. It shows the benefit of the PosIdx index structure in which relation objects enjoy the spatial locality for fast access.

The space requirement of Algorithm 3 is linear to  $N$  which may be quite large. But in practice, a 256MB memory can already hold the two bit vectors for  $N$  as large as 1 billion and the memory can be reused across multiple executions of the algorithm.

### 3.3 Implementation

We implemented all the above index and query evaluation algorithms in the Semplore engine. It uses the popular open source IR engine Lucene to do classic inverted indexing and perform all the basic IR operations. V2.0.0 Java version of Lucene is used which implements byte-aligned index compression and self-indexing. We use the COLT package<sup>3</sup> for fast bit vector operations in Semplore.

## 4 Experiment and Application

### 4.1 Experiment Setup

Both synthetic and real world semantic web data are used in our experiment. We use LUBM [18] benchmark data sets, from LUBM(5,0) to LUBM(400,0), to test Semplore's scalability. To evaluate Semplore's performance on hybrid queries, we use Wikipedia content as the real world knowledge base because it contains both rich textual properties and relationships between entities. We combine TBox from YAGO [19] and ABox from DBpedia [20] as our Wikipedia dataset. Some simple heuristic rules are used for data cleaning.

All the data sets are preprocessed by Minerva [7] to do reasoning. IBM DB2 v8.1.7 is used as the backend database of Minerva. Table 2 shows the number of triples of each data set after reasoning. We then index the inferred data set in Semplore, by extracting the triples from Minerva. The index time of Semplore excludes the time of extracting triples from Minerva. Table 2 shows Semplore's index performance w.r.t different data sets using a single index thread. We can see that both the index time and space of Semplore increase basically linearly with the size of dataset. The following query evaluation experiments are carried out on these indices. For Sesame [4], we used the 1.2.6 version with MySQL 5.0.21 as its backend RDBMS. All the experiments are conducted on a normal

<sup>3</sup> <http://dsd.lbl.gov/~hoschek/colt/>

**Table 2.** Datasets and Index Performance

Dataset	Triples After Reasoning	Index Time(s)	Index Space(MB)
LUBM(5,0)	729,253	198	25.6
LUBM(10,0)	1,485,029	1379	52
LUBM(20,0)	3,135,033	2712	110
LUBM(50,0)	7,763,131	9699	272
LUBM(400,0)	62,233,512	49783	2150
Wikipedia	13,991,407	25873	1320

**Table 3.** Query Response Time for LUBM Datasets (ms) ('Sem': Semplere, 'Min': Minerva, 'Ses': Sesame; As in [21], we omit Sesame's performance on LUBM(50) and LUBM(400) due to its excessive loading time.)

Query	LUBM(5)			LUBM(10)			LUBM(20)			LUBM(50)		LUBM(400)	
	Sem	Min	Ses	Sem	Min	Ses	Sem	Min	Ses	Sem	Min	Sem	Min
BQ1	0	203	47	0	359	16	16	703	16	16	1407	94	10890
BQ3	0	156	47	0	203	31	0	360	31	0	719	0	5391
BQ4'	0	47	0	0	31	0	0	47	15	0	63	0	297
BQ5	0	156	218	0	156	250	0	312	204	0	578	16	4235
BQ6	0	109	1000	0	234	1953	16	484	2968	32	1000	203	7750
BQ7'	0	266	47	0	375	32	0	765	32	0	1813	0	13750
BQ8'	0	203	2781	0	234	4468	0	1078	8874	16	2609	0	19688
BQ10	0	250	0	0	344	0	0	734	32	0	1406	0	10750
BQ11	0	47	94	0	47	110	0	47	109	0	62	0	266
BQ12'	16	62	125	0	78	703	0	109	2625	0	218	15	1313
BQ13	0	94	16	0	31	15	0	32	78	15	31	47	63
BQ14	0	93	578	0	156	1125	0	360	2672	31	797	156	5922
NQ1	47	453	4890	78	1046	12594	156	2016	53468	360	5438	2938	67078
NQ2	31	2625	281	78	4688	609	172	7094	532	406	17922	3297	523815

desktop PC with Pentium 4 CPU of 3.2 GHz and 2G memory, running Microsoft Windows Server 2003 with Sun Java JRE 1.5.0. The Wikipedia data set and the test queries used in the following experiments are all available from our Semplere web site<sup>4</sup> which also contains live demos.

## 4.2 Query Evaluation

First, we consider the 14 LUBM benchmark queries(BQ) in [18]. Some modifications are applied to the queries due to the unary tree-shaped query capability of Semplere. Queries with multiple target variables (BQ4, BQ7, BQ8, BQ12) are modified to unary ones, in which we all choose  $x$  as target variable. We also remove two cyclic queries (BQ2, BQ9). The remaining 12 queries are all path queries and primarily designed to test reasoning capability. We therefore add two new tree-shaped queries (NQ1, NQ2) to test Semplere.

<sup>4</sup> [http://apex.sjtu.edu.cn/apex\\_wiki/Demos/Semplere](http://apex.sjtu.edu.cn/apex_wiki/Demos/Semplere)

**Table 4.** Hybrid Queries and Their Response Time on Wikipedia Dataset (ms)

Query Set	Set Size	Pattern	Example Query	Ave Time	Max Time
QS1 (TBox)	10	node	Find all concepts with label containing the term “chinese”	8	16
QS2 (ABox)	10	node	Find documentary films about “world war”	7	16
QS3 (ABox)	20	path ave_len:2.3 max_len:3	Find films reaching “Academy Award” and starring a “James Bond” actor; Find artists originating from New York City and having “hip hop” albums	11	94
QS4 (ABox)	10	tree ave_dep:2.2 max_dep:3	Find directors who have directed “romantic” films starring Best Actor Academy Award Winners and Best Actress Academy Award Winners and also films about “war”	25	101

We compare the query evaluation performance on pure structured queries of Semplore with the DB-based ontology stores Minerva and Sesame. Table 3 shows the response time of the three systems for all the 14 queries. The query time includes traversing the whole result set of each query. All the three systems have inferred data materialized and do not have run-time reasoning. Thanks to the benefit of spatial locality for fast access using PosIdx index, Semplore achieves very good scalability on all the 14 queries. Its maximal query processing time of the 12 BQs is less than 0.25 second even for the largest LUBM(400) data set, which is much better than that of Minerva. Considering the two NQs, although Semplore needs more relation expansion operations, it manages to return answers within 4 seconds and still keeps orders of magnitude faster than Minerva and Sesame. One important reason is that while Minerva and Sesame depends on complex nested table joins, Semplore turns to relation expansions along the edges of query graph, which are more lightweight operations. This advantage comes from Semplore’s designed trade-off between query capability and scalability. Certainly, we note that Minerva is well-designed to deal with reasoning, whereas Semplore focuses more on indexing and querying. The comparison here is somewhat unfair on this aspect, but it certainly shows that IR-based approach is also promising for querying semantic web data.

What we test above is on queries without keyword searches. Since there are no benchmarks for hybrid queries yet, we created four sets of queries for the Wikipedia dataset with increasing complexity in query patterns, from node, path to tree. The queries include 10 TBox queries and 40 ABox queries. All the queries contain one or more keyword searches to express vague information needs. The details of the hybrid queries are shown in Table 4.

The table also summarizes the response time of Semplore w.r.t the four query sets. The query time includes traversing the whole result set of each query. By providing sub-second query respond time on the 1.3GB Wikipedia data set index, Semplore provides good scalability on hybrid queries as well. This mainly benefits from its unified index method for both semantic information and textual information.

### 4.3 Applications

In this section, we very briefly describe two applications of the Semplere engine. The first one is a search application for the Wikipedia data set we used in the experiments. To provide both structured queries and keyword searches in the user interface (UI), we adopted and extended the faceted search paradigm [22]. Users can mix browsing, querying, searching and discovering in the search application easily. User's actions in the UI are translated to hybrid queries to the Semplere engine. Our users reported that it improves access to the Wikipedia's rich structural and textual information. The application can be accessed from our Semplere web site.

In another application, we exploit the use of semantic web technologies for product information management (PIM) as reported in [23]. Modern enterprises have very complex product information that contains a large variety of categories, attributes, relationships and rich textual descriptions. Hybrid query capability and easy-to-use UI is needed for the PIM user to browse, query and search all the products. We modeled a real IBM customer's product information in OWL and converted a subset of the real data into RDF. We then use the Semplere engine and the extended faceted search developed above to help user browse, query and search the product information. Semplere successfully delivered answers to hybrid queries with sub-second performance on the 2GB index of the RDF data set. It thus augments the semantic PIM system in [23] with a powerful search engine.

## 5 Related Work

Existing work on querying and searching semantic web data can be roughly divided into two categories: IR-based and DB-based.

Swoogle [3] is a crawler-based indexing and retrieval system for the Semantic Web. It uses IR engine to index the crawled semantic web documents using either n-gram or URIs as terms and computes a ranked list of these documents given the search terms. [1] augments keyword search with semantic information collected from different sources while [2] uses keyword search results as a seed to do spread activation on semantic networks. These work takes a pure IR approach and does not support structured queries on the semantic web data. [24] combines keyword searches and structured queries but focuses on result ranking.

Most DB-based work such as [4,5,6,7] rely on relational database engines for indexing and querying. The primary database schema used for storing triples is a vertical schema with various optimizations. Query answering is achieved primarily through self-join on the vertical table. Indices are built on combinations of subjects, predicates and objects. These DB-based approaches have no or weak support to keyword searches. YARS [25] does not rely on a relational database but uses similar index structures (i.e. B+ trees). Both YARS and Kowari<sup>5</sup> support keyword searches but they use IR engine separately for that purpose only.

<sup>5</sup> <http://www.kowari.org>

BigOWLIM<sup>6</sup> is a scalable repository supporting structured queries but uses its own proprietary storage and index format. LUBM [18] benchmark are developed alongside those work to evaluate semantic web knowledge base systems [21].

Our work is certainly related to the cross field of DB and IR. Actually, the integration of IR and DB is a long desired research goal in the information management area. IR methods have been successfully borrowed to DB area for many tasks such as keyword searches (e.g. [26]) and searching XML (e.g. [27]). At the same time, work in IR area is also adding more structures and semantics to the keyword search. [28] borrows XML Fragment query language to express more semantic-rich queries for searching a semantically annotated text corpora. These work does not address the hybrid query capability for semantic web data (i.e. the RDF triples) but most of them supports ranking of results which is currently lacking in our method.

## 6 Conclusion

Having hybrid query capability and scalable query evaluation algorithm is important in querying semantic web data. In this paper, we define such a hybrid query capability that combines structured queries with keyword searches. Unlike most of the current work that uses DB engines, our work uses existing IR index structure and engine to support the hybrid query capability. Our work shows that this IR approach not only is possible but also achieves good scalability due to its trade-off on query capability and reuse of existing IR optimizations. It leads us to believe that it may be possible to evolve current web search engine's powerful backend IR infrastructure for querying and searching the Semantic Web and ultimately evolve them to web-scale semantic web search engines. Our future work includes the evaluation of the effectiveness of the hybrid query capability and the support of ranking of results.

## References

1. Guha, R., McCool, R., Miller, E.: Semantic search. In: Proc. of the 12th Intl. Conf. on World Wide Web (2003)
2. Rocha, C., Schwabe, D., Aragao, M.P.: A hybrid approach for searching in the semantic web. In: Proc. of the 13th Intl. Conf. on World Wide Web (2004)
3. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V., Sachs, J.: Swoogle: a search and metadata engine for the semantic web. In: Proc. of the 13th ACM CIKM Conf., ACM Press, New York (2004)
4. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A generic architecture for storing and querying RDF and RDF Schema. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, Springer, Heidelberg (2002)
5. Pan, Z., Heflin, J.: DLDB: Extending relational databases to support semantic web queries. In: Workshop on Practical and Scalable Semantic Systems (2003)
6. Chong, E.I., Das, S., Eadon, G., Srinivasan, J.: An efficient SQL-based RDF querying scheme. In: Proc. of the VLDB 2005 (2005)

---

<sup>6</sup> <http://www.ontotext.com/owlim/big/>



7. Zhou, J., Ma, L., Liu, Q., Zhang, L., Yu, Y., Pan, Y.: Minerva: A scalable OWL ontology storage and inference system. In: Mizoguchi, R., Shi, Z., Giunchiglia, F. (eds.) ASWC 2006. LNCS, vol. 4185, Springer, Heidelberg (2006)
8. Zhang, L., Yu, Y., Zhou, J., Lin, C., Yang, Y.: An enhanced model for searching in semantic portals. In: Proc. of the 14th Intl. World Wide Web Conf. (2005)
9. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. *Computer Networks* 30(1-7) (1998)
10. Horrocks, I., Tessaris, S.: Querying the semantic web: A formal approach. In: International Semantic Web Conference, 177–191 (2002)
11. Siberański, W., Pan, J.Z., Thaden, U.: Querying the semantic web with preferences. In: International Semantic Web Conference (2006)
12. Christophides, V., Karvounarakis, G., Plexousakis, D.: Optimizing taxonomic semantic web queries using labeling schemes. *Journal of Web Semantics* 1(2) (2004)
13. Scholer, F., Williams, H.E., Yiannis, J., Zobel, J.: Compression of inverted indexes For fast query evaluation. In: Proc. of the 25th ACM SIGIR Conf., ACM Press, New York (2002)
14. Moffat, A., Zobel, J.: Self-indexing inverted files for fast text retrieval. *ACM Transaction on Information Systems* 14(4), 349–379 (1996)
15. Melink, S., Raghavan, S., Yang, B., Garcia-Molina, H.: Building a distributed full-text index for the web. *ACM Trans. Inf. Syst.* 19(3) (2001)
16. Dean, J., Ghemawat, S.: MapReduce: Simplified data processing on large clusters. In: Proc. of the 6th Symp. on Operating System Design and Implementation (2004)
17. Cormen, T.H., et al.: Introduction to Algorithms. MIT Press, Cambridge (2001)
18. Guo, Y., Pan, Z., Heflin, J.: LUBM: A benchmark for OWL knowledge base systems. *Journal of Web Semantics* 3(2) (2005)
19. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: A core of semantic knowledge unifying wordnet and wikipedia. In: Proc. of WWW 2007 (2007)
20. Auer, S., Lehmann, J.: What have innsbruck and leipzig in common? extracting semantic from wiki content. In: Proc. of ESWC 2007 (2007)
21. Guo, Y., Pan, Z., Heflin, J.: An evaluation of knowledge base systems for large OWL datasets. In: Proc. of the 3rd Intl. Semantic Web Conf. (2004)
22. Yee, K.P., Swearingen, K., Li, K., Hearst, M.: Faceted metadata for image search and browsing. In: Proc. of CHI 2003 (2003)
23. Brunner, J.S., Ma, L., Wang, C., Zhang, L., Wolfson, D.C., Pan, Y., Srinivas, K.: Explorations in the use of semantic web technologies for product information management. In: Proc. of WWW 2007 (2007)
24. Nejdil, W., Siberański, W., Thaden, U., Balke, W.T.: Top-k query evaluation for schema-based peer-to-peer networks. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, Springer, Heidelberg (2004)
25. Harth, A., Decker, S.: Optimized index structures for querying RDF from the web. In: IEEE Proc. of the 3rd Latin American Web Congress, IEEE Computer Society Press, Los Alamitos (2005)
26. Liu, F., Yu, C., Meng, W., Chowdhury, A.: Effective keyword search in relational databases. In: Proc. of SIGMOD 2006 (2006)
27. Kaushik, R., Krishnamurthy, R., Naughton, J.F., Ramakrishnan, R.: On the integration of structure indexes and inverted lists. In: Proc. of the 2004 ACM SIGMOD Conference, ACM Press, New York (2004)
28. Chu-Carroll, J., Prager, J., Czuba, K., Ferrucci, D., Duboue, P.: Semantic search via XML Fragments: a high-precision approach to IR. In: Proc. of the 29th ACM SIGIR Conference, ACM Press, New York (2006)

# Semantic Cooperation and Knowledge Reuse by Using Autonomous Ontologies

Yuting Zhao<sup>1</sup>, Kewen Wang<sup>1</sup>, Rodney Topor<sup>1</sup>, Jeff Z. Pan<sup>2</sup>, and Fausto Giunchiglia<sup>3</sup>

<sup>1</sup> Griffith University, Australia

<sup>2</sup> University of Aberdeen, UK

<sup>3</sup> University of Trento, Italy

{yuting.zhao, k.wang, r.topor}@griffith.edu.au  
jpan@csd.abdn.ac.uk, fausto@dit.unitn.it

**Abstract.** Several proposals have been put forward to support distributed agent cooperation in the Semantic Web, by allowing concepts and roles in one ontology be reused in another ontology. In general, these proposals reduce the autonomy of each ontology by defining the semantics of the ontology to depend on the semantics of the other ontologies.

We propose a new framework for managing autonomy in a set of cooperating ontologies (or ontology space). In this framework, each language entity (concept/role/individual) in an ontology may have its meaning assigned either locally with respect to the semantics of its own ontology, to preserve the autonomy of the ontology, or globally with respect to the semantics of any neighbouring ontology in which it is defined, thus enabling semantic cooperation between multiple ontologies.

In this way, each ontology has a “subjective semantics” based on local interpretation and a “foreign semantics” based on semantic binding to neighbouring ontologies. We study the properties of these two semantics and describe the conditions under which entailment and satisfiability are preserved. We also introduce two reasoning mechanisms under this framework: “cautious reasoning” and “brave reasoning”. Cautious reasoning is done with respect to a local ontology and its neighbours (those ontologies in which its entities are defined); brave reasoning is done with respect to the transitive closure of this relationship. This framework is independent of ontology languages. As a case study, for Description Logic  $\mathcal{ALCN}$  we present two tableau-based algorithms for performing each form of reasonings and prove their correctness.

## 1 Introduction

The Semantic Web vision is to develop a distributed environment in which software agents can automatically, conveniently and effectively interpret and apply the data that is available on the Web. To this end, a system of knowledge representation which supports semantic cooperation between distributed agents is required. Such a system must be based on ontologies which define the terms and relationships used in a particular application domain. Each such ontology reflects the objective and shared views of a community of users working in that domain. However, the original use of the word

“ontology” in philosophy was to describe a complete, self-contained domain of discourse. This usage does not scale to the open and distributed Web, where there are ontologies for each different application domain and even different ontologies for the same domain. Thus, to support semantic cooperation between agents, it is necessary to manage and reason about multiple ontologies, which we call an ontology space. How to do this effectively is a major research problem for the Semantic Web.

AI researchers have also studied management and reasoning in multiple representations of application domains using contextual reasoning [6,7,5]. For this reason, it is a natural and interesting issue to combine ontology-based and context-based approaches so that the advantages of both ontology and contextual reasoning can be employed in the same system. This idea led a series of interesting work.

C-OWL [4] sets up relations outside the ontologies by a set of “bridge rules” between the concepts (individuals) from different ontologies. Its semantics relies on the *domain relation* in DDL [3], which is a directional mapping from the elements of one domain to the other domain.  $\mathcal{E}$ -Connection [10] puts the relations inside the ontology, by extending OWL with a new kind of “link property”. It connects two sets of strictly disjoint concepts from different domains. P-DL [2] treats every foreign term as an imported relation, and semantically interprets it by an *image domain relation* which is a one-to-one and compositional consistent mapping between two domains. Semantic Importing [12] focuses on the overlaps of domains; it allows a subconcept which falls into the conjunction of two domains to be semantically imported and used in the other ontology. Conservative Extension [9] restricts multiple ontology modules in the same global interpretation domain and allows them be interpreted using standard semantics.

It is easy to see that the above approaches somehow weaken the autonomy of an ontology. In order to bridge the gaps between the semantics of different ontologies, a class of approaches based on cross-domain relations (i.e., domain relations in C-OWL, directional binary relation in  $\mathcal{E}$ -connection, and image domain relation in P-DL) needs the information of the domain element in the other ontology to interpret a bridge rule in C-OWL, a link property in  $\mathcal{E}$ -connection, or an importing relation in P-DL. For example, suppose that there is well-accepted ontology on the Web called “Vehicle”, in which a concept “Car” is defined as: “A car consists of two parts: engine and body”. In C-OWL, this concept is represented as  $Vehicle : Car$  which indicates that the concept “Car” is defined in the ontology “Vehicle”. Both BMW and Toyota may wish to borrow the concept “Car” when they design their own ontologies. These two companies have different interpretations on “engine” and “body”. In C-OWL,  $Vehicle : Car$  has different local interpretations in BMW and Toyota. However, a user cannot distinguish this difference and confusion may be caused when other people use  $Vehicle : Car$ . Semantic Importing does not rely on domain relations, but one needs to know exactly the domain elements and how the concepts/roles are interpreted in the other ontology. Conservative extensions require a single global domain, and prevent each module from local interpretation its axioms under its own context.

In this paper, we propose a new framework for managing autonomy in a distributed ontology space. On the one hand a language entity (concept/role/individual) is interpreted totally under local domain semantics in order to preserve the autonomy of an ontology; on the other hand a (shared) language entity is restricted by a *semantic*

*binding* if necessary in order to enable semantic cooperation among several ontologies. In this way, one ontology is able to express its “subjective” opinion by local interpretation, and to receive its *foreign semantics* by semantic binding. We use the term “foreign semantics” of one ontology to express the “semantic meaning” of an foreign entity from another ontology connected by semantic binding. Accordingly, we also introduce two reasoning mechanisms: *cautious reasoning* and *brave reasoning*. The former relies on an ontology and its foreign semantics from its neighbors’ ontologies connected by semantic binding, but it does not trust the foreign semantics of its neighbour from their neighbors. The latter believes an ontology and its foreign semantics, and also its neighbour ontologies and their foreign semantics.

The main contribution of this paper are the followings:

- We introduce a novel approach to define semantic cooperation between different ontologies. By semantic binding, the semantics of language entities from one ontology is able to be accessed in other ontologies. This is different from the existing domain-relation based approaches.
- We formalize two forms of reasoning mechanisms: cautious reasoning and brave reasoning. The former only does reasoning in one ontology and its neighbors’ ontologies associated by its semantic binding; the latter does reasoning in one ontology and its neighbours’ ontologies and their neighbours’ ontologies and so on.

In the rest, we briefly review ontologies and ontology spaces in Section-2, and introduce the notion of an autonomous ontology in Section-3. In Section-4 we introduce the two reasoning mechanisms and the algorithms. The related work review and conclusion are given in Section-5.

## 2 Preliminaries: Ontology Space and Foreign Entity

In general understanding [14], an ontology is a set of annotated *terminological axioms* and *facts*. Current discussion is based on normal Description Logic (DL) [11]. The proposed framework can be restricted or generalized to some DL languages such as OWL, SHOIN(D+), etc.

### 2.1 Ontology and Ontology Space

Let  $\mathbb{C}$  be a finite set of concept names,  $\mathbb{R}$  a finite set of role names, and  $\mathbb{E}$  a finite sets of individual names. A *language*  $\mathbb{L}$  has a vocabulary of the disjoint union of  $\mathbb{C}$ ,  $\mathbb{R}$  and  $\mathbb{E}$ .

**Definition 1 (Ontology).** Let  $\mathbb{L}$  be the language. An Ontology  $\mathcal{O}$  is a tuple  $\langle T, A \rangle$ , where  $T$  and  $A$  are TBox and ABox respectively in Description Logic on  $\mathbb{L}$ .

**Definition 2 (Ontology interpretation).** An (abstract) ontology interpretation  $\mathcal{I}$  is a pair  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ , in where  $\Delta^{\mathcal{I}}$  is an nonempty domain, and  $\cdot^{\mathcal{I}}$  is a mapping that assigns

1. to each concept name  $c \in \mathbb{C}$  a subset of  $\Delta^{\mathcal{I}}$ ,
2. to each role name  $R \in \mathbb{R}$  a subset of  $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ ,
3. to each individual name  $e \in \mathbb{E}$  an element of  $\Delta^{\mathcal{I}}$ .

**Definition 3 (Ontology Space).** Let  $I$  be a set of indexes, standing for a set of URIs for ontologies. Let  $\mathbb{L}_I = \{\mathbb{L}_i\}_{i \in I}$  be a set of languages. An Ontology Space  $\mathbb{O}_I$  on  $\mathbb{L}_I$  is a family  $\{O_i\}_{i \in I}$ , s.t. every  $O_i$  is an ontology on language  $\mathbb{L}_i$ , where  $i \in I$ .

In ontology space  $\mathbb{O}_I = \{O_i\}_{i \in I}$ , we denote, by  $\mathbb{C}_i$  the set of concept names in ontology  $O_i$ . Analogous to  $\mathbb{R}_i$  and  $\mathbb{E}_i$ . Actually language  $\mathbb{L}_i$  has a vocabulary of the disjoint union of  $\mathbb{C}_i$ ,  $\mathbb{R}_i$  and  $\mathbb{E}_i$ . In the rest of the paper, we use *language entity* to denote concept, role, or individual in one ontology.

## 2.2 Foreign Entity

In ontology space, sometimes a language entity is defined in one ontology, but could be used in another ontology. So we partition the language  $\mathbb{L}_i$  in two parts: the *local entity* and the *foreign entity* (originated from local language and foreign language in [4]). Intuitively, local entities are the roles, concepts, and individuals that one invites in her own ontology; foreign entities are the roles, concepts, and individuals that she borrows from the other ontologies in order to define something in her ontology.

In this paper, when we are talking about semantics and reasoning, we always tell a language entity in the ontology space by a way showing (1) where it is used, and (2) where it is originally defined. Suppose that  $C \in \mathbb{L}_i$  and  $i, j \in I$ , then formally in ontology space we have a language entity like,  $(i : j : C)$ , which means a language entity  $C$  appears in ontology  $O_i$ , but is originally defined in ontology  $O_j$ . This kind of denotation is applied to concepts, roles, and individuals in ontology space.

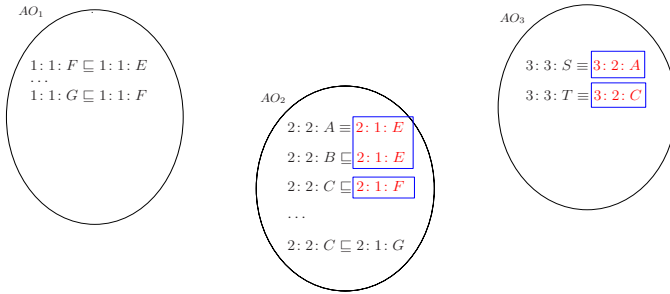
One of the advantages of this denotation is, in syntax two concepts/roles with the same name but used in different ontologies are distinguishable. For example, suppose we use  $(BMW : auto : engine)$  for the engines in BMW car ontology, and  $(Toyota : auto : engine)$  for those in Toyota car ontology; obvious these two concepts should be different, and it is easy to see from the syntax:  $(BMW : auto : engine) \neq (Toyota : auto : engine)$ . This denotation is important in this paper because, the approach proposed in this paper assigns local semantics to this kind of concepts/roles, and treats them as totally different entities.

## 3 Autonomous Ontology

In one ontology space, each ontology reflects the subjective opinion on a partial structure of the universe. In Semantic Web, in general one party presents her personal knowledge (understanding) by her ontology. Thereafter we argue each ontology should be semantical independent and keeping autonomy.

Formally an *autonomous ontology* is composed by two parts: one is an ontology which is to be interpreted locally in order to keep the autonomy of one party; the other is a set of foreign entities, which is called *semantic binding* in this paper, in order to accept foreign information or knowledge from the other parties.

**Definition 4 (Autonomous Ontology).** Let  $\mathbb{O}_I = \{O_i\}_{i \in I}$  be an ontologiespace, where  $I$  is a set of indexes. An autonomous ontology is a tuple  $AO_i = \langle \mathbb{B}_i, O_i \rangle$ , in which  $O_i \in \mathbb{O}_I$ , and  $\mathbb{B}_i$  is the semantic binding of  $AO_i$ , which is a set of foreign entities.



**Fig. 1.** Autonomous ontology space

An *autonomous ontology space*  $\mathbb{A}\mathbb{O}_I$  is a set of autonomous ontologies. In an autonomous ontology space  $\mathbb{A}\mathbb{O}_I = \{AO_i\}_{i \in I}$ , if an entity  $(j:j:x)$ , which is original defined in  $AO_j$ , is semantic bounded in  $AO_i$ ; i.e.,  $(i:j:x) \in \mathbb{B}_i$ , then we say  $AO_j$  is the *binding neighbour* of  $AO_i$ .

In fact it is easy to treat an autonomous ontology as a normal ontology; for example in OWL we could just introduce a binding annotation like:  
 Annotation ( binding <http://www.auto.org/engine#> )  
 to express the semantic binding. So in the rest of the paper sometime we also mean an autonomous ontology by ontology.

*Example 1 (Autonomous Ontology).* Consider the autonomous ontology space in Figure-1, suppose we have following semantic bindings:  $\mathbb{B}_1 = \emptyset$ ,  $\mathbb{B}_2 = \{(2:1:E), (2:1:F)\}$ , and  $\mathbb{B}_3 = \{(3:2:A), (3:2:C)\}$ . So  $AO_1$  is the binding neighbour of  $AO_2$ , and  $AO_2$  is the binding neighbour of  $AO_3$ . We note that in autonomous ontology it allows some foreign entity outside of the semantic binding, e.g.,  $(2:1:G)$  in  $AO_2$ .  $\square$

### 3.1 Local Interpretation

**Definition 5 (Local Interpretation).** For autonomous ontology  $AO_i = \langle \mathbb{B}_i, O_i \rangle$ , a local interpretation  $\mathcal{I}_i$  is a pair  $\langle \Delta^{\mathcal{I}_i}, \cdot^{\mathcal{I}_i} \rangle$ , in where  $\Delta^{\mathcal{I}_i}$  is an nonempty domain, and  $\cdot^{\mathcal{I}_i}$  is a mapping, s.t.  $\cdot^{\mathcal{I}_i}$  assigns

1. (for local entities)
  - (a) a subset of  $\Delta^{\mathcal{I}_i}$  to each local concept name  $(i:i:c)$ ;
  - (b) a subset of  $\Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_i}$  to each local role name  $(i:i:r)$ ;
  - (c) an element of  $\Delta^{\mathcal{I}_i}$  to each local individual name  $(i:i:e)$ ,
2. and for  $i \neq j$  (for foreign entities)
  - (a) a subset of  $\Delta^{\mathcal{I}_i}$  to each foreign concept name  $(i:j:c)$ ;
  - (b) a subset of  $\Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_i}$  to each foreign role name  $(i:j:r)$ ;
  - (c) an element of  $\Delta^{\mathcal{I}_i}$  to each foreign individual name  $(i:j:e)$ .

As we see in the above definition, in local interpretation not only local entities but also foreign entities including those in the semantic binding are interpreted under local domain. Actually, the local interpretation of an autonomous ontology is the interpretation of the ontology. From this aspect the abstract interpretation of ontology in Section-2 is

also a local interpretation. The difference is, the definition in the last section is for a single ontology, but here we focus on one ontology among an ontology space.

Following common understanding, if an axiom  $a \in A$  is true under an interpretation  $\mathcal{I}$ , we say that interpretation  $\mathcal{I}$  *satisfies* the axiom  $a$ , and denote this by  $\mathcal{I} \models a$ .

**Definition 6 (Satisfiability of autonomous ontology).** Let  $AO_i$  be an autonomous ontology and an  $\mathcal{I}$  its local interpretation, we say that  $\mathcal{I}$  satisfies  $AO_i$ , if for any axiom  $a \in AO_i$ , we have  $\mathcal{I} \models a$ . We call  $\mathcal{I}$  a local model of  $AO_i$ , and denote this fact by  $\mathcal{I} \models_{\mathcal{L}} AO_i$ .

**Definition 7 (Local Entailment).** Let  $AO_i$  be an autonomous ontology,  $\lambda$  a concept description or an assertion. We say that  $\lambda$  is a local entailment of  $AO_i$ , iff for any local model  $\mathcal{I}_i$  of  $AO_i$ ,  $\mathcal{I}_i \models \lambda$ . This fact is denoted by  $AO_i \models_{\mathcal{L}} \lambda$ .

*Example 2 (Local semantics).* Consider the autonomous ontology space in Figure-1. We have  $AO_2 \models_{\mathcal{L}} (2:2:B) \sqsubseteq (2:2:A)$ , because for any local model  $\mathcal{I}$ ,

$$\frac{\begin{array}{l} \mathcal{I} \models (2:2:B) \sqsubseteq (2:1:E), \quad \text{and} \\ \mathcal{I} \models (2:2:A) \equiv (2:1:E) \end{array}}{\mathcal{I} \models (2:2:B) \sqsubseteq (2:2:A)}.$$

### 3.2 C-Binding Consistency

For autonomous ontology  $AO_i \in \mathbb{A}_{O_I}$ , let  $\mathbb{B}_i = \{\bigcup_{j \in I} \mathbb{B}_{ij}\}$  be the semantic binding [1], in which  $\mathbb{B}_{ij} = \{(i:j:C)\}_{i,j \in I}$  contains all of the semantic-bounded foreign entities which are original defined in  $O_j$ .

In autonomous ontology  $AO_i \in \mathbb{A}_{O_I}$ , a  $j$ -concept ( $j$ -role) is an class (property) description which is composed by the entities in  $\mathbb{B}_{ij}$ . For example, a  $j$ -concept of  $AO_i$  could be  $\exists(i:j:hasChild).(i:j:Male)$ .

Let  $\lambda_j$  be a  $j$ -concept, obviously in  $\lambda_j$  all of the entities are prefixed by “ $i:j:$ ”. If we change the prefix of every entity in  $\lambda_j$  from “ $i:j:$ ” to “ $j:j:$ ”, and then we get  $\lambda'_j$ . We call  $\lambda'_j$  the *original image* of  $\lambda_j$  in  $O_j$ . For example, let  $\lambda_j$  be  $(i:j:Person) \sqcap \forall(i:j:hasChild).(i:j:Female)$ , then its original image  $\lambda'_j$  is  $(j:j:Person) \sqcap \forall(j:j:hasChild).(j:j:Female)$ .

We note that the original image  $\lambda'_j$  of a  $j$ -concept  $\lambda_j$  may not has a concept name in  $O_j$ ; it may not be explicitly defined there.

**Definition 8 (C(j)-binding Consistent Model).** Let  $AO_i = \langle \mathbb{B}_i, O_i \rangle$  be an autonomous ontology, and  $j \in I$ . Let  $\mathcal{I}$  be a local model of  $AO_i$ , if for any  $j$ -concept  $\lambda_j$ , we have

$$\mathcal{I} \models \lambda_j \quad \text{iff} \quad AO_j \models_{\mathcal{L}} \lambda'_j \quad (1)$$

then we say  $\mathcal{I}$  is the C(j)-binding consistent model of  $AO_i$ . This fact is denoted by  $\mathcal{I} \models_{c(j)} AO_i$

*Example 3 (C(j)-binding consistent model).* Let  $\Delta_2 = \{\alpha, \beta\}$  be the domain of  $AO_2$  in Figure-1. Considering following two local interpretations of  $AO_2$ :

<sup>1</sup> It is possible that some foreign entities are not included in  $\mathbb{B}_i$ . Actually these foreign entities are not semantic-bounded; they are “free-access” entities according to [17].



**Table 1.** Example of interpretations for the autonomous ontology  $AO_2$

symbol	$\mathcal{I}_{21}$	$\mathcal{I}_{22}$
$\top$	$\{\alpha, \beta\}$	$\{\alpha, \beta\}$
$(2:2:A)$	$\{\alpha\}$	$\{\alpha, \beta\}$
$(2:2:B)$	$\{\alpha\}$	$\{\alpha\}$
$(2:2:C)$	$\{\alpha, \beta\}$	$\{\beta\}$
$(2:1:E)$	$\{\alpha\}$	$\{\alpha, \beta\}$
$(2:1:F)$	$\{\alpha, \beta\}$	$\{\beta\}$
$(2:1:G)$	$\{\alpha, \beta\}$	$\{\alpha, \beta\}$
...	...	...
$(2:1:F) \sqsubseteq (2:1:E)$	<i>Not satisfiable</i>	<i>Satisfiable</i>

Obviously both  $\mathcal{I}_{21}$  and  $\mathcal{I}_{22}$  are local models of  $AO_2$ , i.e.,  $\mathcal{I}_{21} \models_{\mathcal{L}} AO_2$  and  $\mathcal{I}_{22} \models_{\mathcal{L}} AO_2$ . Let  $\lambda = (2:1:F) \sqsubseteq (2:1:E)$ , from Table-1 we find  $\mathcal{I}_{21} \not\models \lambda$ . Since  $\mathbb{B}_{21} = \{(2:1:E), (2:1:F)\}$ ,  $\lambda$  is the  $I$ -concept, and  $AO_1 \models_{\mathcal{L}} \lambda$ , we have  $\mathcal{I}_{21}$  is not a  $C(I)$ -binding consistent model. Actually  $\mathcal{I}_{22}$  is a  $C(I)$ -binding consistent model of  $AO_2$ . In this example we also note that foreign entity outside of the semantic binding does not carry any semantic information from its original ontology; e.g., since  $(2:1:G) \notin \mathbb{B}_{21}$ , although  $AO_1 \models_{\mathcal{L}} (1:1:G) \sqsubseteq (1:1:F)$ , in  $AO_2$   $C(I)$ -binding consistent model  $\mathcal{I}_{22}$  does not need to satisfy  $(2:1:G) \sqsubseteq (2:1:F)$ .  $\square$

**Definition 9 (C(j)-binding Entailment).** Let  $AO_i$  be an autonomous ontology,  $\lambda$  a concept description<sup>2</sup> or an assertion. We say that  $\lambda$  is the  $C(j)$ -binding entailment of  $AO_i$ , iff for any  $C(j)$ -binding consistent model  $\mathcal{I}$  of  $AO_i$ ,  $\mathcal{I} \models \lambda$ . This fact is denoted by  $AO_i \models_{C(j)} \lambda$ .

In Example-3 we have  $AO_2 \models_{C(I)} (2:1:F) \sqsubseteq (2:1:E)$ , and  $AO_2 \models_{C(I)} (2:2:C) \sqsubseteq (2:2:A)$ .

**Definition 10 (C-binding Consistency).** Let  $\mathcal{I}$  be a local model of  $AO_i$ . If for any  $j \neq i \in I$ ,  $\mathcal{I}$  is the  $C(j)$ -binding consistent model of  $AO_i$ , then  $\mathcal{I}$  is the  $C$ -binding consistent model of  $AO_i$ . We say  $AO_i$  is  $C$ -binding consistent in  $\mathbb{A}\mathbb{O}_I$  if there exists a  $C$ -binding consistent model.

We say an autonomous ontology  $AO_i = \langle \mathbb{B}_i, O_i \rangle$  is  $C$ -satisfiable, if it is  $C$ -binding consistent and  $O_i$  is satisfiable.

Let  $AO_i$  be an autonomous ontology and  $i \neq j \in I$ , we note that not every local model is a  $C(j)$ -binding consistent model; it is not necessary for a local model to satisfy a  $j$ -concept, but it is for a  $C(j)$ -binding consistent model. We also note that not every  $C(j)$ -binding consistent model is a  $C$ -binding consistent model; a  $C(j)$ -binding consistent model may not satisfies a  $k$ -concept for  $k \neq j$ . So an autonomous ontology which is satisfiable under the local semantics could be unsatisfiable under the autonomous semantic.

<sup>2</sup> Here for convenient we treat a subsumption  $x \sqsubseteq y$  as a concept description  $\neg x \sqcup y$ . Same for the rest of the paper.



**Definition 11 (C-entailment).** Let  $AO_i$  be an autonomous ontology,  $\lambda$  a concept description or an assertion. We say that  $\lambda$  is the C-entailment of  $AO_i$ , iff for any C-binding consistent model  $\mathcal{I}_i$  of  $AO_i$ ,  $\mathcal{I}_i \models \lambda$ . This fact is denoted by  $AO_i \models_C \lambda$ .

**Lemma 1.** Let  $AO_i$  be an autonomous ontology,  $\lambda$  a concept description or an assertion. We have  $AO_i \models_{\mathcal{L}} \lambda \implies AO_i \models_C \lambda$ .

In this paper, we use  $\Pi_C(AO_i) = \{\lambda \mid AO_i \models_C \lambda\}$  to denote the set of C-entailments of  $AO_i$ , and call it *Cautious theory* of  $AO_i$ . Comparably, we also use  $\Pi_{\mathcal{L}}(AO_i) = \{\lambda \mid AO_i \models_{\mathcal{L}} \lambda\}$  to denote the set of local entailment of  $AO_i$  under local semantics, and call it *local theory* of  $AO_i$ .

**Theorem 1.** For a autonomous ontology  $AO_i$ , we have  $\Pi_{\mathcal{L}}(AO_i) \subseteq \Pi_C(AO_i)$ .

**Proposition 1.** Let  $AO_i = \langle \mathbb{B}_i, O_i \rangle$  be C-binding consistent, for any  $j \neq i \in I$  and any j-concept  $\lambda$  of  $AO_i$ , we have  $AO_i \models_C \lambda$  if  $AO_j \models_{\mathcal{L}} \lambda'$ .

*Example 4.* In the autonomous ontology space in Figure-□ we have  $AO_2 \models_C (2:2:C) \sqsubseteq (2:2:A)$ . However for  $AO_3$ , since  $AO_2 \not\models_{\mathcal{L}} (2:2:C) \sqsubseteq (2:2:A)$ ,  $AO_3 \not\models_C (3:2:C) \sqsubseteq (3:2:A)$ , and then we do not have  $AO_3$  entails  $(3:3:T) \sqsubseteq (3:3:S)$  under C-entailment. □

In this paper when we say *cautious semantics* of an autonomous ontology we mean the C-binding model(s). We say an autonomous ontology space  $\mathbb{A}\mathbb{O}_I$  is *C-binding consistent* if every autonomous ontology is C-binding consistent.

**Definition 12 (C-entailment of Ontology Space).** Suppose autonomous ontology space  $\mathbb{A}\mathbb{O}_I = \{AO_i\}_{i \in I}$  is C-binding consistent. Let  $AO_i \in \mathbb{A}\mathbb{O}_I$ , and  $\lambda$  a concept description or an assertion. We say that  $\lambda$  is the C-entailment of autonomous ontology space  $\mathbb{A}\mathbb{O}_I$ , denoted by  $\mathbb{A}\mathbb{O}_I \models_C \lambda$ , iff there exists  $i \in I$ , s.t.  $AO_i \models_C \lambda$ . We also say  $i \in I$  is the provenance of the entailment  $\lambda$ .

### 3.3 B-Binding Consistency

*B-binding* stands for “brave binding”. One autonomous ontology not only relies on its binding neighbors, it also trusts the neighbors of its binding neighbors. In this way B-binding could build more stronger semantic cooperation among multiple ontologies than C-binding, in the sense that some information in one ontology is transitively reused by, not only its neighbour but also the neighbour’s neighbour.

For example in the autonomous ontology space in Figure-□  $AO_3$  does not entail  $(3:3:T) \sqsubseteq (3:3:S)$  under C-binding semantics because  $AO_2 \not\models_{\mathcal{L}} (2:2:C) \sqsubseteq (2:2:A)$ ; but it could entail this subsumption under B-binding semantics because  $AO_2 \models_C (2:2:C) \sqsubseteq (2:2:A)$ . Details will be given later.

**Definition 13 (B-binding Entailment).** Let  $AO_i = \langle \mathbb{B}_i, O_i \rangle$  be an autonomous ontology, and  $\mathcal{I}$  a local interpretation of it. We say  $\mathcal{I}$  is a B-binding consistent model of  $AO_i$ , which is denoted by  $\mathcal{I} \models_B AO_i$ , if

- 1  $\mathcal{I}$  is a local model of  $AO_i$ , and
- 2 for any  $j \in I$  and any  $j$ -concept  $\lambda_j$  we have  $\mathcal{I} \models \lambda_j$  iff
  - (a)  $AO_j \models_{\mathcal{L}} \lambda'_j$ , or
  - (b)  $AO_j \models_{\mathcal{C}} \lambda'_j$ , or
  - (c)  $AO_j \models_{\mathcal{B}} \lambda'_j$

We say  $AO_i$  is **B-binding consistent** in  $\mathbb{A}\mathbb{O}_I$  if there exists a **B-binding consistent model**. Let  $\psi$  be a formula. For any **B-binding model**  $\mathcal{I}$  of  $AO_i$ , if  $\mathcal{I} \models \psi$ , then we say  $\psi$  is the **B-entailment** of  $AO_i$ . This fact is denoted by  $AO_i \models_{\mathcal{B}} \psi$

Above is a recursive definition on the **B-binding semantics** for an autonomous ontology. Local knowledge in one ontology is used to support a logical result in its *binding neighbour-reached* ontology. We say  $AO_j$  is **binding neighbour-reached** from  $AO_i$ , if there exists a sequence  $(AO_{x,1}, AO_{x,2}, \dots, AO_{x,k})$  such that for  $1 \leq y < k$   $AO_{x,(y+1)}$  is a **binding neighbour** of  $AO_{x,y}$  and  $AO_{x,1} = AO_i$  and  $AO_{x,k} = AO_j$ .

From above definition it is easy to get the following lemma.

**Lemma 2.** For a autonomous ontology  $AO_i \in \mathbb{A}\mathbb{O}_I$ , every **B-binding model** is a **C-binding model**.

**Lemma 3.** Let  $AO_i$  be an autonomous ontology,  $\lambda$  a concept description or an assertion. We have  $AO_i \models_{\mathcal{C}} \lambda \implies AO_i \models_{\mathcal{B}} \lambda$ .

In this paper, we use  $\Pi_{\mathcal{B}}(AO_i) = \{\lambda \mid AO_i \models_{\mathcal{B}} \lambda\}$  to denote the set of **B-entailments** of  $AO_i$ , and call it *brave theory* of  $AO_i$ . From Lemma-2 and Lemma-3, the following theorem is obvious.

**Theorem 2.** For a autonomous ontology  $AO_i$ , we have  $\Pi_{\mathcal{C}}(AO_i) \subseteq \Pi_{\mathcal{B}}(AO_i)$ .

According to Definition-13 the following proposition is obvious.

**Proposition 2.** Let  $AO_i = \langle \mathbb{B}_i, O_i \rangle$  be **B-binding consistent**, for any  $j \neq i \in I$  and any  $j$ -concept  $\lambda$  of  $AO_i$ , we have  $AO_i \models_{\mathcal{B}} \lambda$  if  $AO_j \models_{\mathcal{C}} \lambda'$ .

*Example 5.* In the autonomous ontology space in Figure-11 we have  $AO_3 \models_{\mathcal{B}} (3:3:T) \sqsubseteq (3:3:S)$  under the brave semantics.

In this paper when we say *brave semantics* of an autonomous ontology we mean the **B-binding model(s)**. We say an autonomous ontology space  $\mathbb{A}\mathbb{O}_I$  is **B-binding consistent** if every autonomous ontology is **B-binding consistent**.

**Definition 14 (B-entailment of Ontology Space).** Suppose autonomous ontology space  $\mathbb{A}\mathbb{O}_I = \{AO_i\}_{i \in I}$  is **B-binding consistent**. Let  $AO_i \in \mathbb{A}\mathbb{O}_I$ , and  $\lambda$  a concept description or an assertion. We say that  $\lambda$  is the **B-entailment** of autonomous ontology space  $\mathbb{A}\mathbb{O}_I$ , denoted by  $\mathbb{A}\mathbb{O}_I \models_{\mathcal{B}} \lambda$ , iff there exists  $i \in I$ , s.t.  $AO_i \models_{\mathcal{B}} \lambda$ . We also say  $i \in I$  is the *provenance of the entailment*  $\lambda$ .

## 4 Tableaux Algorithms of Reasoning on $\mathcal{ALCN}$

In this section, we present two distributed tableaux algorithms to realize *cautious reasoning* (under cautious semantics) and *brave reasoning* (under brave semantics) in autonomous ontology space respectively.

Here we consider ontologies represented as  $\mathcal{ALCN}$  TBoxes (which consists of only class axioms). These algorithms are designed for verifying class satisfiability in an autonomous ontology space  $\mathbb{A}\mathbb{O}_I$ , and can also be used to verify class subsumption<sup>3</sup>.

### 4.1 Preliminary of Tableaux Algorithm

Tableaux algorithms are very useful to solve class satisfiability problem. They test the satisfiability of a class  $\lambda$ <sup>4</sup> by trying to construct an interpretation for  $\lambda$ , which is represented by a *completion tree*  $\mathcal{T}$  which is formally defined as following: A *completion tree* is a tuple  $\mathcal{T} = \langle x_0, N, E, \mathcal{L} \rangle$ , where  $x_0$  is the root of  $\mathcal{T}$ ,  $N$  and  $E$  are the sets of nodes and edges, respectively, of  $\mathcal{T}$ , and  $\mathcal{L}$  is a function that maps a node  $x$  in  $\mathcal{T}$  to its label  $\mathcal{L}(x)$ , and an edge  $\langle x, y \rangle$  in  $\mathcal{T}$  to its  $\mathcal{L}(\langle x, y \rangle)$ , respectively.

A tableaux algorithm starts from an labeled initial tree (usually simply a root node), and is expanded by repeatedly applying the completion rules. The algorithm terminates either when  $\mathcal{T}$  is *complete* (no further completion rules can be applied) or when an obvious contradiction, or *clash*, has been revealed.

Intuitively, our tableaux algorithm expands a completion tree *w.r.t.* the local axiom box, and then project some part of the tree (which is related to other autonomous ontology) for further expansion *w.r.t.* the axiom boxes of the neighbour autonomous ontology, (e.g., sending the original image of a  $j$ -concept to  $AO_j$  and start a new tableaux algorithm to check the satisfiability) and then back-project some semantics results to the local completion tree. We say that a completion tree  $\mathcal{T}$  is  $S(j)$ -bound if there exist some  $j$ -concept or  $j$ -role descriptions in the labels of all nodes and edges of  $\mathcal{T}$ . In this section, we use procedure  $\text{Tab}(AO_i, \mathcal{T})$  as a well known (local)  $\mathcal{ALCN}$  tableaux algorithm to expand  $\mathcal{T}$  *w.r.t.* a local ontology  $AO_i$ .  $\text{Tab}(AO_i, \mathcal{T})$  has two distinguished features that we need: (1) it takes not only a single node but an arbitrary initial completion tree, (2) the algorithm can cache reasoning states, *i.e.*, backtracking points. In the algorithms below we also use  $\text{backtrack}(\text{Tab}(AO_i, \mathcal{T}))$  to denote the operation to expand tree  $\mathcal{T}$  from the backtracking point and return a completion tree.

*Projection* is to bring information from one autonomous ontology to the binding neighbour.

**Definition 15 (Projection of Completion Tree).** Let  $AO_i = \langle \mathbb{B}_i, O_i \rangle$  be an autonomous ontology, and  $\mathbb{B}_{ij} \in \mathbb{B}_i$  the set of semantic binding of foreign entities from another autonomous ontology  $AO_j$ . Let  $\mathcal{T}$  be a clash-free completion tree with root  $x_0$  in  $AO_i$ . The projection of  $\mathcal{T}$  *w.r.t.*  $\mathbb{B}_{ij}$ , denoted as  $P(\mathcal{T}, \mathbb{B}_{ij})$ , is a completion tree  $\mathcal{T}' = \langle x'_0, N', E', \mathcal{L} \rangle$  generated in the following way:

<sup>3</sup> Since subsumption relation  $C \sqsubseteq D$  in  $AO_i$  *w.r.t.*  $\mathbb{A}\mathbb{O}_I$  iff  $C \sqcap \neg D$  is unsatisfiable in  $AO_i$  *w.r.t.*  $\mathbb{A}\mathbb{O}_I$ .

<sup>4</sup> Here we assume  $\lambda$  is in negation normal form; *i.e.*, negation is only applied to class names.

1.  $N' = \{x' \mid \text{if } x \in N\}$ ,
2.  $E' = \{\langle x', y' \rangle \mid \text{if } \langle x, y \rangle \in E\}$ ,
3.  $\mathcal{L}(x') = \{(j:j:C) \mid \text{if } (i:j:C) \in \mathcal{L}(x) \text{ and } (i:j:C) \in \mathbb{B}_{ij}\}$ ,
4.  $\mathcal{L}(\langle x', y' \rangle) = \{(j:j:R) \mid \text{if } (i:j:R) \in \mathcal{L}(\langle x, y \rangle) \text{ and } (i:j:R) \in \mathbb{B}_{ij}\}$ ,

## 4.2 Cautious Reasoning

*Cautious reasoning* relies on the knowledge of an ontology as well as the knowledge of its binding neighbour ontologies.

Given an autonomous ontology space  $\mathbb{A}\mathbb{O}_I = \{AO_i \mid i \in I\}$ , the procedure  $\mathcal{C}\text{-Tab}(\mathbb{A}\mathbb{O}_I, k, \lambda)$  verifies the satisfiability of an  $\mathcal{ALC}$  class description  $\lambda$  in ontology  $AO_k$  under the cautious semantics *w.r.t.*  $\mathbb{A}\mathbb{O}_I$ .

**Algorithm A-1.**  $\mathcal{C}\text{-Tab}(\mathbb{A}\mathbb{O}_I, k, \lambda)$

- 1: Let  $\mathcal{T} := \text{Tab}(AO_k, \langle x_0, \{x_0\}, \emptyset, \{\mathcal{L}(x_0) = \{\lambda\}\})$  // local expansion *w.r.t.*  $AO_k$
- 2: **repeat**
- 3:   **if**  $\mathcal{T}$  has a clash **then**
- 4:     **return** *unsatisfiable*
- 5:   **end if**
- 6:   **for** every binding neighbour autonomous ontology  $AO_i$  ( $i \in I$ ) of  $AO_k$  **do**
- 7:     **if** there exist  $S(i)$ -bound maximal sub-trees  $\mathcal{T}_1, \dots, \mathcal{T}_n$  of  $\mathcal{T}$  with roots  $x_1, \dots, x_n$ , respectively **then**
- 8:        $\mathcal{T}'_1 := P(\mathcal{T}_1, \mathbb{B}_{ki}), \dots, \mathcal{T}'_n := P(\mathcal{T}_n, \mathbb{B}_{ki})$  // sub-trees projection to  $AO_i$
- 9:        $\mathcal{T}'_1 := \text{Tab}(AO_i, \mathcal{T}'_1), \dots, \mathcal{T}'_n := \text{Tab}(AO_i, \mathcal{T}'_n)$  // local expansion of  $\mathcal{T}'_1, \dots, \mathcal{T}'_n$  *w.r.t.*  $AO_i$
- 10:       **if** any of  $\mathcal{T}'_1, \dots, \mathcal{T}'_n$  has a clash **then**
- 11:         **if**  $\mathcal{T}$  is backtrackable **then**
- 12:          $\mathcal{T} := \text{backtrack}(\text{Tab}(AO_i, \mathcal{T}))$  // backtrack and expand
- 13:         **else**
- 14:         **return** *unsatisfiable*
- 15:         **end if**
- 16:       **end if**
- 17:     **end if**
- 18:   **end for**
- 19:   **return** *satisfiable*
- 20: **until** false

In this algorithm,  $\mathcal{T}$  is initialized with a root  $x_0$  with  $\mathcal{L}(x_0) = \{\lambda\}$ , and is expanded by local completion rules *w.r.t.*  $AO_k$  (line 1 of **A-1**). As  $\mathcal{T}$  can have multiple binding neighbour ontologies, each of them should be taken care (line 6 of **A-1**). Note that  $\mathcal{T}$  might not be  $k$ -bound, the algorithm just project the maximal  $k$ -bound sub-trees, and then expand them by local completion rules *w.r.t.*  $AO_i$ , and expanded *w.r.t.*  $\mathbb{A}\mathbb{O}_I$  (lines 7-9 of **A-1**). If any of the projected sub-tree has a clash,  $\mathcal{T}$  needs to be backtracked (line 12 of **A-1**), expanded and start the checking all over again.

**Theorem 3.**  $\mathcal{C}\text{-Tab}(\mathbb{A}\mathbb{O}_I, k, \lambda)$  is a decision procedure to verify the cautious semantics satisfiability of an  $\mathcal{ALCN}$ -class description  $\lambda$  in ontology  $AO_k$  *w.r.t.*  $\mathbb{A}\mathbb{O}_I$ .

<sup>5</sup> This Algorithm is originated from [12], in which only positive concepts/roles can be projected to the original ontology, but the approach in this paper does not have this restriction.

To prove the theorem, we need to show that: (1) The algorithm always terminates. (2) The algorithm returns unsatisfiable if  $\mathbb{A}\mathbb{O}_I \not\models_C \lambda$ . (3) The algorithm returns satisfiable if  $\mathbb{A}\mathbb{O}_I \models_C \lambda$ . Due to limited space, we skip the detail.

### 4.3 Brave Reasoning

*Brave reasoning* not only relies on its binding neighbors, but also trusts the neighbors of its binding neighbors.

Given an autonomous ontology space  $\mathbb{A}\mathbb{O}_I = \{AO_i\}_{i \in I}$ , the procedure  $\mathcal{B}\text{-Tab}(\mathbb{A}\mathbb{O}_I, k, \lambda)$  verifies the satisfiability of an  $\mathcal{ALC}$  class description  $\lambda$  in ontology  $AO_k$  w.r.t.  $\mathbb{A}\mathbb{O}_I$  under brave semantics. It calls a recursive procedure  $\mathcal{DB}\text{-Tab}(\mathbb{A}\mathbb{O}_I, k, \mathcal{T})$  to expand a completion tree  $\mathcal{T}$  of  $AO_k$  w.r.t.  $\mathbb{A}\mathbb{O}_I$  under brave semantics.

**Algorithm A-2.**  $\mathcal{B}\text{-Tab}(\mathbb{A}\mathbb{O}_I, k, \lambda)$

- 1: Let  $\mathcal{T} := \langle x_0, \{x_0\}, \emptyset, \{\mathcal{L}(x_0) = \{\lambda\}\} \rangle$
- 2:  $\mathcal{T} := \mathcal{DB}\text{-Tab}(k, \mathcal{T})$
- 3: **if**  $\mathcal{T}$  has a clash **then**
- 4:     **return** *unsatisfiable*
- 5: **else**
- 6:     **return** *satisfiable*
- 7: **end if**

**Algorithm A-3:**  $\mathcal{DB}\text{-Tab}(\mathbb{A}\mathbb{O}_I, k, \mathcal{T})$

- 1: let  $\mathcal{T} := \text{Tab}(AO_k, \mathcal{T})$  // local expansion w.r.t.  $AO_k$
- 2: **repeat**
- 3:     **if**  $\mathcal{T}$  has a clash **then**
- 4:         **return**  $\mathcal{T}$  // *unsatisfiable*
- 5:     **end if**
- 6:     **for** every binding neighbour autonomous ontology  $AO_i$  of  $AO_k$  **do**
- 7:         **if** there exist  $S(i)$ -bound maximal sub-trees  $\mathcal{T}_1, \dots, \mathcal{T}_n$  of  $\mathcal{T}$  with roots  $x_1, \dots, x_n$ , respectively **then**
- 8:              $\mathcal{T}'_1 := P(\mathcal{T}_1, \mathbb{B}_{ki}), \dots, \mathcal{T}'_n := P(\mathcal{T}_n, \mathbb{B}_{ki})$  // sub-trees projection from  $AO_k$  to  $AO_i$
- 9:              $\mathcal{T}'_1 := \mathcal{DB}\text{-Tab}(\mathbb{A}\mathbb{O}_I, i, \mathcal{T}'_1), \dots, \mathcal{T}'_n := \mathcal{DB}\text{-Tab}(\mathbb{A}\mathbb{O}_I, i, \mathcal{T}'_n)$  // recursive calling  $\mathcal{DB}\text{-Tab}$  for brave reasoning on  $\mathcal{T}'_1, \dots, \mathcal{T}'_n$  w.r.t.  $AO_i$
- 10:             **if** any of  $\mathcal{T}'_1, \dots, \mathcal{T}'_n$  has a clash **then**
- 11:                 **if**  $\mathcal{T}$  is backtrackable **then**
- 12:                      $\mathcal{T} := \text{backtrack}(\text{Tab}(\mathcal{O}_i, \mathcal{T}))$  // backtrack and expand
- 13:                 **else**
- 14:                     **return**  $\mathcal{T}$  // *unsatisfiable*
- 15:                 **end if**
- 16:             **else**
- 17:                  $\mathcal{T}_1 := \beta(\mathbb{B}_{ki}, \mathcal{T}_1, \mathcal{T}'_1), \dots, \mathcal{T}_n := \beta(\mathbb{B}_{ki}, \mathcal{T}_n, \mathcal{T}'_n)$  //back-project
- 18:             **end if**
- 19:             **end if**
- 20:     **end for**
- 21:     **if**  $\mathcal{T}$  is not changed, **then**
- 22:         **return**  $\mathcal{T}$  // *satisfiable*
- 23:     **end if**
- 24: **until** false

In this algorithm, initially  $\mathcal{T}$  has a root  $x_0$  with  $\mathcal{L}(x_0) = \{\lambda\}$  (line 1 of **A-2**), and then it is expanded by local completion rules *w.r.t.*  $AO_k$  (line 1 of **A-3**). As  $\mathcal{T}$  can have multiple binding neighbour ontologies, each of them should be taken care (line 6 of **A-3**). Note that  $\mathcal{T}$  might not be  $S(i)$ -bound, maximal  $S(i)$ -bound sub-trees then should be projected, and expanded by local completion rules *w.r.t.*  $AO_i$ , if possible it also need to project to the binding neighbour of  $AO_i$ . (lines 7-9 of **A-3**). If any of the projected sub-tree has a clash,  $\mathcal{T}$  needs to be backtracked (line 12 of **A-3**), expanded and start the checking all over again; otherwise, we need to back-project the new  $S(i)$ -bound labels back to  $\mathcal{T}$  (line 17 of **A-3**). The algorithm **A-3** would not stop until  $\mathcal{T}$  is not changed.

**Theorem 4.**  $\mathcal{B}\text{-Tab}(\mathbb{A}\mathbb{O}_I, k, \lambda)$  is a decision procedure to verify the brave semantics satisfiability of an *ALCN*-class description  $\lambda$  in ontology  $AO_k$  *w.r.t.*  $\mathbb{A}\mathbb{O}_I$ .

To prove the theorems, we need to show that: (1) The algorithms always terminates. (2) The algorithm returns unsatisfiable if  $\mathbb{A}\mathbb{O}_I \not\models_{\mathcal{B}} \lambda$ . (3) The algorithm returns satisfiable if  $\mathbb{A}\mathbb{O}_I \models_{\mathcal{B}} \lambda$ . Due to limited space, we skip the proof.

## 5 Conclusions

In general understanding, ontologies are used for describing the structure of domain knowledge. Techniques for (partial) ontology reuse are important for ontology building, ontology discovery, and practical application of ontologies.

How to realise semantic cooperation among multiple ontologies is an important problem in the field of (partial) ontology reuse. In this paper we have proposed and analyzed a new framework for managing multiple ontologies that both preserves the autonomy of individual ontologies and also enables the semantic cooperation of different ontologies. We have also proposed two different reasoning mechanisms, called cautious reasoning and brave reasoning, for this framework and studied their properties. As we discussed in Section-1 this work is related to DDL [3], C-OWL [4],  $\mathcal{E}$ -Connection based approach [10], P-DL [2], Semantic Importing [12], and conservative extension [9].

Ontology modularization [15,16,8,13,11] is another interesting problem in this field. It attempts to partition one ontology and isolate functional modules. How to present formal functional modules in a single ontology by semantic binding (as proposed in this paper) could be an interesting extension of this approach.

Another interesting task is to explore whether our approach to ontology spaces and semantic binding can be applied to problems of forgetting, importing and extending in ontology spaces.

A natural task for the immediate future is to implement our two reasoning mechanisms and empirically evaluate their performance.

**Acknowledgment.** Thanks to Luciano Serafini at ITC-IRST in Trento Italy for discussions on primary ideas of this paper. This work was partially supported by the Australia Research Council (ARC) Discovery Project 0666107.

## References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.: *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, Cambridge (2003)
2. Bao, J., Caragea, D., Honavar, V.: Towards collaborative environments for ontology construction and sharing. In: *Proc. of the International Symposium on Collaborative Technologies and Systems (CTS 2006)*, pp. 90–108. IEEE Computer Society Press, Los Alamitos (2006)
3. Borgida, A., Serafini, L.: Distributed description logics: Assimilating information from peer sources. *Journal of Data Semantic* 1, 153–184 (2003)
4. Bouquet, P., Giunchiglia, F., van Harmelen, F., Serafini, L., Stuckenschmidt, H.: C-OWL: Contextualizing ontologies. In: *Proc. of the 2nd International Semantic Web Conference (ISWC 2003)*. USA, pp. 164–179 (2003)
5. Ghidini, C., Giunchiglia, F.: Local models semantics, or contextual reasoning = locality + compatibility. *Artificial Intelligence* 127(2), 221–259 (2001)
6. Giunchiglia, F.: Contextual reasoning. *Epistemologia*, special issue on *I Linguaggi e le Macchine XVI*, 345–364 (1993)
7. Giunchiglia, F., Serafini, L.: Multilanguage hierarchical logics, or: how we can do without modal logics. *Artificial Intelligence* 65(1), 29–70 (1994)
8. Grau, B.C., Parsia, B., Sirin, E., Kalyanpur, A.: Modularizing OWL Ontologies. In: *Proc. of the KCAP 2005 Workshop on Ontology Management* (2005)
9. Grau, B.C., Kutz, O.: Modular ontology languages revisited. In: *Proc. of the IJCAI 2007 Workshop on Semantic Web for Collaborative Knowledge Acquisition* (2007)
10. Grau, B.C., Parsia, B., Sirin, E.: Working with multiple ontologies on the semantic web. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) *ISWC 2004*. LNCS, vol. 3298, Springer, Heidelberg (2004)
11. Grau, B.C., Parsia, B., Sirin, E., Kalyanpur, A.: Modularity and web ontologies. In: *Proc. of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006)*. UK (June 2006)
12. Pan, J.Z., Serafini, L., Zhao, Y.: Semantic import: An approach for partial ontology reuse. In: *Proc. of the ISWC2006 Workshop on Modular Ontologies (WoMO)* (2006)
13. Paslaru, E.B., Mochol, M., Tolksdorf, R.: Case Studies on Ontology Reuse. In: *I-Know 2005*. *Proc. of the 5th International conference on Knowledge Management*, Austria (2005)
14. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL web ontology language semantics and abstract syntax (February 10, 2004)
15. Rector, A.L.: Modularisation of domain ontologies implemented in description logics and related formalisms including owls. In: *Proc. of the 2nd International Conference on Knowledge Capture (K-CAP)*. USA (2003)
16. Stuckenschmidt, H., Klein, M.: Structure-based partitioning of large concept hierarchies. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) *ISWC 2004*. LNCS, vol. 3298, Springer, Heidelberg (2004)
17. Zhao, Y., Serafini, L., Giunchiglia, F.: Autonomous ontology: Operations and semantics **or** local semantics with semantic binding on foreign entity. In: Mizoguchi, R., Shi, Z., Giunchiglia, F. (eds.) *ASWC 2006*. LNCS, vol. 4185, Springer, Heidelberg (2006)

# An Unsupervised Model for Exploring Hierarchical Semantics from Social Annotations

Mianwei Zhou, Shenghua Bao, Xian Wu\*\*, and Yong Yu

APEX Data and Knowledge Management Lab  
Department of Computer Science and Engineering  
Shanghai Jiao Tong University, 200240, Shanghai, P.R. China  
{kopopt,shhbao,yyu}@apex.sjtu.edu.cn wuxian@cn.ibm.com

**Abstract.** This paper deals with the problem of exploring hierarchical semantics from social annotations. Recently, social annotation services have become more and more popular in Semantic Web. It allows users to arbitrarily annotate web resources, thus, largely lowers the barrier to cooperation. Furthermore, through providing abundant meta-data resources, social annotation might become a key to the development of Semantic Web. However, on the other hand, social annotation has its own apparent limitations, for instance, 1) ambiguity and synonym phenomena and 2) lack of hierarchical information. In this paper, we propose an unsupervised model to automatically derive hierarchical semantics from social annotations. Using a social bookmark service Del.icio.us as example, we demonstrate that the derived hierarchical semantics has the ability to compensate those shortcomings. We further apply our model on another data set from Flickr to testify our model's applicability on different environments. The experimental results demonstrate our model's efficiency.

## 1 Introduction

Social annotation services have recently attracted considerable users and interest. Prominent web sites like Flickr<sup>1</sup>, Del.icio.us<sup>2</sup> are widely used and achieve significant success. These services not only provide user-friendly interfaces for people to annotate and categorize web resources, but also enable them to share the annotations and categories on the web, encouraging them to collaboratively enrich meta-data resources. In 2004, Thomas Vander Wal named these services “Folksonomy”, which came from the terms “folk” and “taxonomy” [1].

Compared with the traditional meta-data organization, folksonomy represents high improvement in lowering barriers to cooperation. Traditional taxonomy, which is predefined only by small groups of experts, is limited and might easily become outdated. Social annotation just solves these problems by transferring the burden from several individuals to all web users. Users could arbitrarily

---

\*\* Xian Wu is now working in IBM China Research Lab.

<sup>1</sup> <http://www.flickr.com>

<sup>2</sup> <http://del.icio.us>



annotate web resources according to their own vocabularies, and largely enrich the meta-data resources for Semantic Web.

However, although social annotation services have large potential to boom the Semantic Web, development of these services are impeded by their own shortcomings. Such shortcomings are mainly due to two features of folksonomy:

- Uncontrolled vocabulary. Breaking away from the authoritatively determined vocabulary, folksonomy suffers several limitations. One is ambiguity. People might use the same word to express different meanings. Another phenomenon is synonym. Different tags might denote the same meaning. With ambiguity and synonym, users might easily miss valuable information while gain some redundant information.
- Non-hierarchical structure. Folksonomy represents a flat but not hierarchical annotation space. This property brings difficulties in browsing those systems, moreover, makes it hard to bridge folksonomy and traditional hierarchical ontologies.

Aimed at overcoming those shortcomings, many researches have been conducted, for instance [2][3][4]. [2] introduced the concept of “navigation map” which described the relationship between data elements. The author showed how to gain semantic related images when users made queries. [3] gave a probabilistic method to allocate tags into a set of parallel clusters, and applied these clusters to search and discover the Del.icio.us bookmarks. Both of [2] and [3] focused on exploring relations between tags in the uncontrolled vocabulary, but still did not solve the non-hierarchy problem. In [4], the author proposed an algorithm to derive synonymic and hierarchical relations between tags, and demonstrated promising results. But the model is supervised, thus could not be effectively extended to other contexts, and also, lacks a sound theoretical foundation.

In our paper, we propose an unsupervised model, which could automatically derive hierarchical semantics from the flat tag space. Although search engines which aim to derive hierarchy out of search results have already existed (e.g. Vivísimo<sup>3</sup>), to the best of our knowledge, no work has been done before on exploring hierarchical semantics from tags. We demonstrate that the derived hierarchical semantics well compensates folksonomy’s shortcomings.

In order to derive the hierarchical semantics, our model proceeds in a top-down way. Beginning with the root node containing all annotations, we apply the splitting process to gain a series of clusters, each of which represents a specific topic. Further apply the splitting process on each cluster, smaller clusters with narrower semantics are gained. It’s easy to observe, this recursive process helps us obtain a hierarchal structure. A probabilistic unsupervised method named Deterministic Annealing(DA) algorithm is utilized in each splitting process. Unlike other clustering algorithm, DA algorithm could well control the cluster number and each cluster’s size with the help of a parameter  $T$ . We make use of this feature to ensure that each node’s semantics could be identified by a few tags.

Different from previous work, our model has several important features:

---

<sup>3</sup> <http://vivisimo.com/>

- Unsupervised model. Without any need of training data, it could be easily extended to other social annotation services.
- Hierarchical model. In the derived structure, each node represents an emergent concept and each edge denotes the hierarchical relationship between concepts.
- Self-controlled model. In our model, the number and the size of clusters are automatically determined during the annealing process.

The hierarchical semantics derived from our model has a large number of applications. Take two for example: 1) Semantic Web. The derived hierarchical semantics well serves as a bridge between the traditional strict ontology and the distributed social annotations. It would make ontology more sensitive to users' interests and demands, and reflect the current trends in the Internet; 2) Resource Browsing & Organization. The derived hierarchical semantics could also be utilized as effective tools for resources browsing and organization. Users could easily trace the path from the root to the node which contains information they want.

The rest of the paper is organized as follows. Section 2 briefly reviews the previous study of social annotation and DA algorithm. Section 3 gives a detailed description of our algorithm. Section 4 gives the experimental results and related evaluations. Finally we make a conclusion in Section 5.

## 2 Related Work

### 2.1 Related Work on Social Annotation

In these years, social annotation becomes a hot topic, on which many researches have been conducted. Part of these researches focused on discussing features of social annotations. [5,6] pointed out the advantages and limitations of social annotation, and described the contribution it would make to World Wide Web. [7] gave a brief review of those social annotation services available on network. In [8], the author discovered statistical regularities behind those collaborative tagging systems, and predicted the stable patterns through a dynamic model. [9] improved [8]'s work. The author showed the regularity behind those services could be described by a power law distribution. Furthermore, it showed that co-occurrence networks could be utilized to explore tags' semantic meaning.

For Semantic Web, the metadata resources usually exist as a form of pre-defined ontology. As social annotation services popularize, researchers aim to derive emergent semantics [10] from those systems, and utilize the derived structure to enrich the Semantic Web (e.g. [11,3,12]). [11] proposed an approach to extend the traditional bipartite model of ontologies with the social annotations. [3,12] are similar with our work. They respectively proposed model to derive emergent semantics from social annotations. However, in [3], the derived structure was still flat but not hierarchical. In [12], although the author constructed a topical hierarchy among tags, the derived structure was a simple binary tree, which might not be applicable for some complex social annotation environments.

Different from their work, we propose a novel model to derive hierarchical semantics which could effectively reflect the semantic concepts and hierarchical relationship from social annotations.

In addition to Semantic Web, some researches aimed at facilitating the social annotation application itself. In [2], the author proposed a similarity search model that allowed users to get concept-related data elements. [4] further explored the hierarchical relation between tags. [13] changed the perspective. The author presented a model to visualize the evolution of tags on the Flickr, thus users could gain the hottest images in any time interval. In [14], the author presented a model named FolkRank to exploit the structure of the folksonomy. [15] proposed two algorithms to incorporate the information derived from social annotations into page ranking.

## 2.2 Deterministic Annealing Background

The key algorithm in our model is named Deterministic Annealing (DA). It is an algorithm motivated by physical chemistry and mainly based on the information theory. In computer science, DA was widely utilized in the area of computer linguistics, computer vision and machine learning (e.g. [16,17,18,19]).

## 3 The Proposed Method

In this section, we give a detailed description of our model. The social annotations we use as our data set come from a popular bookmark service called Del.icio.us. It is very easy to extend our model to other common social annotation services such as Flickr, Technorati and so on.

### 3.1 Data Analysis

Del.icio.us is a social bookmark web service for sharing web bookmarks. Users could not only store and manage their own bookmarks, but also access others' bookmark storage at any time [20]. It is a flexible and useful tool for users with similar interests to share topics.

The data in Del.icio.us could be described as a set of quadruples:

$$(user, tag, website, time)$$

which means that the *website* is annotated by the *user* with the *tag* at the specific *time*. In our model, we focus on the tag and web elements. Let us denote the set

$$S_{tag} = \{t_1, t_2, \dots, t_N\}, S_{website} = \{w_1, w_2, \dots, w_M\}$$

$$S_{pair} = \{\langle t(i), w(i) \rangle | i \in [1, L]\}$$

where  $N$ ,  $M$ ,  $L$  respectively represent the number of tags, websites and pairs, and  $\langle t(i), w(i) \rangle$  represent that the  $i$ th pair includes the  $t(i)$ th tag and the  $w(i)$ th website.

### 3.2 Algorithm Overview

Our model builds the hierarchical structure in a top-down way. Beginning with the root node, the model recursively applies splitting process to each node until termination conditions are satisfied. In each splitting process, Deterministic Annealing(DA) algorithm is utilized. Figure 1 gives an intuitive description of this splitting process.

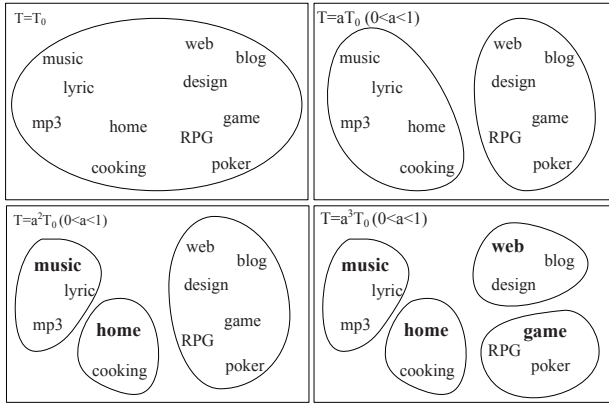


Fig. 1. The Emergent Semantics during the Annealing Process

In Figure 1, we observe that controlled by a parameter  $T$ , DA algorithm splits the node in a gradual way. As  $T$  is lowered from the first to the fourth subgraph, the cluster number increases from one to four finally. This process terminates when all clusters become “Effective Cluster”, or the number of “Effective Cluster” reaches a upper bound. The term “Effective Cluster” refers to those clusters whose semantics could be generalized by some specific tags. We name those tags “Leading Tag” for this cluster. It should be noted that the effective clusters do not emerge immediately. In the second sub-graph, neither of the clusters are effective clusters, because their semantics are too wide to be generalized by any tag. In the fourth sub-graph, all clusters are effective clusters, leading tags for which are “music”, “home”, “web” and “game” respectively. In our model, we design a criterion, which is given in section 3.4, to identify an effective cluster.

An overview of our model is given in Algorithm 1. In Algorithm 1, we maintain a queue  $Q$  to store the information of nodes which are waiting for splitting. Vector  $\mathbf{P}$  in the queue indicates the probability that each tag emerges in this node. At line 1, elements of  $\mathbf{P}_0$  are all initialized with 1, because all tags are contained in the root node. From line 2 to 10, the algorithm recursively splits each node until the termination condition is satisfied. We finally gain a hierarchical structure and each node’s semantics is identified by its corresponding leading tags.

Line 4 is a key part of our model. The function  $f_D$  serves as a clustering machine. Input the node’s information, and  $f_D$  outputs a series of effective clusters

**Algorithm 1.** Deriving Hierarchical Semantics

---

```

1: Initialize  $Q$ .  $Q$  is a queue containing one  $N$  dimensions vector  $\mathbf{P}_0 = (1, 1, \dots, 1)$ 
2: while  $Q$  is not empty do
3:   Pop  $\mathbf{P}$  from  $Q$ . Let  $\mathbf{P} = (p_0, p_1, \dots, p_N)$ .
4:    $\{p(c_i|t_j)|i \in [1, C], j \in [1, N]\} \leftarrow f_D(\mathbf{P})$ 
5:   for each cluster  $c_i, i = 1, 2, \dots, C$  do
6:     Extract leading tags  $t_{c_i}$  to stand for the semantics of cluster  $c_i$ 
7:     if  $c_i$  could be further split then
8:       Let  $\mathbf{P}' = (p'_0, p'_1, \dots, p'_N)$ 

$$p'_j = \begin{cases} p_j * p(c_i|t_j) & t_j \neq t_{c_i} \\ 0 & t_j = t_{c_i} \end{cases}$$

       Push  $\mathbf{P}'$  into  $Q$ .
9:     else
10:      The remaining tags except leading tags  $t_{c_i}$  form leaves for the current node.
11:    end if
12:  end for
13: end while

```

---

derived from this node. Each cluster is described by the value  $p(c_i|t_j)$  represents the relativity between the  $j$ th tag and the  $i$ th cluster. As discussed before, DA algorithm is utilized in  $f_D$ . Detailed implementation of this algorithm is given in the following section. The termination condition for DA algorithm is given in section 3.4.

### 3.3 Apply Deterministic Annealing for Clustering

In this section, we introduce how to apply Deterministic Annealing(DA) algorithm to split a tag set on a node into several effective clusters. In mathematics, DA and other similar optimizing algorithms could all be stated as a process to minimize a predefined criterion. In our model, such criterion is given below:

$$D = \sum_{i=1}^N \sum_{j=1}^C p(c_j|t_i) * d(t_i, c_j) \quad (1)$$

where  $d(t_i, c_j)$  measures the relativity between tag  $t_i$  and the cluster  $c_j$ . We used KL-divergence to describe this distance.

$$d(t_i, c_j) = \sum_{k=1}^M p(w_k|t_i) * \log\left(\frac{p(w_k|t_i)}{p(w_k|c_j)}\right) \quad (2)$$

where  $p(w|t_i)$  and  $p(w|c_j)$  respectively measure tag  $t_i$ 's and cluster  $c_j$ 's distributions on all websites. Through measuring KL-divergence between these two distributions, we gain the semantic distance between tag  $t_i$  and the concept that cluster  $c_j$  represents. With closer semantic relation between them,  $d(t_i, c_j)$

becomes smaller. It is easy to observe, as  $D$  is minimized, the value of  $p(c|t)$  indicates a clustering result.

In the minimizing process of  $D$  above, general clustering algorithm might easily suffer a poor local minimum. In order to overcome this problem, DA recasts the minimization problem by introducing an annealing process. The minimization of  $D$  is converted to the minimization of free energy  $F$  subject to a specified level of randomness.

$$F = D - TH \tag{3}$$

$H$  is a measure of level of randomness, given below

$$H = - \sum_{i=1}^N \sum_{j=1}^C p(c_j|t_i) * \log[p(c_j|t_i)] \tag{4}$$

Free energy  $F$  and entropy  $H$  are two terms in the physical annealing theory. Temperature  $T$  could control entropy  $H$  in different scales during the minimization of  $F$ . As  $T$  is lowered,  $H$  also decreases. As illustrated in Figure 1, with low entropy  $H$ , every tag is more definitely linked to clusters, resulting in the increment of cluster number.

---

**Algorithm 2.** Apply DA Algorithm for Clustering

---

```

1: Input:  $P$ 
2:  $C \leftarrow 2, T \leftarrow T_0.$ 
3: Set  $p(c_i|t_j)$  with random values between 0 and 1, satisfying  $\sum_{i=1}^C p(c_i|t_j) = 1$ , for all  $j = 1, 2, \dots, N.$ 
4: loop
5:    $p^{(0)}(c_i|t_j) \leftarrow p(c_i|t_j), k \leftarrow 0,$  calculate  $F^{(0)}.$ 
6:   repeat
7:      $k \leftarrow k + 1$ 
8:     Calculate  $p^{(k)}(c_i|t_j)$  with  $p^{(k-1)}(c_i|t_j)$  according to Equation (6)
9:     Calculate  $F^{(k)}$  according to Equation (3)
10:  until  $|F^{(k)} - F^{(k-1)}| < \epsilon$ 
11:  Let  $p^{(K)}(c|t)$  be the final iteration result.
12:  if all clusters are effective clusters then
13:    return  $p^{(K)}(c|t)$ 
14:  end if
15:  if Critical Temperature for cluster  $c_i$  is reached then
16:     $p(cc_{+1}|t_j) \leftarrow p(c_i|t_j)/2 + \delta, p(c_i|t_j) \leftarrow p(c_i|t_j)/2 - \delta,$  where  $\delta$  indicates a random perturbation.
17:     $C \leftarrow C + 1$ 
18:  else
19:     $p(c_i|t_j) \leftarrow p^{(K)}(c_i|t_j)$ 
20:  end if
21:   $T \leftarrow \alpha T (0 < \alpha < 1)$ 
22: end loop

```

---

A detailed implementation of DA is given in Algorithm 2 as a supplement of line 4 in Algorithm 1. In Algorithm 2, line 1 is the input  $\mathbf{P}$  which contains the information of the node waiting for splitting. Line 2 to 3 are the initialization steps. Line 4 to 21 represent the annealing process of the algorithm. Among them, Expectation-Maximum(EM) algorithm is utilized to minimize the free energy  $F$  in line 5 to 11. The termination condition for this algorithm is given in line 12 to 14. From line 15 to 20, we determine when the cluster number should be increased. In line 19, temperature  $T$  is lowered preparing for next annealing process. In the following section, we would further discuss the detail about minimizing  $F$  and determining the increment of the cluster number.

**EM Algorithm for Minimizing F.** We utilize EM algorithm to iteratively minimize  $F$ . Firstly, the equation (3) is recast as

$$F = \sum_{i=1}^N \sum_{j=1}^C p(c_j|t_i) * (\sum_{l=1}^M p(w_l|t_i) * \log(\frac{p(w_l|t_i)}{p(w_l|c_j)}) + T * \log(p(c_j|t_i))) \quad (5)$$

Through EM algorithm,  $p(c|t)$  could be estimated by iteratively minimizing the free energy  $F$ . Beginning with the initial value for  $p^{(0)}(c_i|t_j)$ , we give the  $p^{(k)}(c_i|t_j)$  in the  $k$ th iteration.

$$p^{(k)}(c_i|t_j) = \frac{\exp(-\frac{d^{(k)}(t_j, c_i)}{T}) * p^{(k)}(c_i)}{\sum_{l=1}^C \exp(-\frac{d^{(k)}(t_j, c_l)}{T}) * p^{(k)}(c_l)} \quad (6)$$

where

$$p^{(k)}(c_i) = \sum_{j=1}^N p^{(k-1)}(c_i|t_j) * p(t_j) * p_j \quad (7)$$

$$p^{(k)}(w_l|c_i) = \frac{\sum_{j=1}^N p^{(k-1)}(c_i|t_j) * p(t_j) * p_j * p(w_l|t_j)}{p^{(k)}(c_i)} \quad (8)$$

$$d^{(k)}(t_j, c_i) = \sum_{l=1}^M p(w_l|t_j) * \log(\frac{p(w_l|t_j)}{p^{(k)}(w_l|c_i)}) \quad (9)$$

where,  $p(c)$  denotes the probability that the cluster is assigned.  $p(t)$  denotes the probability that the tag occurs in the data set.  $p_i$  denotes the probability that the  $i$ th tag occurs in the current sub-node.  $p(w|t)$  denotes the relativity between the website and the tag. Among them,  $p(w|t)$  and  $p(t)$  are invariants which could be computed directly from the data set, while  $p(c)$ ,  $p(w|c)$ , and  $d(t, c)$  are variant, which are converging during the whole iteration process. Given  $P$  and  $T$ ,  $F$  finally converges to a minimum after a series of iterations. For further details about the derivation of the formulas, refer to [19].

**Critical Temperature Determination.** From line 15 to 20 in Algorithm 2, we introduce a new concept “Critical Temperature”. In the DA algorithm theory, once the temperature reaches certain clusters’ critical temperature, those clusters should be split, so that the Free Energy could be further minimized. This process is named “Phase Transition”. The increment of cluster number in DA algorithm is achieved by a series of phase transitions. It has been theoretically proved that this critical temperature could be calculated, but the computation is too complex. [19] introduced a simple alternative to estimate critical temperature. In this method, an extra copy is kept for each cluster. Only when the critical temperature is reached for a cluster, its copy would split away, otherwise, the copy would merge again after the iteration. We utilize this method in our model. Once phase transition for certain clusters is detected, we add a new cluster in line 16.

### 3.4 Effective Cluster Identification

As discussed in the previous section, DA algorithm in our model terminates only when all clusters become effective clusters, or the number of effective clusters reaches an upper bound. In this section, we give a criterion to identify whether a cluster is effective.

The main difference of the effective cluster from other common ones is that, as an effective cluster, its semantics could be generalized by some specific tags, which we name “Leading Tag”. To measure a tag’s capability to summarize the whole cluster’s semantics, we define  $Cov(t_i, c_j)$  to measure a tag’s coverage as below.

$$Cov(t_i, c_j) = \sum_{k=1}^N p(t_k|c_j)b_{i,k} \quad (10)$$

where,  $b_{i,k} \in \{0, 1\}$  indicates whether there exists a website annotated by both tags  $t_i$  and  $t_k$ .  $p(t|c)$  could be easily gained by applying Bayesian Theorem on  $p(c|t)$ . The high value of  $Cov(t_i, c_j)$  indicates that tag  $t_i$  has covered lots of other tags in cluster  $c_j$ , so  $t_i$  is more capable to summarize cluster  $c_j$ ’s semantics. Using  $Cov(t_i, c_j)$ ,  $E(c_j)$  measuring whether  $c_j$  is an effective cluster is defined.

$$E(c_j) = \max_{i \in [1, N]} Cov(t_i, c_j) \quad (11)$$

The qualification for a cluster to be an effective one is measured by the leading tag with highest  $Cov(t_i, c_j)$  in it. If multiple leading tags are allowed,  $E(c_j)$  could also be measured by several largest ones. During the annealing process,  $E(c_j)$  increases as the size of clusters is reduced. Once  $E(c_j)$  reaches a high value, it indicates that the leading tag  $t_{c_j}$  has emerged, so we accept this cluster as an effective cluster.

## 4 Experiment

### 4.1 Experiment Setup

Our experiment is mainly conducted on two samples of Data: Del.icio.us and Flickr. We filter those tags and urls which emerge less than 20 times in the data set. The statistics for both of the raw and the filtered data is present in Table 1.

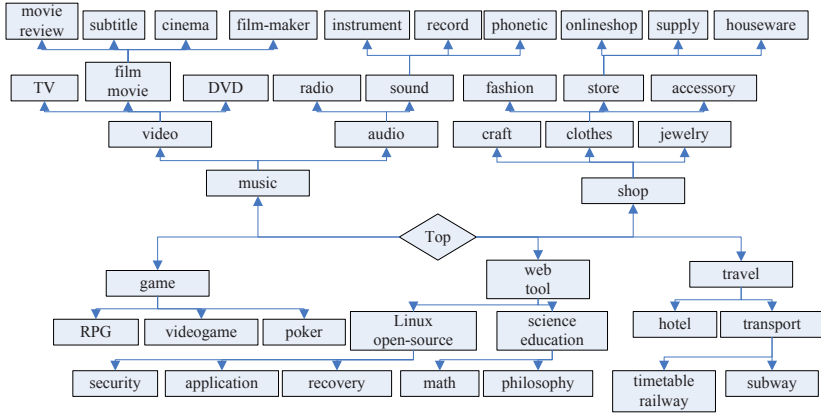


**Table 1.** Statistics of Data Sets

Source	Raw Data			Filtered Data			Crawled Time
	tag	url	pair	tag	url	pair	
Del.icio.us	192143	784617	3357809	8445	16963	479035	April 2006
Flickr	32465	23713	204717	3927	6127	70761	April 2007

## 4.2 Experiment on Del.icio.us

**Derived Hierarchical Structure.** We apply our model on the Del.icio.us data set described above. Figure 2 shows part of the derived hierarchical result.

**Fig. 2.** Hierarchical Semantics Derived from Del.icio.us

In Table 2, we randomly choose some nodes from each hierarchy, and display their locations and child-clusters. Each node “(tag1, tag2,...)” in Table 2 denotes a cluster with several leading tags. In Figure 2 and Table 2, we observe that the derived hierarchical semantics is well matched with people’s common knowledge.

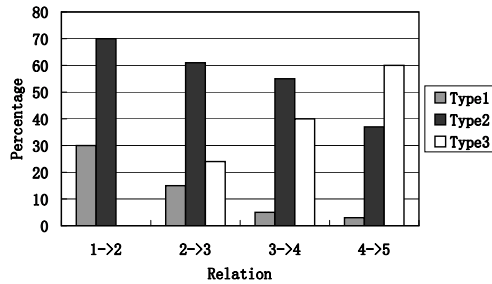
Because our model is based on statistics about human behaviors, it is hard to restrict the derived relationship to a specific type. In further experiment, we discover that the hierarchical relationship mainly includes three types. Suppose B is the child node of A

1. B is the sub-type of A (e.g. “RPG” and “videogame” are both “game”).
2. B is the related aspect of A (e.g. “hotel” and “transportation” to “travel”).
3. B is parallel to A (e.g. the sub-node of “DVD” is “WMA”, “DV”).

In Figure 3, we present a statistics of each type’s portion between different hierarchies. It’s observed that type 1 and 2 mainly exist in the higher level of the tree, and type 3 exists in the lower level. Although type 3 deviates our original purpose, we should not expect our model to derive a precise ontology like Wordnet containing only type 1 and 2. When the semantics of a node becomes

**Table 2.** Clusters in Different Hierarchies

Leading Tag	Ancestor Node	Child Node
food, health	<Top>	(fit), (sport), (eat, bread, coffee), (cook, recipe), (beer)
politics	<Top>	(government), (law, right), (active), (censorship), (conspiracy, 911), (Israel, Iran, Syria), (military, war), (Africa), (habitat, human)
language	<Top>→ (web, tool)	(write), (English, linguist, word), (translate), (encyclopedia), (Chinese, Mandarin)
jewelery	<Top>→ (shop)	(Chicago, glass), (ear, bracelet, bridal, necklace, ring), (handmade), (unusual), (stainless, diamond)
webdesign, webdevise	<Top>→ (web, tool) → (program, develop)	(html, xhtml, standard), (ajax, xml), (tutorial, code, opensource), (sql, mysql), (framework, python), (menu, navigate), (color, palette), (encode, unicode, UTF8)
DVD	<Top>→ (music) → (video)	(WMA, MP4, quicktime), (DV, camcorder, miniDV), (codec, Divx, mpeg, avi)
cryptography, encrypt	<Top>→ (web, tool) → (Linux, opensource) → (security)	(PKI), (computers and internet), (GPG, GNUPG), (MD5), (OpenSSL)



**Fig. 3.** Statistics for Each Type of Relation between Different Hierarchy Levels

narrower in lower level, it is a hard task to select leading tags to summarize the semantics of the node by human, let alone by computer.

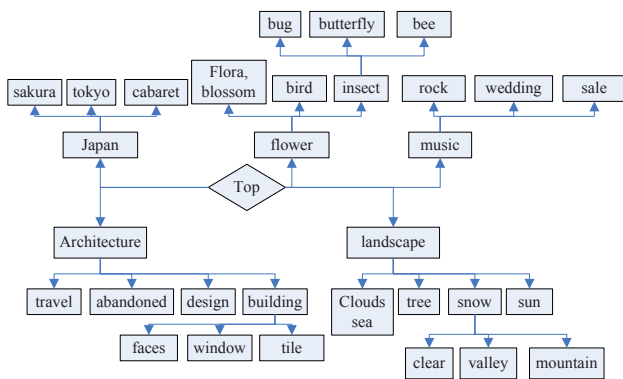
**Distribution of Tags on Different Nodes.** The distribution of tags on different nodes is also studied. In Table 3, we randomly select some tags and give their linked clusters with largest probabilities. For those well-known polysemantic words (e.g. “wine”, “apple”), their diverse meanings could be observed through different paths. For other common words, different nodes could represent their distinct related aspects. For instance, the word “honeymoon” is related not only to “travel” and “holiday”, but also to “gift”. This feature of our model well solves the ambiguity problem. In the derived hierarchical structure, a lot of tags has more than one related node, but at most five. It is because when temperature is lowered in the iterative steps, tags would easily converge to one or two clusters, but not scatter equally on several ones.

**Table 3.** Distribution of Tags on Different Nodes

Tags	Distribution on Different Nodes
agriculture	1. (environment) → (sustain, green) → (agriculture) 2. (food) → (garden) → (agriculture, farm)
wine	1. (web, tool) → (Linux, opensource) → (freeware) → (Wine) 2. (food) → (coffee, eat, tea) → (wine)
price	1. (money, finance) → (bill) → (price) 2. (shop) → (deal, buy) → (price)
gasoline	1. (shop) → (deal, buy) → (gasoline) 2. (travel) → (transport) → (automobile) → (gasoline)
honeymoon	1. (travel) → (hotel) → (holiday) → (honeymoon) 2. (gift) → (jewelry) → (bridal, wed) → (honeymoon)
apple	1. (web, tool) → (Linux, open-source) → (Apple, Mac) 2. (food) → (coffee, eat, tea) → (apple)

### 4.3 Experiment on Flickr

We also apply our model on a sample of Flickr data set to demonstrate our model’s wide applicability. With effective self-controlled capability, our model well captures different features of social annotation environment in Flickr. Figure 4 gives part of the result.



**Fig. 4.** Hierarchical Semantics Derived from Flickr

Form Figure 4, we discover that the derived relation is reasonable according to people’s knowledge. Compared with the structure derived from Del.icio.us, the number of derived hierarchical relations is much less. Most of the nodes concentrate on the first and second hierarchies with parallel relations. It is mainly because Flickr is a “Narrow Folksonomy” [21] compared with the “Broad Folksonomy” Del.icio.us. In the Narrow Folksonomy, most of the tags are singular

and directly linked to the object. This property largely limits the hidden semantics in social annotations. However, our model still captures the hidden topics behind Flickr and presents a satisfying hierarchical result.

## 5 Conclusion and Future Work

Social annotation has become more and more popular because of its strengths. But at the same time, it also has its own shortcomings, for instance, 1) ambiguity and synonymous phenomena 2) non-hierarchical structure. In order to overcome these shortcomings, we build an unsupervised model to derive hierarchical semantics from social annotations. The main contributions can be concluded as follows:

1. The proposal to study the problem of deriving hierarchical semantics from social annotations.
2. The proposal of an unsupervised model for automatic semantic clustering, and hierarchical relationship identification.
3. The evaluation of the proposed model on both Del.icio.us and Flickr. The preliminary experimental result demonstrates the model's effectiveness.

In our current work, the evaluation of our model is mainly based on people's intuition and common sense. We would do more detailed evaluation by comparing this hierarchical semantics with other web taxonomy, like ODP. Moreover, we would emphasize on applying our results in real applications to measure our model's efficiency.

## Acknowledgement

The authors would like to thank Xiao Ling, Xiaojun Zhang, Rui Li, Bai Xiao and Hao Zheng for their valuable suggestions. The authors also appreciate the four anonymous reviewers for their elaborate and helpful comments.

## References

1. Smith, G.: Folksonomy: social classification. *Atomiq/Information Architecture [blog]* (2004), [http://atomiq.org/archives/2004/08/folksonomy\\_social\\_classification.html](http://atomiq.org/archives/2004/08/folksonomy_social_classification.html)
2. Aurnhammer, M., Hanappe, P., Steels, L.: Augmenting navigation for collaborative tagging with emergent semantics. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006*. LNCS, vol. 4273, Springer, Heidelberg (2006)
3. Wu, X., Zhang, L., Yu, Y.: Exploring social annotations for the semantic web. In: *Proceedings of the WWW 2006*, pp. 417–426 (2006)
4. Li, R., Bao, S., Fei, B., Su, Z., Yu, Y.: Towards effective browsing of large scale social annotations. In: *Proceedings of the WWW 2007*, pp. 943–952 (2007)

5. Mathes, A.: Folksonomies-cooperative classification and communication through shared metadata. *Computer Mediated Communication, LIS590CMC (Doctoral Seminar)*, Graduate School of Library and Information Science, University of Illinois Urbana-Champaign (December 2004)
6. Quintarelli, E.: Folksonomies: power to the people. ISKO Italy-UniMIB meeting (June 2005), available at <http://www.iskoi.org/doc/folksonomies.htm>
7. Hammond, T., Hannay, T., Lund, B., Scott, J.: Social bookmarking tools (i). *D-Lib Magazine* 11(4), 1082–9873 (2005)
8. Golder, S., Huberman, B.: Usage patterns of collaborative tagging systems. *Journal of Information Science* 32(2), 198 (2006)
9. Halpin, H., Robu, V., Shepherd, H.: The complex dynamics of collaborative tagging. In: *Proceedings of the WWW 2007*, pp. 211–220 (2007)
10. Aberer, K., Cudre-Mauroux, P., Ouksel, A., Catarci, T., Hacid, M., Illarramendi, A., Kashyap, V., Mecella, M., Mena, E., Neuhold, E., et al.: Emergent semantics principles and issues. In: Lee, Y., Li, J., Whang, K.-Y., Lee, D. (eds.) *DASFAA 2004. LNCS*, vol. 2973, Springer, Heidelberg (2004)
11. Mika, P.: Ontologies are us: a unified model of social networks and semantics. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005. LNCS*, vol. 3729, pp. 522–536. Springer, Heidelberg (2005)
12. Brooks, C., Montanez, N.: Improved annotation of the blogosphere via autotagging and hierarchical clustering. In: *Proceedings of the WWW 2006*, pp. 625–632 (2006)
13. Dubinko, M., Kumar, R., Magnani, J., Novak, J., Raghavan, P., Tomkins, A.: Visualizing tags over time. In: *Proceedings of the WWW 2006*, pp. 193–202 (2006)
14. Hotho, A., Jaschke, R., Schmitz, C., Stumme, G.: Information retrieval in folksonomies: Search and ranking. In: Sure, Y., Domingue, J. (eds.) *ESWC 2006. LNCS*, vol. 4011, Springer, Heidelberg (2006)
15. Bao, S., Wu, X., Fei, B., Xue, G., Su, Z., Yu, Y.: Optimizing web search using social annotations. In: *Proceedings of WWW 2007*, pp. 501–510 (2007)
16. Pereira, F., Tishby, N., Lee, L.: Distributional clustering of english words. In: *Proceedings of the 31st conference on Association for Computational Linguistics*, pp. 183–190 (1993)
17. Yang, X.L., Song, Q., Zhang, W.: Kernel-based deterministic annealing algorithm for data clustering. *IEEE Proceedings-Vision, Image, and Signal Processing* 153, 557 (2006)
18. Wanhyun, C., Park, J., Lee, M., Park, S.: Unsupervised color image segmentation using mean shift and deterministic annealing em. In: *ICCSA 2004. Internat. Conf. on Computational Science and Its Applications*, vol. 3, pp. 867–876 (2004)
19. Rose, K.: Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of the IEEE* 86(11), 2210–2239 (1998)
20. Schachter, J.: Del.icio.us about page (2004), <http://del.icio.us/about/>
21. Vander Wal, T.: Explaining and showing broad and narrow folksonomies (2005), [http://www.personalinfocloud.com/2005/02/explaining\\_and\\_.html](http://www.personalinfocloud.com/2005/02/explaining_and_.html)

# SPARK: Adapting Keyword Query to Semantic Search

Qi Zhou, Chong Wang, Miao Xiong, Haofen Wang, and Yong Yu

Apex Data and Knowledge Management Lab\*  
Department of Computer Science and Engineering,  
Shanghai JiaoTong University, 200240, Shanghai, P.R. China  
{jackson,wangchong,xiongmiao,whfcarter,yyu}@apex.sjtu.edu.cn

**Abstract.** Semantic search promises to provide more accurate result than present-day keyword search. However, progress with semantic search has been delayed due to the complexity of its query languages. In this paper, we explore a novel approach of adapting keywords to querying the semantic web: the approach automatically translates keyword queries into formal logic queries so that end users can use familiar keywords to perform semantic search. A prototype system named ‘SPARK’ has been implemented in light of this approach. Given a keyword query, SPARK outputs a ranked list of SPARQL queries as the translation result. The translation in SPARK consists of three major steps: term mapping, query graph construction and query ranking. Specifically, a probabilistic query ranking model is proposed to select the most likely SPARQL query. In the experiment, SPARK achieved an encouraging translation result.

## 1 Introduction

In the next stage of web revolution, termed the semantic web, web resources will be made available with various kinds of metadata described in ontologies<sup>1</sup>. Correspondingly, many semantic query languages (e.g. RQL<sup>2</sup>, RDQL<sup>3</sup>, SquishQL<sup>2</sup> and SPARQL<sup>3</sup>) have been proposed for querying these ontologies. However, in order to use these semantic query languages, end users have to master complex formal logic representations and be familiar with the underlying ontologies. This has become a critical gap between semantic search and end users<sup>3</sup><sup>4</sup>. Meanwhile, most users have been accustomed to the traditional keyword search for years. Therefore, it is valuable to enable the users to carry out semantic search by inputting keyword query. However, keyword query is very different from semantic search. To adapt keyword query to semantic search, we have to overcome the following obstacles: 1) *Vocabulary Gap*: Casual web users usually

---

\* This work is funded by IBM China Research Lab.

<sup>1</sup> In this paper, ontology refers to a knowledge base (KB) that includes concepts, relations, instances and instance relations that together model a domain.

<sup>2</sup> <http://www.w3.org/Submission/RDQL/>

<sup>3</sup> <http://www.w3.org/TR/rdf-sparql-query/>

have no knowledge of the underlying ontology, so the words in their queries may be quite different from those in the ontology. 2) *Lack of Relation*: Relations between concepts/instances are required to be explicitly stated in formal logic queries, which are often missing in keyword queries [5]. How to automatically discover these missing relations becomes a big problem. 3) *Query Ranking*: Due to the ambiguity of keyword query, there may be multiple formal queries produced from one keyword query. How to rank these queries is a big challenge [6].

Faced with these difficulties, we present our novel approach in SPARK system. SPARK can automatically translate keyword queries into corresponding SPARQL queries under the domain ontology, with the aim of adapting keyword query to semantic search. The main translation steps are: *term mapping*, *query graph construction* and *query ranking*. *Term mapping* maps the terms of a keyword query to the resources of the knowledge base to narrow the vocabulary gap. After that, *query graph construction* links the mapped resources so that the missing relations and concepts can be obtained and a complete query graph can be constructed. Finally, the probabilistic *query ranking* model estimates the most likely SPARQL query among all the candidate queries. In this way, the end users can keep the habit of typing keywords and querying the semantic web data transparently, which increases the social utilization of semantic search. From the evaluation of 750 various keyword queries over three ontologies, SPARK achieved an encouraging MRR [4] score of 0.677.

The rest of this paper is organized as follows: Section 2 defines the problem of formal query construction. Section 3 details the main steps of formal query construction in SPARK. The implementation and experimental result are presented in Section 4. Section 5 outlines the related work. We give the conclusion and future work in the last section.

## 2 From Keywords to Formal Query

Answering many kinds of semantic query can be formulated as a problem of finding a group of objects which are connected by certain relationships and restrictions. In ontology, a semantic query is equal to a query graph with constrained object nodes and property arcs. An example query is “*Find all the states that the mississippi river runs through and border texas*” querying the geography ontology (Fig. 1). It can be rewritten as an equivalent conjunctive formal logic expression:

$$?x \leftarrow (?x, \text{is}, \text{State}) \cap (\text{Mississippi River}, \text{runThrough}, ?x) \cap (?x, \text{border}, \text{Texas})$$

where *class* ‘State’, *instance* ‘Mississippi River’ and ‘Texas’ are ontological restrictions on nodes while ‘border’ and ‘runThrough’ are the required connecting arcs in the pattern. ‘?x’ is the variable passing these restrictions on the query. So we can reduce the problem of translating keyword queries into formal queries to the problem of constructing equivalent query graphs from keywords. To clarify the problem, we give the formal definitions as follows:

<sup>4</sup> MRR: Mean Reciprocal Rank [16].

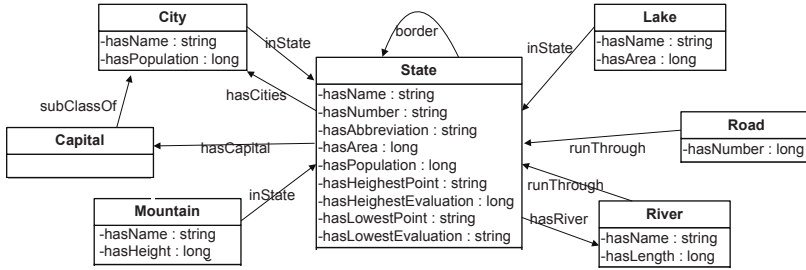


Fig. 1. Geography Ontology (Schema)

Knowledge base  $D : \langle C, I, L, R, \tau \rangle$  is a directed graph  $G_D$  where:  $C$  and  $R$  define the sets of class and relation.  $I$  and  $L$  are the sets of instance and literal [8]. Function  $\tau : (C \cup I) \times (C \cup I \cup L) \rightarrow R$  defines all the triples in  $D$ . Additionally, we use symbol resource  $\{e\} : \{C \cup R \cup I \cup L\}$  to represent all classes, relations, instances and literals.

Keyword query  $K$  is a bag of terms  $\{t\}$ . In our assumption, the end users can have no knowledge of the underlying ontology and any arbitrary keyword queries can be issued by the users.

Formal query  $F : \langle C', R', I', L', V, \tau' \rangle$  over  $D$  is a graph  $G_F$  subsumed by  $G_D$ .  $V$  is the set of variable nodes which conjunct the relations and nodes.  $\tau' : (I' \cup C' \cup V) \times (I' \cup C' \cup V \cup L') \rightarrow R$  defines all the triples in  $F$ .

From the definitions above, the formal query construction problem can be modeled as:

*Under knowledge base  $D$ , given a keyword query  $K$ , constructing and ranking candidate formal queries  $\{F\}_{\succeq}$  such that the most likely query is among the highest ranked ones.*

For example, assume that a web user intends to express the information need “Show me the states which the mississippi river runs through?”. He may type “mississippi river state” as his keyword query. In SPARK, the formal query construction process will map these keywords into knowledge base and complete the candidate formal query graphs under  $D$ . In resolving the ambiguity (term ‘mississippi’ may refer to ‘mississippi state’ or ‘mississippi river’) and information loss (lack of relations among terms), there are many candidate formal queries with different senses as the construction results. The query ranking process then estimates a confidence score for each candidate formal query so that the more likely formal queries are ranked higher. Take the formal queries in Table 1 as an example, each one is assigned a ranking score (Column ‘Total’). In the following section, we will illustrate the detailed approach in SPARK framework for formal query construction.

<sup>5</sup> To simplify the problem, we will treat all literal as instance in later sections.



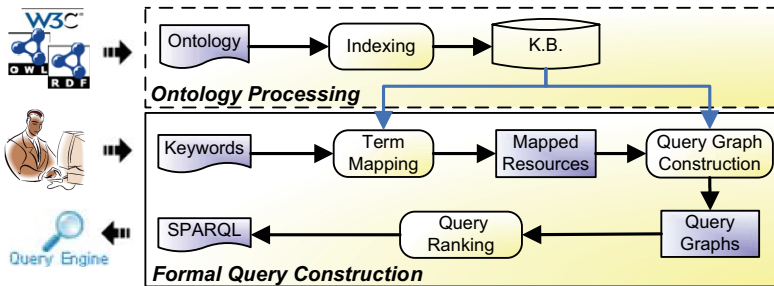
**Table 1.** Some formal queries translated from keyword query ‘mississippi river state’

Formal Logic Representation	KQM	KBM	Total	Rank
$?x \leftarrow (?x, \text{is}, \text{State})$	0.160	0.495	0.0792	7
$?x \leftarrow (?x, \text{is}, \text{River})$	0.160	0.544	0.0870	6
$?x \leftarrow (?x, \text{is}, \text{State}) \cap (\text{Mississippi River}, \text{runThrough}, ?x)$	0.333	0.653	0.2898	1
$?x, \leftarrow (?x, \text{is}, \text{State}) \cap (?x, \text{border}, \text{Mississippi State})$	0.246	0.854	0.2100	3
$?x \leftarrow (?x, \text{is}, \text{River}) \cap (?x, \text{runThrough}, \text{Mississippi State})$	0.289	0.877	0.2534	2
$?x \leftarrow (?x, \text{is}, \text{State}) \cap (?x, \text{hasCities}, \text{Fall-River City})$	0.211	0.891	0.1880	4
$?x \leftarrow (?x, \text{is}, \text{State}) \cap (?x, \text{hasCities}, \text{Fall-River City}) \cap (\text{Mississippi River}, \text{runThrough}, ?x)$	0.129	0.979	0.1263	5

### 3 The SPARK Approach

The framework of SPARK consists of two modules (Fig. 2): *ontology processing* module and *formal query construction* module. When an ontology is selected as the underlying knowledge base, the *ontology processing* module automatically indexes its resources. The *formal query construction* module takes keywords as input, and returns a ranked list of SPARQL queries as output.

Once a user inputs a keyword query, the *term mapping* step uses a group of mapping methods to find the corresponding resources in the knowledge base according to user’s keywords. Then, the *query graph construction* step enumerates all possible query combinations and applies *Minimum Spanning Tree* algorithm [7] to construct complete query graphs with different senses from the mapped resources, during which some missing relations or concepts will be made up in the query. Finally, the *query ranking* step evaluates the constructed formal queries from two perspectives: the keyword query model (KQM) and the knowledge base model (KBM). A ranked list of SPARQL queries will be given back to the end user. In the next three sections, we will detail the *term mapping* and *query graph construction* steps, and model the *query ranking* problem.

**Fig. 2.** SPARK Framework

### 3.1 Term Mapping

The purpose of *term mapping* is to find corresponding ontology resources (i.e. *classes*, *instances*, *properties* and *literals*) for each term in the keyword query. The names and labels of ontology resources are used for mapping. In our implementation, two types of mapping methods are employed: 1) *morphological mapping* employs string comparison techniques such as stemming, Sub-String, Edit-Distance, and I-Sub [8] to find morphologically similar words; 2) *semantic mapping* mainly utilizes general dictionaries like WordNet [11] to find semantically relevant words (e.g. synonyms). During a term mapping process, each term in  $K$  is matched against the knowledge base with different mapping methods, e.g. term ‘river’ can be mapped to two ontology resources: *class* ‘River’ by direct mapping and *instance* ‘Fall-River City’ by Sub-String mapping. We assign a pre-defined confidence value  $P(e|t)(P(e|t) \in (0, 1])$  to each mapping method to determine the mapping quality. Generally, the confidence value for a direct matching is higher than that for a synonym-based matching. Term mapping associates each term in keyword query with senses under the knowledge base. Therefore, after the term mapping process, a term is no longer a lexical string but represents a list of resources indicating what kinds of elements the user wants. In the next section, *query graph construction* process will construct candidate formal query graphs with different query senses from these mapped ontology resources.

### 3.2 Query Graph Construction

The *query graph construction* process builds up candidate query graphs with the ontology resources mapped above. Firstly, the mapped resources are split into different query sets. Then, *Minimum Spanning Tree* algorithm is applied to construct possible query graphs for each query set. Finally, each query graph is interpreted into a SPARQL query by conversion rules.

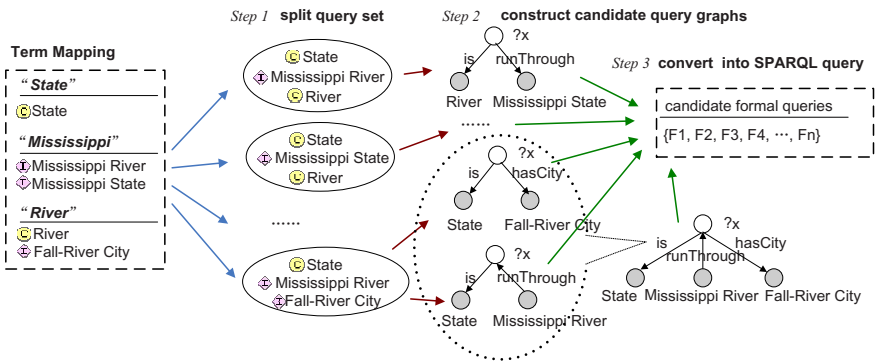


Fig. 3. Query Graph Construction

In the step of query set split: for each term  $t_i$  in  $K$ , if there is only one resource mapped from  $t_i$ , we directly add it into every  $F$  of query set  $\{F\}$ ; otherwise we duplicate the original query set  $\{F\}$  and add each mapped resource of  $t_i$  into every  $F$  in different sets. The purpose of query split is to assign a definite query sense to each formal query, which is a process of enumerating all possible combinations from different senses of each term.

After enumerating possible query set  $\{F\}$ , SPARK uses the *Kruskal's Minimum Spanning Tree* algorithm [7] to join these mapped resources of  $F$  into a complete query graph for each  $F$  in query set. Besides SPARK, the *Minimum Spanning Tree* has been introduced for inferring SQL query by DBExplorer [9]. In the *query graph construction*, SPARK makes up the missing relations and concepts for each query set by exploring its schema graph. Take the keyword query 'state mississippi river' for example: there is no explicit relation between 'state' and 'mississippi river' expressed. From the schema graph of Geography Ontology (Fig. 11), one relation 'runThrough' entails between instance 'Mississippi River' and class 'State'. So this edge is introduced to construct our final query graph. We use these discovered nodes as variable nodes pointing to edges of query graph and discovered edges are added into the query graph. If a connected query graph can be joined with all resources in query set  $F$ , we directly regard it as the candidate formal query; otherwise we take the largest component instead.

Finally, we generate the query graph according to the following rules: 1) Resource *class* mapped by terms or discovered by graph exploration are regarded as variable nodes. 2) Resource *instance* and *literal* are regarded as end nodes. 3) Resource *property* are regarded as the edges of query graph. Since SPARQL is a graph pattern based query language, it is straightforward to convert the query graph into corresponding SPARQL query string.

### 3.3 Query Ranking

After the *term mapping* step and *query graph construction* step, multiple candidate formal queries will be produced from the original keyword query. There comes the problem: how to pick up the most likely formal query for the end users? In this section, *query ranking* is used to solve the problem. We model *query ranking* as: "In knowledge base  $D$ , what is the probability of a constructed formal query  $F$  being a user issued query from the given keyword query  $K$ ?". That is, we determine  $P(F|D, K)$ : the probability of generating event  $F$  under event  $D$  and  $K$ , which is the core idea of treating the query ranking problem as conditional probability event. The challenge turns to how to estimate this probability. Instead of estimating this probability directly, we apply *Bayes' Theorem* and obtain

$$P(F|D, K) = \frac{P(D, K|F)P(F)}{P(D, K)} \quad (1)$$

where  $P(F)$  is the priori probability of formal query  $F$ .  $P(D, K|F)$  is the probability of generating knowledge base and keyword from the constructed formal

query. We assume that the keyword query  $K$  and knowledge base  $D$  are independent events. Accordingly,  $P(D, K|F)$  can be divided into two parts, and  $P(F|D, K)$  can be written as:

$$P(F|D, K) = \frac{P(D|F)P(K|F)P(F)}{P(D)P(K)} \quad (2)$$

Formula 2 is not intuitive, so we apply *Bayes' Theorem* again into  $P(D|F)$  and  $P(K|F)$  in Formula 2 and obtain a more intuitive formula:

$$P(F|D, K) = \frac{P(F|D)P(F|K)}{P(F)} \propto P(F|D)P(F|K) \quad (3)$$

$P(F|D)$  is the probability of generating  $F$  from  $D$ .  $P(F|K)$  is the probability of generating  $F$  from  $K$ .  $P(F)$  is the priori probability of  $F$ . In this paper, we assume all the formal queries are in uniform distribution and  $P(F)$  is equal among all candidate formal queries. Thus, the query ranking model is proportional to two sub probabilistic models: the keyword query model  $P(F|K)$  and the knowledge base model  $P(F|D)$ . The next two sub-sections will illustrate the estimation of the two models in SPARK.

**Keyword Query Model (KQM).** The keyword query model reflects the probability of generating a formal query from a keyword query. Generally, there are two intuitive properties.

- *Mapping Proximity*: The ranking function should take keyword mapping proximity into account. In the keyword mapping, for every term  $t_i$  in  $K$ , the mapping probability is introduced to indicate the similarity between terms of  $K$  and resources of  $D$ . A formal query with resources of higher mapping scores should be given a higher priority than those with lower mapping score.
- *Relevance to Keyword Query*: The ranking function should rank the formal queries higher which are more relevant to user's keyword. The concepts mentioned in formal query with more keywords from user should be given a higher score.

Take the constructed queries in Table 1 for example, the third and the fourth queries are more reasonable from the perspective of the keyword query while the rest queries either have too few of user's expressed resources or add too many resources from the knowledge base. We define the captured information need of  $K$  by the distance measurement between  $K$  and  $F$ . To calculate the distance, we resolve the distance into two features: keyword mapping proximity  $proximity(F, K)$  and query relevance  $relevance(F, K)$ :

Keyword mapping proximity score  $proximity(F, K)$  bases on the average *term mapping* proximity  $P(e|t)$  score. A higher  $proximity(F, K)$  indicates that  $F$  is more relevant to user's terms.

$$proximity(F, K) = \frac{\sum_{e_i \in F} P(e_i|t)}{|t \in K|} \quad (4)$$

Query relevance score  $relevance(K, F)$  is determined by the proportion of common resources in  $F$  and  $K$ . The proportion of common resources for  $K$  reflects how many resources in  $F$  are directly mapped from  $K$ . And the proportion of common resources for  $F$  reflects how many resources mapped from  $K$  contained in  $F$ .

$$relevance(F, K) = \frac{|(e \in F) \cap (P(e|t) > 0)|}{|t \in K|} \cdot \frac{|(e \in F) \cap (P(e|t) > 0)|}{|e \in F|} \tag{5}$$

Thus, the overall keyword query model is interpreted as:

$$P(F|K) \propto proximity(F, K) \cdot relevance(F, K) \tag{6}$$

**Knowledge Base Model (KBM).** We measure  $P(F|D)$  for each formal query with its information content. Information content has been used by [10, 12] for ranking the semantics of query. In information theory, information contained in an event is measured by the negative logarithm of the occurrence probability of the event. For example, given that  $\{x_i\}$  is a discrete random event set with probabilities  $(p_1, \dots, p_n$  and  $\sum p_i=1$ ) of their occurrence, the information content of event  $x_i$  is given by  $information(X = x_i) = -\ln p_i$ .

In query graph, we measure the information content by its query graph patterns. Take the query graphs in Fig. 4 for example, the right one employs a more complex structure than that of the left one. Its question node  $?x$  is not only constrained by the instance ‘Fall-River City’ but also the instance ‘Mississippi River’. Obviously, its query result contains more information than those of the left query graph. In the query graph  $F_G$ , question node ( $?x$ ) is modeled as the overall event

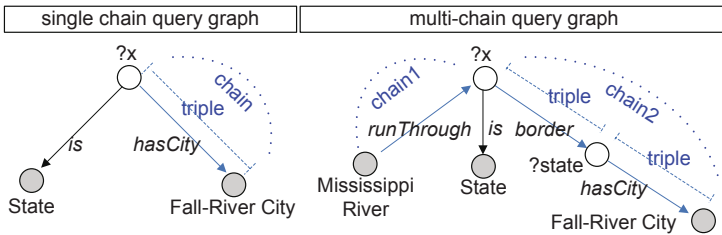


Fig. 4. Triple chain in query graph

and the path from it to end instance is viewed as a chain of events. Each event is a statement triple  $\tau'$  like  $(?state, hasCity, Fall-River City)$ . The right query graph in Fig. 4 consists of two chains. *chain1*:  $(?x, hasCity, Fall-River City)$  and *chain2*:  $(?x, border, ?state) (?state, hasCity, Fall-River City)$ . Each  $\tau'$  states an isolated event: given a resource (*Fall-River City*) or a variable node (*?state*), the probability of choosing the relation stated in  $\tau'$  among all other relations is modeled as an event. The lower probability of choosing certain relation, the more information contained in  $\tau'$ . Assuming a resource has a set of relations  $\{R'\}$ , the probability

of choosing relation  $r' \in \{R'\}$  to construct  $\tau'$  is modeled as the probability of the event  $\tau'$ . To determine the weight of relation, we assign the probability of choosing  $r'$  as its frequency in  $D$ . Frequency reflects the importance of the relation in knowledge base: the more triples share the relation, the more common the relation is. Therefore, the probability of event  $\tau'$  is:

$$P(\tau') = \frac{\text{freq}(r')}{\sum_{r_i \in \{R'\}} \text{freq}(r_i)} \quad (7)$$

Given a chain with triples  $(\tau' \in \text{chain})$  from the question node to the end node. We assign the probability of the chain as its triple with lowest probability  $P(\text{chain}) = \min\{P(\tau')\}_{\tau' \in \text{chain}}$ . If multi-chain exists in  $F$  (Fig. 4), the event for the join variable node is determined by the overall probability of these separated event chain:  $P(\text{Event}_F) = P(\text{chain1}) \cdot P(\text{chain2})$ . Thus, the information content can be computed by:

$$I(F) = -\ln(P(\text{Event}_F)) = -\ln \prod_{\text{chain}_i \in F} P(\text{chain}_i) \quad (8)$$

After computing the information content score of the query, we use *sigmoid function* to adjust the information content score and estimate its probability  $P(F|D)$  by.

$$P(F|D) \propto \left| \alpha - \frac{1}{1 + e^{-I(F)}} \right| \quad (9)$$

Using information content biases the formal queries with lower-probability. To balance the information content, we use a parameter  $\alpha (\alpha \in (0, 1))$  for adjusting the ranking threshold. If the user wants the frequently-asked query, he can adjust  $\alpha$  with corresponding slider in SPARK's web page. In the end, the *query ranking* process sorts these candidate formal queries by its overall probability score  $P(F|K, D)$  from two sub-models.

## 4 Implementation and Experiment

SPARK is implemented in Java and Jena<sup>6</sup> API. It has a web interface<sup>7</sup> for online users. A user can choose the domain ontology, type his keywords and get a ranked list of SPARQL queries. These translated SPARQL queries can also be directly sent to ARQ<sup>8</sup> search engine to find related resources. Additionally, each SPARQL query has an automatically translated natural language query to clarify their information need.

In the rest of this section, we describe the experiments to validate the performance of SPARK and exploit its usability in querying various ontologies. Our goal is to observe the performance of the query ranking model as well as the query construction capability under different ontologies with various keyword queries. Additionally, we will also discuss the usability of SPARK from the experiences of end users.

<sup>6</sup> <http://jena.sourceforge.net/>

<sup>7</sup> <http://spark.apexlab.org>

<sup>8</sup> <http://jena.sourceforge.net/ARQ/>

## 4.1 Experiment Setup

Our experiment was performed on a PC with 3.2GHz Pentium(D) CPU and 2GB Memory. As far as we know, there is no test data specially designed for translating keyword queries on ontology. Therefore, we manually constructed these test knowledge base and keyword queries from Mooney Natural Language Learning Data [14], which has been used by many database and ontology-based querying experiments [3, 13, 15]. Firstly, we converted the test dataset (*geography*, *job* and *restaurant*) into RDFS ontologies. Then, we manually translated its natural language query into keyword queries for two purposes: to set up a test data set for similar applications and to make the evaluation more realistic. In the translating process, each natural language query was re-written as multiple short keywords query according to the understanding on it. For example: the keyword query ‘*state ohio river flow*’ was translated from the natural language query ‘*show me all the states that river ohio runs through?*’. Then, these translated keyword queries were sent to SPARK and the constructed SPARQL queries were evaluated with the gold-standard SPARQL query created by its original natural language query. If the constructed SPARQL is not semantically equivalent to the gold-standard one, it is considered wrong.

We took two metrics in the evaluation: Recall and MRR. Recall indicates whether SPARK can construct at least one proper SPARQL query in its candidate queries while MRR focuses on the overall performance of SPARK. For each set of the constructed SPARQL queries by SPARK, we computed the performance by Reciprocal Rank (RR) of the first correct answer. If none of the SPARQL queries is correct, a score of 0 is given. Otherwise, the score is equal to the reciprocal of the rank of the first correct one. We calculated the mean reciprocal rank of all test keyword queries as MRR which is widely used in evaluating question-answering tasks [16].

## 4.2 Experiment Result

We used 250 keyword queries for each ontology. From the result in Table 2, the recall is 0.846, 0.824, 0.711 and the MRR is 0.755, 0.764, 0.513 for *geography*, *restaurant* and *job* ontology respectively, which is rather encouraging. The right constructed query is ranked as the first (MRR=1) or the second query (MRR=0.5) on average. From the evaluation result, SPARK can process most of the keyword queries entailing conjunctive relations such as ‘*city in virginia*’, ‘*capital of the states*, *border new mexico*’. However, due to the limitation of keyword query, some of the semantics in keyword query can not be handled. For example: given ‘*area*’ and ‘*population*’ in ontology, we can’t find a correct formal query for the keyword query ‘*state of smallest population density*’. Negation and superlative forms in keyword queries are inscrutable to be understood in current implementation, which is the main cause to loss in recall.

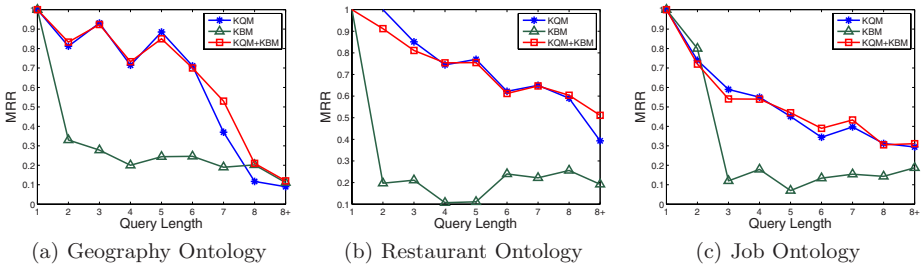
The ambiguity of keyword queries has been a big challenge since the invention of keyword search. To exploit the ability of processing ambiguous keyword query, we made an analysis of our test keywords and picked up those with ambiguous



**Table 2.** Evaluation Result over Geography, Restaurant, Job Ontology. ( Query Total: count of all keyword query; AMB: count of ambiguous keyword query; AVL: average keyword query length.)

Ontology	Knowledge Base				Query			Result		Time
	Concept	Relation	Instance	Triple	Total	AMB	AVL	Recall	MRR	Avg(sec)
<i>Geography</i>	7	16	1018	3748	250	60	3.204	0.846	0.755	0.191
<i>Restaurant</i>	7	10	4315	108817	250	75	6.79	0.824	0.764	0.235
<i>Job</i>	12	19	11018	56868	250	49	6.24	0.711	0.513	0.249
Average	-	-	-	-	-	-	-	0.793	0.677	0.225

terms. There are 60(24%), 75(30%), 79(31.6%) ambiguous keyword queries in the evaluation set respectively. From the result, there is no obvious differences between unambiguous and ambiguous keyword queries. This is due to the tight coupling between the resources in query graph: the triple relation fit in with the intuition that resources with strong semantic meanings have other resources to reinforce one another. For the keyword query: ‘mississippi city’, term ‘Mississippi’ may refer to the instance ‘Mississippi River’ or instance ‘Mississippi State’. In the knowledge base, ‘Mississippi State’ has a relation property ‘hasCity’ while ‘Mississippi River’ does not. Therefore, the final formal query with instance ‘Mississippi State’ is ranked with higher overall score.



**Fig. 5.** MRR score for keyword query model (KQM), knowledge base model (KBM) and integration of two models (KQM+KBM)

In Fig. 5 we present comparison of ranking performance of keyword query model (KQM) and knowledge base model (KBM). As we can see, based on the integration of the two models, SPARK outperforms the best. Compared to the knowledge base model, the keyword query model contributes to the ranking performance more. However, with the increase in query length, the performance of keyword query model decreases. In that case, the knowledge base query can smooth the performance of the overall ranking score. The overall performance on keyword query of different length indicates that our query ranking model works best on the medium and short sized keyword queries (2-6 terms), which are the most frequently-used in today’s web search engine. From the statistics



on time cost (Table 2), the time to process a keyword query takes 0.225 second on average, which is efficient for the implementation.

Besides the evaluation on test data set, we also made a study on the user experience of SPARK. From the feedback of 50 online users in two months, SPARK can construct correct SPARQL queries from their keywords most of the time. They thought that it is very easy for them to locate their desired information in ontologies quickly. The query ranking is effective: almost all of the right SPARQL queries can be found within the first three queries. Additionally, SPARK can handle abbreviated query. For example, query *'hill of nm'* issued by user can be easily translated into a formal query representing *'Show me all the mountains in the state of New Mexico'* in SPARK.

## 5 Related Work

In the last few years, there has been increasing interest in applying keyword query to structured data such as XML [17] and Relational Database [9] [18] [19]. The attractions are that users can keep the habit in traditional web search and do not need to know about the data schema. Banks [18] works by starting shortest-path search on each matching element in order to find out a join tree. DBExplorer [9] and Discover [19] take advantage of data schema to compute a set of possible join networks. In the context of semantic web, there has been little work on inferring candidate query graphes from keywords, which is very important to semantic search.

Faced with the gap between the formal logic based semantic query and the end users, some communities have proposed various solutions to narrow it: Bernstein et al. explore providing controlled language [15] and guided natural language interface [3]. From the perspective of query refinement [12] [20], the gap between users' information needs and its semantic querying is quantified by measuring several types of query ambiguities through incremental interaction. Graphical based search [21] also contributes a way, by building graph queries through browsing and selection on ontology.

Compared to these semantic search methodologies, using keywords lowers the formal scaffolding of semantic search. SemSearch [6] has a little-structured keyword query interface to hide the complexity of semantic search. Avatar Semantic Search [22] is a prototype search engine that exploits annotations in the context of classical keyword search. Another representative keyword based semantic search application is OntoLook [5]: a prototype relation-based search engine. OntoLook mentioned the weakness of keyword query in the context of semantic web and inferred possible relations among keywords to improve the precision of the search. Compared to these applications, SPARK not only covers the relation and concept missing problem but also gives solutions to the problems of query ranking and semantic matching proposed in SemSearch.

Many methodologies on deciding the best query have been proposed before SPARK: Ontologer [23] builds a query mechanism by recording user's behaviors in an ontology and recall it for ranking. Banks [18] incorporates different weight

on vectors into the relevance score. SemRank [10] uses information gain theory to rank the discovered path between two resources. To the best of our knowledge, most of the ranking approaches are feature based and there is no work today addressing the query ranking problem systematically. In this paper, we give definition of the problem and present an effective query ranking implementation.

## 6 Conclusion and Future Work

In this paper, we formalize the formal query construction problem and present an effective approach in SPARK to resolve it. SPARK aims at translating keyword queries into SPARQL queries to narrow the gap between formal logic based semantic search and end users. Additionally, the probabilistic ranking model implemented in SPARK can well explain what the desired properties are for a likely constructed formal query translated from keywords.

The main contributions of this paper are: 1) it puts forward the problem of translating keywords into formal logic based queries for semantic search; 2) it presents a novel solution to this problem which is implemented as SPARK that translates keywords into SPARQL queries; 3) it provides a probabilistic query ranking model for picking most likely translated formal queries from keywords. From the evaluation of 750 keyword queries over three ontologies, SPARK achieved an encouraging translation result.

In future work, we consider to enhance SPARK in two ways: 1) enlarge its query scope by introducing some structured operators (e.g. NOT, OR, etc) and improving human interaction support to translate more complicated information needs. 2) extend SPARK's approach to multiple ontologies for web-scaled usage.

## References

1. Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D., Scholl, M.: RQL: A Declarative Query Language for RDF. In: WWW 2002. Proceedings of the 11th international conference on World Wide Web, Honolulu, Hawaii, USA, pp. 592–603. ACM Press, New York (2002)
2. Miller, L., Seaborne, A., Reggiori, A.: Three Implementations of SquishQL, a Simple RDF Query Language. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, Springer, Heidelberg (2002)
3. Bernstein, A., Kaufmann, E.: GINO - A Guided Input Natural Language Ontology Editor. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
4. Chakrabarti, S.: Breaking Through the Syntax Barrier: Searching with Entities and Relations. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 9–16. Springer, Heidelberg (2004)
5. Li, Y., Wang, Y., Huang, X.: A Relation-Based Search Engine in Semantic Web. Proceedings of IEEE Transactions on Knowledge and Data Engineering 19(2), 273–282 (2007)
6. Lei, Y., Uren, V., Motta, E.: SemSearch: A Search Engine for the Semantic Web. In: Proceedings of EKAW 2006, pp. 238–245 (2006)

7. Kruskal, J.: On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. In: Amer. Math. Soc. (1956)
8. Stoilos, G., Stamou, G., Kollias, S.: A String Metric for Ontology Alignment. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 623–637. Springer, Heidelberg (2005)
9. Chaudhuri, S., Das, G., Narasayya, V.: DBExplorer: A System for Keyword Search over Relational Databases. In: Proceedings of Data Engineering 2002 (2002)
10. Anyanwu, K., Maduko, A., Sheth, A.: SemRank: ranking complex relationship search results on the semantic web. In: Proceedings of WWW 2005, Chiba, Japan, pp. 117–127. ACM Press, New York (2005)
11. Miller, G.A.: Wordnet: a lexical database for english. *Commun. ACM* 38(11), 39–41 (1995)
12. Stojanovic, N., Stojanovic, L.: A Logic-based Approach for Query Refinement in Ontology-based Information Retrieval Systems. In: Proceedings of the 16th IEEE Int. Conf. on Tools with Artificial Intelligence, Illinois USA, IEEE Computer Society Press, Los Alamitos (2004)
13. Popescu, A.M., Etzioni, O., Kautz, H.A.: Towards a theory of natural language interfaces to databases. In: IUI, pp. 149–157 (2003)
14. Tang, L.R., Mooney, R.J.: Using Multiple Clause Constructors in Inductive Logic Programming for Semantic Parsing. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, Springer, Heidelberg (2001)
15. Bernstein, A., Kaufmann, E., Gohring, A., Kiefer, C.: Querying ontologies: A controlled English interface for end-users. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
16. Voorhees, E.: Overview of the TREC 2001 Question Answering Track. In: Proceedings of TREC-X, Gaithersburg, Maryland, pp. 157–165 (2001)
17. Guo, L., Shao, F., Botev, C., Shanmugasundaram, J.: XRANK: Ranked Keyword Search over XML Documents. In: Proceedings of SIGMOD 2003, San Diego, California, pp. 16–27. ACM Press, New York (2003)
18. Bhalotia, G., Hulgeri, A., Nakhe, C., Chakrabarti, S., Sudarshan, S.: Keyword Searching and Browsing in Databases Using BANKS. In: Proceedings of ICDE 2002, Illinois USA, ACM Press, New York (2002)
19. Hristidis, V., Papakonstantinou, Y.: DISCOVER: Keyword Search in Relational Databases. In: Proceedings of VLDB 2002 (2002)
20. Carlos A. Hurtado, A.P., Wood, P.T.: A Relaxed Approach to RDF Querying. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
21. N Athanasis, V.C., Kotzinos, D.: Generating On the Fly Queries for the Semantic Web: The ICS-FORTH Graphical RQL Interface (GRQL). In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, Springer, Heidelberg (2004)
22. Kandogan, E., Krishnamurthy, R., Raghavan, S., Vaithyanathan, S., Zhu, H.: Avatar Semantic Search: A Database Approach to Information Retrieval. In: Proceedings of SIGMOD 2006, pp. 790–792. ACM Press, New York (2006)
23. Stojanovic, N., Gonzalez, J., Stojanovic, L.: Ontologer: A System for Usage-Driven Management of Ontology-Based Information Portals. In: Proceedings of L-CAP 2003 (2003)

# Unlocking the Potential of Public Sector Information with Semantic Web Technology

Harith Alani<sup>1</sup>, David Dupplaw<sup>1</sup>, John Sheridan<sup>2</sup>, Kieron O'Hara<sup>1</sup>, John Darlington<sup>1</sup>,  
Nigel Shadbolt<sup>1</sup>, and Carol Tullo<sup>2</sup>

<sup>1</sup> Intelligence, Agents, Multimedia  
School of Electronics and Computer Science  
University of Southampton, Southampton, UK  
{h.alani, dpd, jd, kmo, nrs}@ecs.soton.ac.uk

<sup>2</sup> Office Of Public Sector Information  
Admiralty Arch, The Mall  
London, UK  
{john.sheridan, carol.tullo}@opsi.x.gsi.gov.uk

**Abstract.** Governments often hold very rich data and whilst much of this information is published and available for re-use by others, it is often trapped by poor data structures, locked up in legacy data formats or in fragmented databases. One of the great benefits that Semantic Web (SW) technology offers is facilitating the large scale integration and sharing of distributed data sources. At the heart of information policy in the UK, the Office of Public Sector Information (OPSI) is the part of the UK government charged with enabling the greater re-use of public sector information. This paper describes the actions, findings, and lessons learnt from a pilot study, involving several parts of government and the public sector. The aim was to show to government how they can adopt SW technology for the dissemination, sharing and use of its data.

## 1 Introduction

Public Sector Information (PSI) can make an important contribution to bootstrapping the SW, which in turn will yield many gains. UK government tends to see the web primarily as a medium for the delivery of documents and the dissemination of content to the citizen. With the emergence of a re-use policy agenda for PSI, the UK government is beginning to develop a far richer and deeper understanding of the SW and the contribution it can make in terms of achieving greater efficiency through information sharing and integration to realise broader economic and social gains.

The Office of Public Sector Information (OPSI) is responsible for the management of all of the UK government's intellectual property, including setting standards, delivering access and encouraging the re-use of PSI. In the UK, any work produced by an employee of the government is deemed to be owned by the Crown and thus subject to Crown copyright. Under this constitutional position, Carol Tullo, the Director of OPSI and co-author to this paper, is granted authority by Her Majesty The Queen to manage

---

<sup>1</sup> Developed from Her Majesty's Stationery Office (HMSO).

all copyrights and databases owned by the Crown. OPSI also has an important role as a regulator of holders of public sector information (e.g. the Met Office, Ordnance Survey) for their information trading activities.

In the UK, large quantities of public sector information have been made available, ranging from geospatial, statistical, financial and legal information. However, making data available and making data reusable are two very different things. Most government data is published online in text formats with little structure, thus inhibiting its re-use. By using unstructured, non-semantic representations of the data, it becomes almost impossible for machines to find or understand and integrate this rich source of information. For these reasons, OPSI decided to initiate a research project, AKTivePSI. The aim of AKTivePSI is to show how the use of SW technology can facilitate the large scale integration and re-use of public sector information, ultimately to the benefit of government, business and citizen alike. AKTivePSI was about building prototypes and demonstrators to mainly win the hearts and minds of some government agencies and show them how, and what will it take, to become semantically enabled. Several of the organisations that actively participated are now investing in SW technologies, as will be highlighted in section 6.

In the following sections we will report on the decisions, actions, and results of AKTivePSI, which involved several government and information trading organisations that collect, store, and publish public sector information.

## 2 Related Work

The UK has developed a strong e-Government agenda over the last ten years, initially focussed on providing access to information and more latterly on delivering public services online. The publication of the government's IT strategy document, "Transformational Government - Enabled by Technology" [1] in 2005 marked an important shift in the UK government's thinking to a much broader technology agenda.

Crucially the government has identified overcoming problems with information sharing as being integral to transforming services and reducing administrative burdens on citizens and business. The UK government is committed to leveraging and producing open standards, and the GovTalk programme [2] has key documents that describe interoperability frameworks and metadata standards. With this in place, the scene is ideally set for SW technologies now to take centre stage. To use the new parlance, transformational government will require the use of transformational technology for information sharing.

The Access-eGov [3] project has been investigating how current governmental websites may be annotated using a shared reference ontology and intend to roll out methodologies on a test-bed of Eastern European governmental websites. They suggest that guided markup of current web-pages and content is perhaps the way to go. However, they correctly write that developers do not have the necessary domain knowledge to create the reference ontology, creating an extra layer of bureaucracy in the development of the system [9]. Similarly the Quebec government in Canada have embarked

---

<sup>2</sup> <http://www.govtalk.gov.uk/>

on producing a SW-service-based portal, also using a reference ontology to markup the government's web-pages.

The BRITE [18] project is building a SW infrastructure for specific areas of governmental record keeping, in this case European-wide business registrations. Vitvar and colleagues [19] explain how SW services can be used as part of the proposed Pan-European E-Government Services (PEGS) proposal, which will go some way to addressing the follow-on problem of how to integrate semantic data from different countries.

Information integration is of great importance in B2B scenarios. There are several advantages in using ontology-based architectures for information integration, such as ease of mapping, handling of different terminologies, explicit data models, etc. [3].

Using Semantic Web Services (SWS) for the integration and sharing of distributed data sources has also been suggested and demonstrated in B2B scenarios [13,4]. Existing B2B standards for data exchange usually require considerable effort from organisations to agree how exactly they are to be used and implement that [13]. SWS is offered as an alternative to describe and discover information. This approach could allow for dynamic integration of resources, assuming that they have been appropriately described in a SWS language (eg WSML, WSDL).

### 3 AKTivePSI

Information policy has developed quite quickly in the UK over the last five years, with Freedom of Information legislation as well as the EU Directive, but no large scale work had been done to research the potential for reuse using SW technologies and approaches. OPSI initiated AKTivePSI as an exemplar to show what could be achieved if public sector information was made available for reuse in an enabling way.

#### 3.1 Aims of AKTivePSI

Integrating and sharing information from distributed sources contains several obstacles and problems [5], such as scalability, different terminologies and formats, cost, etc. After meeting with the AKTivePSI government participants, we noticed that many of them shared the following misguided opinions or beliefs:

- Ontologies are very large, complex, and expensive data models
- Everyone has to agree and adopt the same terminology to enable data sharing
- To participate in the SW, their existing data infrastructures will need to be replaced with new technology
- Opening access to data only benefits the consumer, and not the provider

Our first task in this project was to correct the above misunderstandings to gain the support of the participants and encourage to provide data and some resources. The initial aims of the project were to draw together a sufficiently large set of heterogeneous information from a selection of public sector organisations in order to explore: (a) How SW technology can help turn government information into re-useable knowledge to fuel e-government, (b) Investigate the best practical approach to achieve this goal, in terms

of collecting data and constructing ontologies (c) Show how can data be integrated, and identify existing government taxonomies that are useful for this task, and (d) provide evidence that there is added value from undergoing this process.

Throughout the project, we had regular consultations with many government organisations, including the London Boroughs of Camden<sup>3</sup> and Lewisham<sup>4</sup>, Ordnance Survey<sup>5</sup> (OS), The Stationary Office<sup>6</sup> (TSO), The Met Office<sup>7</sup>, The Environment Agency<sup>8</sup>, The Office of National Statistics<sup>9</sup> (ONS), and several others.

To help focus the requests for data, information was collected from the geographical area covered by two of the participating London local authorities; Camden and Lewisham.

### 3.2 Design Decisions

The AKTivePSI project set out to deal with real data, plenty of it, and several, very busy, data providers, keen to find solutions to their knowledge problems. In such *real world* scenarios, it becomes vital to follow a realistic approach that is practical and inexpensive. To this end, the following decisions were made at the start of the project which turned out to have a very positive impact on the project as a whole:

- No disruption to the participants' existing data flows and models. A complete and sudden transition to semantic knowledge bases (KB) is unnecessary and impractical in the short term.
- Minimum cost to the participants. They provide the data, and we provide everything else (ontologies, KB infrastructure, tools for integration, etc.). Data to be delivered in any shape, format, and delivery method. No data preparation is required from the provider. Aim here is to encourage participation, and once the benefits of the SW become more apparent, they will be more willing to invest in this new technology. The outcomes of this project show that this approach has paid off very well.
- Simulate a real-life scenario. In other words, what we build and do can be done the same way outside our lab environment. For example, we treat the KBs as if hosted by the participants.
- Small, well focussed ontologies. It is not realistic to assume that an organisation will build one monolithic ontology for all their data, or that different organisations will agree on one semantic model. Therefore, a new ontology will be constructed for each dataset, and will be designed to represent *only* the data stored in this database, rather than the extended domains that the data might be related to (examples later). These numerous, small ontologies will be mapped together to form a small SW.

---

<sup>3</sup> <http://www.camden.gov.uk/>

<sup>4</sup> <http://www.lewisham.gov.uk/>

<sup>5</sup> <http://www.ordnancesurvey.co.uk/>

<sup>6</sup> <http://www.tso.co.uk/>

<sup>7</sup> <http://www.metoffice.gov.uk/>

<sup>8</sup> <http://www.environment-agency.gov.uk/>

<sup>9</sup> <http://www.statistics.gov.uk/>



- Data provenance must be preserved. Each dataset provided to us was transferred into a *separate* KB with its own ontology to eliminate any risks of data contamination from one database to another. Furthermore, each ontology contains a few classes and properties to represent the source of data, including name of supplier, name of data set, date supplied, etc. Source information is also attached to all triples when stored in the triple store.

## 4 Public Sector Datasets

Several organisations who participated in AKTivePSI made some of their databases available for the project. The data was provided in various formats, including Microsoft SQL databases, Microsoft Excel Spreadsheets, text-dumps from databases, XML files, and Microsoft Access spreadsheets. We developed a number of scripts to automatically convert this data to RDF, in correspondence with their designated ontologies. Table 1 lists the data sets that we used in this work, the number of RDF statements generated for each, and a brief description of the data.

**Table 1.** Datasets provided to AKTivePSI, the number of RDF triples we generated for each dataset, and a description of what the data is about

<b>Camden Borough Council</b>			
Land and Property Gazetteer	2.3M	Excel	Properties in Camden, full address, coordinates, type (residential/non-residential/mixed).
Food Premises	84K	Excel	Food related premises in Camden, their business names, hygiene inspection results, addresses, (eg restaurant, school, bar).
Local Businesses	170K	Excel	Businesses in Camden, names, addresses, contact info, and type of business.
Licences	100K	MSSQL	Licences for businesses in Camden, their addresses, licence types, and expiry dates.
Councillors and Committees	29K	Excel	Councillors and committees, sub committees, who sits on which committee, councillor's personal information.
Meeting Minutes	106K	Text	Web pages of committee's meeting minutes.
<b>Lewisham Borough Council</b>			
Land and Property Gazetteer	4M	Excel	Properties in Lewisham, their full addresses, and coordinates.
Property Tax Bands	10K	Excel	Tax property references, description, rate payers, rate value, and a one string addresses.
<b>Ordnance Survey (data for Camden and Lewisham only)</b>			
Address Layer 1	768K	XML	Data about buildings, addresses, and coordinates.
Address Layer 2	11.7M	XML	Data about buildings, addresses, and coordinates and building classifications (e.g. hospital, university).
PointX POI	467K	XML	Various landmarks and businesses, with names, addresses, and coordinates.
<b>The Stationery Office London Gazette (entire database was provided, but only the below was used)</b>			
Administration Notices	120K	Text	Notices for the appointment of administrator for corporate insolvencies.
Deceased Estates	3.2M	Text	Decease notices of individuals, names, addresses, description and date of death, address of representatives.

Once we receive a new database, we (1) design and build an ontology for this data, (2) convert the data to RDF triples and store in a triple store, and (3) map the data and ontology to our existing ontologies and KBs. These stages are described in the following sections.



## 4.1 Ontology Construction

Ontologies vary according to their formality levels, the purpose for which they are built, and the subject matter they represent [15]. One of the recommended first steps towards building an ontology is to scope its domain to make sure the ontology does not grow too large for what is needed [14] [16].

The appropriate size for an ontology depends on its purpose and the domain it represents. Some ontologies are designed to represent entire domains, and thus tend to be of very large sizes, such as the Gene Ontology (GO) [10], and Foundational Model of Anatomy (FMA) ontology [11]. Ontologies may also be built to serve the needs of specific applications and thus their sizes, though dependent on the needs of these applications, tend to be much smaller than the domain encapsulating ontologies. Other ontologies, as in our case, are data-dependent, where they are mainly built to represent a collection of data, to improve accessibility and understandability of the data. The scale of such dataset-specific ontologies is limited to the scope of the data.

As stated earlier, one of our principals for this project was to ensure the ontologies we build for the provided datasets are of low complexity and limited in scope and size. Small ontologies are cheaper and easier to build, maintain, understand, and use. In AKTivePSI, we found that most of the databases held by the participating organisation only required a small number of concepts and relationships to represent the stored data.

In AKTivePSI, we wanted to show that ontologies are not hard to build if limited to representing databases of defined scopes. We also wanted to show that it is not necessary to come to a common, agreed consensus on vocabulary, but that through ontology mapping techniques, locally-built ontologies can also prove very useful. Figure 1 shows an example of a ontology we have built, that describes, in very simple terms, the domain of Camden's Land and Property Gazetteer. In total, we constructed 13 ontologies, one for each dataset listed in table 1. All the ontologies were in OWL DL, and were mainly used to control vocabulary and to cross-link knowledge bases.

## 4.2 Generating RDF

From the ontology we are able to create instances by running simple scripts over the data to produce RDF. The scripts were hand-rolled specifically for the database and ontology which they were linking (reused across similar databases and ontologies). Although they were manually built, a framework for semi-automatic script generation would not be inconceivable. The scripts were highly reusable and hence were very easy to tune for new datasets and ontologies. We demonstrated to the participants the relative ease of converting legacy data to RDF using cheap and ordinary technology.

As shown in table 1, the total number of RDF triples that we generated for the government data exceeded 23 million. So although we needed small ontologies, we also needed scalable KB to hold all these RDF triples. We used the 3Store [6], an RDF triple-store developed in the AKT project, to store the generated RDF files. This triple-store provides a SPARQL endpoint, which is a servlet that accepts SPARQL queries and returns results in XML.

<sup>10</sup> <http://www.geneontology.org/>

<sup>11</sup> <http://sig.biostr.washington.edu/projects/fm/>



Fig. 1. Ontology for the Camden Land and Property Gazetteer

### 4.3 Webbing the Knowledge

One of main reasons for utilising ontologies is that the need for tight physical integrations between systems is removed [17]. Ontologies allow integration to happen using ‘soft’ mappings between concepts and instances that queries or data browsers can follow to find similar or duplicated entities. In our work, we used the special *owl:sameAs* property to link any mapped entities. By connecting our KBs in this way we are able to provide much greater flexibility and querying power than the original data structures could provide.

One main aim of this work is to show the added value of using SW technology for publishing and using government data. Forming a bigger semantic network by integrating the KBs containing all the participants’ data will add even more value to the data, and ease communication and data exchange between the partners.

We performed three levels of mappings:

- Mapping of local ontologies. It is safe to assume that individual organisations will know most about any of the ontologies they develop for their data, and hence it is possible for these local ontologies to be mapped to each other. For example, we developed two ontologies for datasets from Lewisham. Each ontology has

classes representing Property, Address, Post Code. These concepts were linked with owl:sameAs to indicate that they represent the same concepts. Another example is mapping the concept Premises from the Food Premises ontology of Camden to the Property class in the Land and Property ontology of Camden. To semi-automate these mappings, we used CROSI [7], a freely available tool that offers a wide choice of mapping algorithms.

- Mapping of instances. Because we are using a data-centric approach, it was very useful to map the instance data to each other as well. For example the instance *post-code\_N6\_6DS* in one KB maps to the instance *pc\_N66DS* in another. Since these instances really do refer to the same object we are able to infer much more data about certain objects that refer to this instance. In fact, we found that simply linking on one data object (the postcode) was enough to glean useful information from various datasets to such an extent that mashups are made easier (section 4.5). Instance mappings were done automatically using simple scripts that search for duplicates of specific type of instance (e.g. postcodes, streets, councillors). An owl:sameAs link will be automatically added between the corresponding instances once such a mapping is found.
- Mapping of local ontologies to the government reference taxonomy; IPSV. IPSV (the Integrated Public Sector Vocabulary) is a “structured list of terms for the Subject metadata of public sector resources” [2]. UK e-Government Metadata Standard requires public sector organisations to comply with IPSV. AKTivePSI partners expressed some difficulties mapping their databases to IPSV, and hence part of this project was to explore this taxonomy and assess its suitability for this task. To better understand the problem, we manually mapped our ontologies to the best matched terms in the IPSV.

#### 4.4 Exploring the Knowledge Network

Now that all the data is ontologically represented and stored in KBs, we need to demonstrate to the participating government organisations what and where the added value is.

RDF provides a well-understood grounding on which data may be shared, and this in itself provides added value, such that re-use of the data is made much easier (see section 4.5 on Mash-ups).

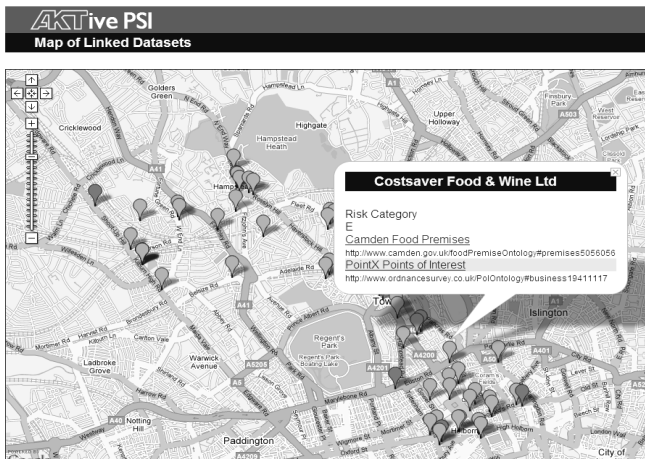
#### 4.5 Mashing-Up Distributed KBs

Once data is available in easily parsable and understandable formats, such as RDF, mash-ups become much easier to generate by searching RDF KBs and mashing-up data on the fly, which is one of the advantages the SW promises. Two examples of such mash-ups were created in AKTivePSI. The aim of building these mash-ups was to demonstrate the relative ease with which they can be constructed from semantically represented knowledge.

The Camden Food Premises database gives information about the hygiene check results and health risk of various premises around the Camden area that handle food. The risk categories are given a level between A, which is high risk, to E which is low risk, and is based on the cleanliness of the premises, compliance with regulations, type of

preparation that is performed, etc. The Food Premises database contains lots of information on these properties, but displaying this information on a map is difficult because the geographical co-ordinates are missing from this particular data set.

However, the Ordnance Survey's Address Layer and Points of Interest (PointX) datasets contain easting and northing coordinates for businesses and properties. The instance mapping of postcodes we performed earlier helped to cut down our search space for finding matching addresses in the datasets. Indeed, once we had found matches we were able to assert them as being the same, thereby avoiding the need for searching again.



**Fig. 2.** Google Maps mashup of the Camden Food Premises dataset made possible by mapping the data to the OS Address Layer II and PointX dataset

To create the mash-up, a number of SPARQL queries were written that searched for each premises' address from the Food Premises dataset in each of the OS two datasets and once a match is found the co-ordinates are retrieved and the premises is displayed on a Google map. The information from Food Premises along with the mapping performed between one dataset and another, provides extra context to instances from both datasets. The PointX dataset gains access to the risk level of the food premises (as well as the implicit knowledge that the premises are used for preparing food), and the food premises dataset garnered exact coordinates for the premises. Figure 2 shows a simple Google Maps mash-up that uses the mapping to provide a visual display of the food premises dataset.

This type of mash-up could be very good for public awareness (and therefore commercial competition). For example, one particular business that scored within the high risk category, has glowing customer reviews on restaurant review sites across the internet.

As for Lewisham, we were able to use the PointX dataset for a similar use for the Lewisham Land and Property Gazetteer. This dataset contains information about all kinds of properties across the Lewisham ward, and includes address and coordinate

information; however, it does not contain information about the business inhabiting a property – information that the PointX data does provide. We provided a mash-up that shows the location of business properties

## 5 Findings

Introducing a new technology, such as the SW to any organisation must be managed very carefully to avoid any disruption to their current work procedures and data flow, and to gain their trust and interest in the new technology. Below are some of the findings and lessons learnt from the AKTivePSI study that relates more specifically to government agencies.

**Minimise disruption to existing infrastructure:** It was critical to show that adopting SW technology does not mean throwing away existing database technologies; the re-engineering of governmental information processing is a complex and difficult task that is facilitated by special conditions and structures that do not obtain in the United Kingdom [11]. An important part of our task was to show that the costs of SW adoption were relatively low. We demonstrated how simple scripts can be used to convert all their data into RDF triples. The approach we adopted in AKTivePSI was to *cache* the given databases into triple stores. However, this might not be the best solution as it duplicates existing databases. A slightly different approach is suggested in [10], where they imported the data into ontologies, then exported it back into relational databases with new structures that are closer to those of the used ontologies. In other words, they changed the database schema to match the ontology. We are now experimenting with an alternative approach, which is to use a technology like D2RQ<sup>12</sup> which enables layering an ontology on top of a non-RDF database, thus removing the need to duplicate or change the structure of the original database. Such technology maintains the benefits of scalability and maturity of RDBMS, as well as providing RDF and SPARQL access points.

**Minimal ontological commitment:** Constructing ontologies requires certain skills and modelling knowledge and expertise. Government organisations worry about the possible high cost of building such complex knowledge structures.

We were encouraged by the results of applying SW technologies to governmental data. Not only were the benefits very high, even on a small quantity of data, but the costs were relatively low. The building of small, data-centric ontologies was an easily achievable goal for developers in governmental institutions, possibly working on limited budgets. This is a lesson that is of interest in the wider Semantic Web field, as arguments continue over the overhead that developing and maintaining ontologies will require. As a result of this work, Camden Council and the London Gazette are now developing their own ontologies to represent some of their datasets.

Some of these organisations thought that Cyc, Gene Ontology, or even IPSV, are the sort of ontologies they need to build to become semantically enabled. We demonstrated a cheap and practical approach, where ontologies are scaled to individual datasets rather

---

<sup>12</sup> <http://sourceforge.net/projects/d2rq-map/>

than to entire domains, then *gradually* linked together to enable data sharing. It is possible that more elaborated ontologies might be required later on if more automation is needed for ontology mapping or for data inconsistency checking.

**Extending IPSV:** One of the initial concerns that some AKTivePSI participants had was the difficulty they were facing in mapping their data collections to IPSV (section 4.3). During our investigation, we found that IPSV is mainly designed to represent subject topics, not data. For example, IPSV contains more than 30 terms related to road issues (e.g. *Road safety*, *Road signs*, *Road cleaning*), but there is not a “Road” term to map a specific road to. Our conclusion was that IPSV is simply not designed to be a reference ontology for representing data and hence a different reference ontology, or an extension to IPSV, for mapping and sharing data. In AKTivePSI, we mapped each ontology to IPSV to demonstrate how IPSV can be extended to cover the required semantics.

The ability to map ontologies together provided a much more practical and less expensive alternative to agreeing or using the same terminology, which some government organisations thought was required to share data. They realised that it is possible to continue using their local terminologies whilst being able to open data exchange channels between different, distributed, databases.

**Showing added value:** The goal of providing better access to data is naturally not enough to win the interest, support, and active participation of data providers. It was vital to show examples of where and what is the added value of integration and shared access. Most of the organisations we met with had some needs, and sometimes laborious procedures, for acquiring data from other government sources. We illustrated the direct benefits of participating in a semantically enabled data exchange channel, especially with respect to data consistency checking, relative ease of integration and distributed querying, data exchange and merging, and lowering the cost of meeting the requests of the public for data as well as the requests of the government for providing better access to public sector information.

Data integration from multiple sources adds the value of knowledge augmentation and verification. Integrating datasets can provide useful insights into the quality of the dataset for the data provider involved. For example, the Ordnance Survey’s *Address Layer 2* dataset provides a list of businesses, including their address and their geo-location, and similarly so does the PointX dataset. However, we found that the two lists of businesses do not match, where some are present in one dataset but not in the other. In some examples, the PointX dataset contained several businesses listed at the same address, while only one was listed in the OS Address Layer 2. Was this an error? Perhaps, due to the lack of temporal information, one business took over the building from another, or perhaps one business is sited in the same building on a different floor to another business. It is difficult to infer an answer, but the integration has provided some information about the quality of the datasets and made such comparisons and cross-matchings possible.

This, of course, applies equally to errors and inconsistencies in the datasets, as well as knowledge gaps. Table 2 gives an overview of one of the examples of inconsistencies in the datasets. The Sunrise Food Mart appears in the PointX database at number 354,

whereas it occupies a number of building plots and is called Sunrise Food Market in the Camden Food Premises. The Ordnance Survey has an error in its naming of the business, and says the business is at number 352, with some courier service at 354. Such inconsistencies cannot easily be automatically resolved, unless a number of other linked datasets are able to provide evidence that supports one or other of the possible addresses. All three sources are official and trusted, and hence not one can be taken as necessarily the correct one.

**Table 2.** Inconsistencies on one entity highlighted by the integration

Name	Number	Postcode	Dataset
Sunrise Food Mart	354	NW62QJ	PointX
Sunrise Food Market	352-354	NW62QJ	Camden Food Premises
Sunrise Food Mart, 352	352	NW62QJ	Ordnance Survey Address Layer 2
London No.1 Courier	354	NW62QJ	Ordnance Survey Address Layer 2

As well as spotting many knowledge overlaps, we also identified several knowledge gaps between various participants. For example, the OS desires to get automatic feeds about accepted applications for property extensions from local councils, local councils need to receive automatic notifications from Land Registry when a property changes hands, and local councils in London would like to know when a business publishes its insolvency notice in the London Gazette. Although AKTivePSI did not implement any of these capabilities, but it showed how the SW can support such processes. Some of these services will be implemented in the second phase of AKTivePSI due to start in the coming few weeks.

**Provenance and Privacy:** Many agencies and institutions are instinctively secretive about their data. The SW vision is to remove human processing from the knowledge acquisition process as far as is feasible, and the idea of publishing data without even controlling the context of its presentation is of course very new in governmental circles.

At present, the *ideal* limits to data publication are unknown. A number of agencies lack understanding of what data they actually possess. These agencies needed to be assured that with SW technology, they will be able to *pick and choose* which data to share and which data to keep locked-up.

Some of AKTivePSI government participants expressed their great unease and worry about possible misuse of the data, once access and reuse are enabled with the SW. Privacy is a complex issue, with post-Enlightenment concepts under technological threat from a number of directions, not only government. Many of us are prepared to surrender our privacy for gains in efficiency or monetary benefit; others defend personal privacy as a vital pillar of a liberal democratic society. Unless and until such political dilemmas are resolved, governments will of necessity have to tread carefully when considering how far to exploit information-processing technologies such as the Semantic Web [12]. Technologies and protocols currently under development in the W3C to create a *policy-aware Web*, allowing information users, owners and subjects to express policies for information use and negotiate about them, will help make the situation clearer [20].



## 6 Conclusions and Future Work

The adoption of Semantic Web technology to allow for more efficient use of data in order to add value is becoming more common where efficiency and value-added are important parameters, for example in business and science. However, in the field of government there are other parameters to be taken into account (e.g. confidentiality), and the cost/benefit analysis is more complex. The work reported here was intended to show that SW technology could be valuable in the governmental context.

An important outcome of the project is the level of awareness that has been built up in government about the potential of SW technology. Having seen what the SW technology is capable of, and the success of the pilot study of AKTivePSI, OPSI is now funding a second project which will focus on implementing and running some of the services and capabilities studied in the first stage at the premises of some of the participating government agencies.

Some of the direct outcomes of this work are: (a) the London Gazette is currently building OWL ontologies to represent parts of their data, and is working towards publishing this data in RDF; (b) OPSI oversaw the development of a URI schema, which is now being used to generate URIs for government official legislations and copyright statements; and (c) Camden Borough Council added a SW engineer to their staff force to help the council in their effort to join the SW.

AKTivePSI has given a glimpse of what is possible by applying SW technology to public sector information. We showed that by using small, purpose-built ontologies and mapping these together, greater value can be sought in the data, and re-use of the data in mash-ups becomes much easier, which should increase public awareness and access to the data.

The issue of providing better access to public sector information has also been identified in the policy review, “The Power of Information”, conducted by Prime Minister’s Strategy Unit<sup>13</sup>. The review aims to position the UK Government in response to developments in the use and communication of citizen and state generated information on the web. The work described in this paper predates that review and helped inform the review team’s analysis. Of particular note is the proposal to link the social power of the web to help address the re-usable format issue, by providing citizens with an on-line facility for people to come together, to discuss and formally request public sector information assets in a particular format.

The commercial re-use of public sector information that the SW enables, opens up countless opportunities for the development of new information products and services, driving forwards and accelerating the development of the knowledge economy.

## References

1. Cabinet Office, Transformational Government: Enabled by Technology, Crown Copyright, Cm 6683 (2005)
2. Clarke, S.D.: Guide to meta-tagging with the IPSV. In: Instructional, e-Government Unit, Cabinet Office, UK (2005), <http://www.esd.org.uk/documents/ipsvhowtometatag.pdf>

<sup>13</sup> <http://www.cabinetoffice.gov.uk/strategy/>



3. Dimitrov, D.A., Heflin, J., Qasem, A., Wang, N.: Information integration via an end-to-end distributed semantic web system. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 764–777. Springer, Heidelberg (2006)
4. Duke, A., Richardson, M., Watkins, S., Roberts, M.: Towards B2B integration in telecommunications with semantic web services. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 710–724. Springer, Heidelberg (2005)
5. Halevy, A.Y., Ashish, N., Bitton, D., Carey, M., Draper, D., Pollock, J., Rosenthal, A., Sikka, V.: Enterprise information integration: Successes, challenges and controversies. In: Proc. ACM SIGMOD Int. Conf. on Management of data, pp. 778–787. ACM Press, New York (2005)
6. Harris, S., Gibbins, N.: 3Store: Efficient bulk RDF storage. In: PSSS 03. Proc. 1st Int. Workshop on Practical and Scalable Semantic Systems, Sanibel Island, FL, USA, pp. 1–20 (2003)
7. Kalfoglou, Y., Hu, B., Reynolds, D., Shadbolt, N.: CROSI: Capturing, Representing, and Operationalising Semantic Integration. Technical Report 11717, University of Southampton, Southampton, UK (2005)
8. Klischewski, R.: Migrating small governments websites to the semantic web. In: AAAI 2006. Proc. The Semantic Web meets eGovernment, Stanford University, California (March 2006)
9. Klischewski, R., Jeenicke, M.: Semantic web technologies for information management within e-government services. In: Proc. of 37th Hawaii Conference on System Sciences 2004 (2004)
10. Maier, A., Schnurr, H.-P., Sure, Y.: Ontology-based information integration in the automotive industry. In: Proc. 2nd Int. Semantic Web Conf., Sanibel Island, FL, USA, pp. 897–912 (2003)
11. O'Hara, K., Stevens, D.: Democracy, ideology and process re-engineering: realising the benefits of e-government in singapore. In: Proc. WWW06 Workshop on e-Government: Barriers and Opportunities, Edinburgh (2006)
12. O'Hara, K., Stevens, D.: inequality.com: Power, Poverty and the Digital Divide, pp. 243–271. Oneworld, Oxford (2006)
13. Preist, C., Esplugas-Cuadrado, J., Battle, S.A., Grimm, S., Willieams, S.K.: Automated business-to-business integration of a logistics supply chain using semantic web services technology. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 987–1001. Springer, Heidelberg (2005)
14. Skuce, D.: Conventions for reaching agreement on shared ontologies. In: Proc. 9th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff Conference Centre, Banff, Alberta, Canada (1995)
15. Uschold, M.: Building ontologies: Towards a unified methodology. In: Proc. of 16th Conf. Expert Systems, Cambridge, UK (1996)
16. Uschold, M., Gruninger, M.: Ontologies: principles, methods and applications. The Knowledge Engineering Review 11(2), 93–136 (1996)
17. Uschold, M., Gruninger, M.: Ontologies and semantics for seamless connectivity. SIGMOD Record 33(4) (2004)
18. Van Elst, L., Klein, B., Maus, H., Schoning, H., Tommasi, A., Zavattari, C., Favaro, J., Giannella, V.: Business register interoperability throughout europe: The brite project. In: AAAI 2006. Proc. The Semantic Web meets eGovernment, Stanford University, California (March 2006)
19. Vitvar, T., Mocan, A., Peristeras, V.: Pan-european e-government services on the semantic web services. In: Proc. WWW 2006 (May 2006)
20. Weitzner, D., Hendlar, J., Berners-Lee, T., Connolly, D.: Creating a policy-aware Web: discretionary, rule-based access for the World Wide Web. In: Ferrari, E., Thuraisingham, B. (eds.) Web and Information Security, Idea Group Inc., Hershey, PA (2005)

# DBpedia: A Nucleus for a Web of Open Data

Sören Auer<sup>1,3</sup>, Christian Bizer<sup>2</sup>, Georgi Kobilarov<sup>2</sup>, Jens Lehmann<sup>1</sup>,  
Richard Cyganiak<sup>2</sup>, and Zachary Ives<sup>3</sup>

<sup>1</sup> Universität Leipzig, Department of Computer Science, Johannsgasse 26,  
D-04103 Leipzig, Germany

{[auer,lehmann](mailto:auer@informatik.uni-leipzig.de)}@informatik.uni-leipzig.de

<sup>2</sup> Freie Universität Berlin, Web-based Systems Group, Garystr. 21,  
D-14195 Berlin, Germany

[chris@bizer.de](mailto:chris@bizer.de), [georgi.kobilarov@gmx.de](mailto:georgi.kobilarov@gmx.de) [richard@cyganiak.de](mailto:richard@cyganiak.de)

<sup>3</sup> University of Pennsylvania, Department of Computer and Information Science  
Philadelphia, PA 19104, USA

[auer@seas.upenn.edu](mailto:auer@seas.upenn.edu), [zives@cis.upenn.edu](mailto:zives@cis.upenn.edu)

**Abstract.** DBpedia is a community effort to extract structured information from Wikipedia and to make this information available on the Web. DBpedia allows you to ask sophisticated queries against datasets derived from Wikipedia and to link other datasets on the Web to Wikipedia data. We describe the extraction of the DBpedia datasets, and how the resulting information is published on the Web for human- and machine-consumption. We describe some emerging applications from the DBpedia community and show how website authors can facilitate DBpedia content within their sites. Finally, we present the current status of interlinking DBpedia with other open datasets on the Web and outline how DBpedia could serve as a nucleus for an emerging Web of open data.

## 1 Introduction

It is now almost universally acknowledged that stitching together the world’s structured information and knowledge to answer semantically rich queries is one of the key challenges of computer science, and one that is likely to have tremendous impact on the world as a whole. This has led to almost 30 years of research into information integration [15,19] and ultimately to the Semantic Web and related technologies [11,13]. Such efforts have generally only gained traction in relatively small and specialized domains, where a closed ontology, vocabulary, or schema could be agreed upon. However, the broader Semantic Web vision has not yet been realized, and one of the biggest challenges facing such efforts has been how to get enough “interesting” and broadly useful information into the system to make it useful and accessible to a *general* audience.

A challenge is that the traditional “top-down” model of designing an ontology or schema *before* developing the data breaks down at the scale of the Web: both data and metadata must constantly evolve, and they must serve many different communities. Hence, there has been a recent movement to build the Semantic Web grass-roots-style, using incremental and Web 2.0-inspired collaborative

approaches [10,12,13]. Such a collaborative, grass-roots Semantic Web requires a new model of structured information representation and management: first and foremost, it must handle inconsistency, ambiguity, uncertainty, data provenance [3,6,8,7], and implicit knowledge in a uniform way.

Perhaps the most effective way of spurring synergistic research along these directions is to provide a rich corpus of diverse data. This would enable researchers to develop, compare, and evaluate different extraction, reasoning, and uncertainty management techniques, and to deploy operational systems on the Web.

The DBpedia project has derived such a data corpus from the Wikipedia encyclopedia. Wikipedia is heavily visited and under constant revision (e.g., according to alexa.com, Wikipedia was the 9th most visited website in the third quarter of 2007). Wikipedia editions are available in over 250 languages, with the English one accounting for more than 1.95 million articles. Like many other web applications, Wikipedia has the problem that its search capabilities are limited to full-text search, which only allows very limited access to this valuable knowledge base. As has been highly publicized, Wikipedia also exhibits many of the challenging properties of collaboratively edited data: it has contradictory data, inconsistent taxonomical conventions, errors, and even spam.

The DBpedia project focuses on the task of converting Wikipedia content into structured knowledge, such that Semantic Web techniques can be employed against it — asking sophisticated queries against Wikipedia, linking it to other datasets on the Web, or creating new applications or mashups. We make the following contributions:

- We develop an information extraction framework, which converts Wikipedia content to RDF. The basic components form a foundation upon which further research into information extraction, clustering, uncertainty management, and query processing may be conducted.
- We provide Wikipedia content as a large, multi-domain RDF dataset, which can be used in a variety of Semantic Web applications. The DBpedia dataset consists of 103 million RDF triples.
- We interlink the DBpedia dataset with other open datasets. This results in a large Web of data containing altogether around 2 billion RDF triples.
- We develop a series of interfaces and access modules, such that the dataset can be accessed via Web services and linked to other sites.

The DBpedia datasets can be either imported into third party applications or can be accessed online using a variety of DBpedia user interfaces. Figure 1 gives an overview about the DBpedia information extraction process and shows how extracted data is published on the Web. These main DBpedia interfaces currently use Virtuoso [9] and MySQL as storage back-ends.

The paper is structured as follows: We give an overview about the DBpedia information extraction techniques in Section 2. The resulting datasets are described in Section 3. We exhibit methods for programmatic access to the DBpedia dataset in Section 4. In Sections 5 we present our vision of how the DBpedia

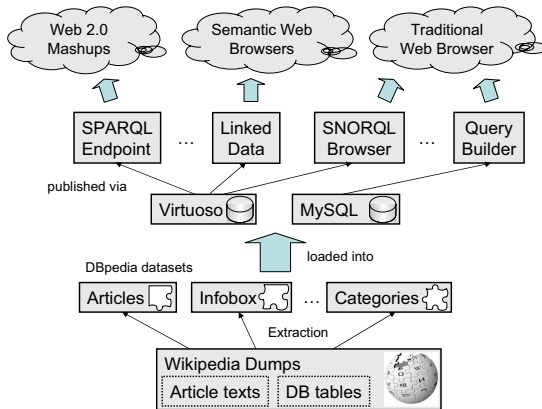


Fig. 1. Overview of the DBpedia components

datasets can be a nucleus for a Web of open data. We showcase several user interfaces for accessing DBpedia in Section 6 and finally review related work in Section 7.

## 2 Extracting Structured Information from Wikipedia

Wikipedia articles consist mostly of free text, but also contain different types of structured information, such as infobox templates, categorisation information, images, geo-coordinates, links to external Web pages and links across different language editions of Wikipedia.

Mediawiki<sup>1</sup> is the software used to run Wikipedia. Due to the nature of this Wiki system, basically all editing, linking, annotating with meta-data is done inside article texts by adding special syntactic constructs. Hence, structured information can be obtained by parsing article texts for these syntactic constructs.

Since MediaWiki exploits some of this information itself for rendering the user interface, some information is cached in relational database tables. Dumps of the crucial relational database tables (including the ones containing the article texts) for different Wikipedia language versions are published on the Web on a regular basis<sup>2</sup>. Based on these database dumps, we currently use two different methods of extracting semantic relationships: (1) We map the relationships that are already stored in relational database tables onto RDF and (2) we extract additional information directly from the article texts and infobox templates within the articles.

We illustrate the extraction of semantics from article texts with an Wikipedia infobox template example. Figure 2 shows the infobox template (encoded within a Wikipedia article) and the rendered output of the South-Korean town

<sup>1</sup> <http://www.mediawiki.org>

<sup>2</sup> <http://download.wikimedia.org/>

```

{{infobox City Korea|
  full_name=Busan Metropolitan City|
  image=[[Image:Haeundaebeachbusan.jpg|
    250px|Haeundae Beach, Busan]]|
  rr=Busan Gwangyeoksi|
  mr=Pusan Kwangyŏksi|
  hangul=부산 광역시|
  hanja=釜山廣域市|
  short_name=Busan (Pusan; 부산; 釜山)|
  population=3,635,389 ...|
  area=763.46 km²|
  government=[[Metropolitan cities of
    South Korea|Metropolitan City]]|
  divisions=15 wards (Gu),
  <br>1 county (Gun)|
  region=[[Yeongnam]]|
  dialect=[[Gyeongsang Dialect|
    Gyeongsang]]|
  map=[[Image:Busan map.png|Map of
    South Korea highlighting the city]]
}}

```

Busan Metropolitan City	
	
Korean name	
Revised Romanization	Busan Gwangyeoksi
McCune-Reischauer	Pusan Kwangyŏksi
Hangul	부산 광역시
Hanja	釜山廣域市
Short name	Busan (Pusan; 부산; 釜山)

Fig. 2. Example of a Wikipedia template and rendered output (excerpt)

Busan. The infobox extraction algorithm detects such templates and recognizes their structure using pattern matching techniques. It selects significant templates, which are then parsed and transformed to RDF triples. The algorithm uses post-processing techniques to increase the quality of the extraction. MediaWiki links are recognized and transformed to suitable URIs, common units are detected and transformed to data types. Furthermore, the algorithm can detect lists of objects, which are transformed to RDF lists. Details about the infobox extraction algorithm (including issues like data type recognition, cleansing heuristics and identifier generation) can be found in [2]. All extraction algorithms are implemented using PHP and are available under an open-source license<sup>3</sup>.

### 3 The DBpedia Dataset

The DBpedia dataset currently provides information about more than 1.95 million "things", including at least 80,000 persons, 70,000 places, 35,000 music albums, 12,000 films. It contains 657,000 links to images, 1,600,000 links to relevant external web pages, 180,000 external links into other RDF datasets, 207,000 Wikipedia categories and 75,000 YAGO categories [16].

DBpedia concepts are described by short and long abstracts in 13 different languages. These abstracts have been extracted from the English, German,

<sup>3</sup> <http://sf.net/projects/dbpedia>

French, Spanish, Italian, Portuguese, Polish, Swedish, Dutch, Japanese, Chinese, Russian, Finnish and Norwegian versions of Wikipedia.

Altogether the DBpedia dataset consists of around 103 million RDF triples. The dataset is provided for download as a set of smaller RDF files. Table 1 gives an overview over these files.

**Table 1.** The DBpedia datasets

<b>Dataset</b>	<b>Description</b>	<b>Triples</b>
<i>Articles</i>	Descriptions of all 1.95 million concepts within the English Wikipedia including titles, short abstracts, thumbnails and links to the corresponding articles.	7.6M
<i>Ext. Abstracts</i>	Additional, extended English abstracts.	2.1M
<i>Languages</i>	Additional titles, short abstracts and Wikipedia article links in German, French, Spanish, Italian, Portuguese, Polish, Swedish, Dutch, Japanese, Chinese, Russian, Finnish and Norwegian.	5.7M
<i>Lang. Abstracts</i>	Extended abstracts in 13 languages.	1.9M
<i>Infoboxes</i>	Data attributes for concepts that have been extracted from Wikipedia infoboxes.	15.5M
<i>External Links</i>	Links to external web pages about a concept.	1.6M
<i>Article Categories</i>	Links from concepts to categories using SKOS.	5.2M
<i>Categories</i>	Information which concept is a category and how categories are related.	1M
<i>Yago Types</i>	Dataset containing rdf:type Statements for all DBpedia instances using classification from YAGO [16].	1.9 M
<i>Persons</i>	Information about 80,000 persons (date and place of birth etc.) represented using the FOAF vocabulary.	0.5M
<i>Page Links</i>	Internal links between DBpedia instances derived from the internal pagelinks between Wikipedia articles.	62M
<i>RDF Links</i>	Links between DBpedia and Geonames, US Census, Musicbrainz, Project Gutenberg, the DBLP bibliography and the RDF Book Mashup.	180K

Some datasets (such as the *Persons* or *Infoboxes* datasets) are semantically rich in the sense that they contain very specific information. Others (such as the *PageLinks* dataset) contain meta-data (such as links between articles) without a specific semantics. However, the latter can be beneficial, e.g. for deriving measures of closeness between concepts or relevance in search results.

Each of the 1.95 million resources described in the DBpedia dataset is identified by a URI reference of the form `http://dbpedia.org/resource/Name`, where *Name* is taken from the URL of the source Wikipedia article, which has the form `http://en.wikipedia.org/wiki/Name`. Thus, each resource is tied directly to an English-language Wikipedia article. This yields certain beneficial properties to DBpedia identifiers:

- They cover a wide range of encyclopedic topics,
- They are defined by community consensus,

- There are clear policies in place for their management,
- And an extensive textual definition of the concept is available at a well-known web location (the Wikipedia page).

## 4 Accessing the DBpedia Dataset on the Web

We provide three access mechanisms to the DBpedia dataset: Linked Data, the SPARQL protocol, and downloadable RDF dumps. Royalty-free access to these interfaces is granted under the terms of the GNU Free Documentation License.

*Linked Data.* Linked Data is a method of publishing RDF data on the Web that relies on `http://` URIs as resource identifiers and the HTTP protocol to retrieve resource descriptions [4,5]. The URIs are configured to return meaningful information about the resource—typically, an RDF description containing everything that is known about it. Such a description usually mentions related resources by URI, which in turn can be accessed to yield their descriptions. This forms a dense mesh of web-accessible resource descriptions that can span server and organization boundaries. DBpedia resource identifiers, such as <http://dbpedia.org/resource/Busan>, are set up to return RDF descriptions when accessed by Semantic Web agents, and a simple HTML view of the same information to traditional web browsers (see Figure 3). HTTP content negotiation is used to deliver the appropriate format.

Web agents that can access Linked Data include: 1. Semantic Web browsers like Disco<sup>4</sup>, Tabulator<sup>[7]</sup> (see Figure 3), or the OpenLink Data Web Browser<sup>5</sup>; 2. Semantic Web crawlers like SWSE<sup>6</sup> and Swoogle<sup>7</sup>; 3. Semantic Web query agents like the Semantic Web Client Library<sup>8</sup> and the SemWeb client for SWI prolog<sup>9</sup>.

*SPARQL Endpoint.* We provide a SPARQL endpoint for querying the DBpedia dataset. Client applications can send queries over the SPARQL protocol to this endpoint at <http://dbpedia.org/sparql>. This interface is appropriate when the client application developer knows in advance exactly what information is needed. In addition to standard SPARQL, the endpoint supports several extensions of the query language that have proved useful for developing user interfaces: full text search over selected RDF predicates, and aggregate functions, notably COUNT. To protect the service from overload, limits on query cost and result size are in place. For example, a query that asks for the store’s entire contents is rejected as too costly. SELECT results are truncated at 1000 rows. The SPARQL endpoint is hosted using Virtuoso Universal Server<sup>10</sup>.

<sup>4</sup> <http://sites.wiwiss.fu-berlin.de/suhl/bizer/ng4j/disco/>

<sup>5</sup> <http://demo.openlinksw.com/DAV/JS/rdfbrowser/index.html>

<sup>6</sup> <http://swse.org>

<sup>7</sup> <http://swoogle.umbc.edu/>

<sup>8</sup> <http://sites.wiwiss.fu-berlin.de/suhl/bizer/ng4j/semwebclient/>

<sup>9</sup> <http://moustaki.org/swic/>

<sup>10</sup> <http://virtuoso.openlinksw.com>



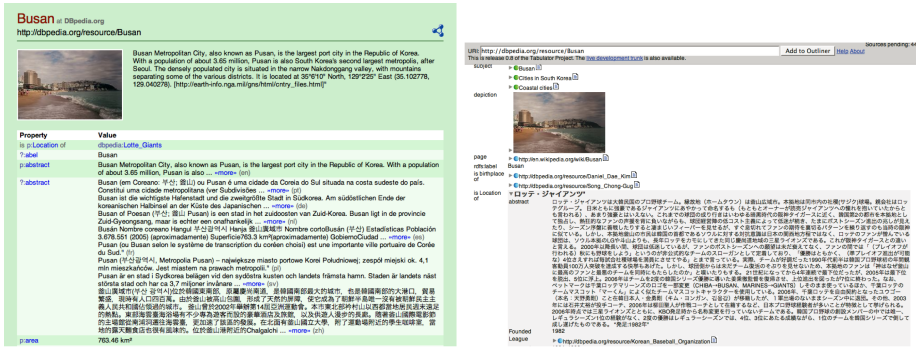


Fig. 3. <http://dbpedia.org/resource/Busan> viewed in a web browser (left) and in Tabulator (right)

*RDF Dumps.* N-Triple serializations of the datasets are available for download at the DBpedia website and can be used by sites that are interested in larger parts of the dataset.

## 5 Interlinking DBpedia with Other Open Datasets

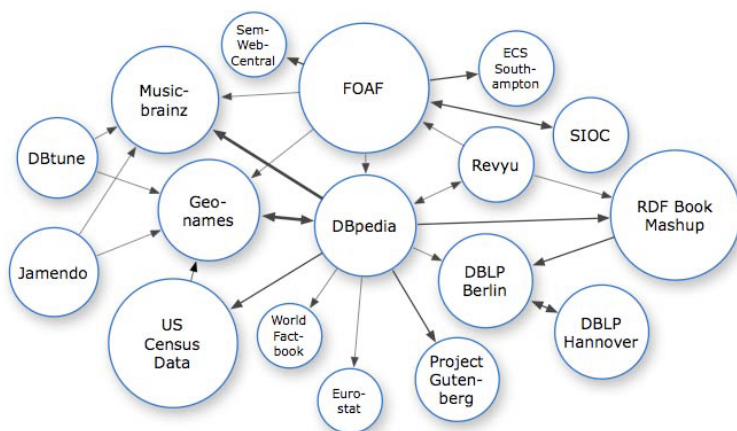
In order to enable DBpedia users to discover further information, the DBpedia dataset is interlinked with various other data sources on the Web using RDF links. RDF links enable web surfers to navigate from data within one data source to related data within other sources using a Semantic Web browser. RDF links can also be followed by the crawlers of Semantic Web search engines, which may provide sophisticated search and query capabilities over crawled data.

The DBpedia interlinking effort is part of the Linking Open Data community project [1] of the W3C Semantic Web Education and Outreach (SWEO) interest group. This community project is committed to make massive datasets and ontologies, such as the US Census, Geonames, MusicBrainz, the DBLP bibliography, WordNet, Cyc and many others, interoperable on the Semantic Web. DBpedia, with its broad topic coverage, intersects with practically all these datasets and therefore makes an excellent “linking hub” for such efforts.

Figure 4 gives an overview about the datasets that are currently interlinked with DBpedia. Altogether this Web-of-Data amounts to approximately 2 billion RDF triples. Using these RDF links, surfers can for instance navigate from a computer scientist in DBpedia to her publications in the DBLP database, from a DBpedia book to reviews and sales offers for this book provided by the RDF Book Mashup, or from a band in DBpedia to a list of their songs provided by Musicbrainz or dbtune.

<sup>11</sup> <http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>





**Fig. 4.** Datasets that are interlinked with DBpedia

The example RDF link shown below connects the DBpedia URI identifying Busan with further data about the city provided by Geonames:

```
<http://dbpedia.org/resource/Busan>
  owl:sameAs <http://sws.geonames.org/1838524/> .
```

Agents can follow this link, retrieve RDF from the Geonames URI, and thereby get hold of additional information about Busan as published by the Geonames server, which again contains further links deeper into the Geonames data. DBpedia URIs can also be used to express personal interests, places of residence, and similar facts within personal FOAF profiles:

```
<http://richard.cyganiak.de/foaf.rdf#cygri>
  foaf:topic_interest <http://dbpedia.org/resource/Semantic_Web> ;
  foaf:based_near <http://dbpedia.org/resource/Berlin> .
```

Another use case is categorization of blog posts, news stories and other documents. The advantage of this approach is that all DBpedia URIs are backed with data and thus allow clients to retrieve more information about a topic:

```
<http://news.cnn.com/item1143>
  dc:subject <http://dbpedia.org/resource/Iraq_War> .
```

## 6 User Interfaces

User interfaces for DBpedia can range from a simple table within a classic web page, over browsing interfaces to different types of query interfaces. This section gives an overview about the different user interfaces that have been implemented so far.

## 6.1 Simple Integration of DBpedia Data Into Web Pages

DBpedia is a valuable source of general-purpose data that can be used within web pages. Therefore, if you want a table containing German state capitals, African musicians, Amiga computer games or whatever on your website, you can generate this table using a SPARQL query against the DBpedia endpoint. Wikipedia is kept up-to-date by a large community and a nice feature of such tables is that they will also stay up-to-date as Wikipedia, and thus also DBpedia, changes. Such tables can either be implemented using Javascript on the client or with a scripting language like PHP on the server. Two examples of Javascript generated tables are found on the DBpedia website<sup>12</sup>.

## 6.2 Search DBpedia.org

*Search DBpedia.org* is a sample application that allows users to explore the DBpedia dataset together with information from interlinked datasets such as Geonames, the RDF Book Mashup or the DBLP bibliography. In contrast to the keyword-based full-text search commonly found on the Web, search over structured data offers the opportunity to make productive use of the relations in the data, enabling stepwise narrowing of search results in different dimensions. This adds a browsing component to the search task and may reduce the common “keyword-hit-or-not-hit” problem.

A *Search DBpedia.org* session starts with a keyword search. A first set of results is computed by direct keyword matches. Related matches are added, using the relations between entities up to a depth of two nodes. Thus, a search for the keyword “Scorsese” will include the director Martin Scorsese, as well as all of his films, and the actors of these films.

The next step is result ranking. Our experiments showed that important articles receive more incoming page links from other articles. We use a combination of incoming link count, relevance of the link’s source, and relation depth to calculate a relevance ranking.

After entering a search term, the user is presented with a list of ranked results, and with a tag cloud built from the classes found in the results, using a combination of the DBpedia and YAGO [16] classifications. Each class weight is calculated from the sum of associated result weights and the frequency of occurrence. The tag cloud enables the user to narrow the results to a specific type of entities, such as “Actor”, even though a simple keyword search may not have brought up any actors.

When a resource from the results is selected, the user is presented with a detailed view of all data that is known about the resource. Label, image and description are shown on top. Single-valued and multi-valued properties are shown separately. Data from interlinked datasets is automatically retrieved by following RDF links within the dataset and retrieved data from interlinked datasets is shown together with the DBpedia data.

<sup>12</sup> <http://dbpedia.org>

The screenshot displays the DBpedia search interface. At the top, there are two search bars with the text 'Search DBpedia.org' and 'Explore Linked Data'. Below the search bars, there are navigation links for various categories: 'bishop', 'city', 'club', 'football player', 'game', 'line', 'market', 'meter', 'operator', 'person', 'player', 'railway station', 'road', 'sextet', 'stadium', 'subdivision', 'tower', 'undertaking', and 'university venue'. The search results show '73 results found (0.00625 sec sparq query time), showing 20 results per page'. The first result is 'Busan W', which is expanded to show details. The details view includes a title 'Busan W', a description of Busan Metropolitan City, a map of the city, and a list of related items such as 'Busan Subway W' and 'Busan Lotte Tower W'. The 'Busan Subway W' section includes a map and text describing the subway system. The 'Busan Lotte Tower W' section includes text about the tower's location and construction. The 'Busan W' section includes a list of 'Linked Datasets', 'Location of', 'Location of', 'City of', 'Body of', 'Citybirth of', 'Birthplace of', and 'Reference'.

Fig. 5. Search results and details view for Busan

### 6.3 Querying DBpedia Data

Compared to most of the other Semantic Web knowledge bases currently available, for the RDF extracted from Wikipedia we have to deal with a different type of knowledge structure – we have a very large information schema and a considerable amount of data adhering to this schema. Existing tools unfortunately mostly focus on either one of both parts of a knowledge base being large, schema *or* data.

If we have a large data set and large data schema, elaborated RDF stores with integrated query engines alone are not very helpful. Due to the large data schema, users can hardly know which properties and identifiers are used in the knowledge base and hence can be used for querying. Consequently, users have to be guided when building queries and reasonable alternatives should be suggested.

We specifically developed a graph pattern builder for querying the extracted Wikipedia content. Users query the knowledge base by means of a graph pattern consisting of multiple triple patterns. For each triple pattern three form fields capture variables, identifiers or filters for subject, predicate and object of a triple. While users type identifier names into one of the form fields, a look-ahead search proposes suitable options. These are obtained not just by looking for matching identifiers but by executing the currently built query using a variable for the currently edited identifier and filtering the results returned for this variable for matches starting with the search string the user supplied. This method ensures, that the identifier proposed is really used in conjunction with the graph pattern under construction and that the query actually returns results. In addition, the identifier search results are ordered by usage number, showing commonly used identifiers first. All this is executed in the background, using the Web 2.0 AJAX technology and hence completely transparent for the user. Figure 6 shows a screenshot of the graph pattern builder.

**UNIVERSITÄT LEIPZIG pedia**

**Query Wikipedia**

This semantic database contains over 10 million statements extracted from the English Wikipedia.

[search for queries](#) | [Most popular](#) | [Upcoming](#)

[Tennis players from Moscow](#)

[Sitcoms set in NYC](#)

[Soccer player with tricot nr. 11, playing for a club having a stadium with >40.000 seats, born in a country with >10M inhabitants](#)

[People influenced by Friedrich Nietzsche](#)

[Films longer than 5 hours](#)

[Space Missions](#)

[Film music composer born 1966](#)

[People being 1.80m tall](#)

[List of Web browser software](#)

[Mayors of US cities higher than 1000m](#)

[Pictures of American guitarists](#)

[Battles in Saxony](#)

[What connects Innsbruck and Leipzig](#)

[Hip hop CDs from Texas Artists](#)

[Scientists and their doctoral advisors](#)

<< 1 >>

**Soccer player with tricot nr. 11, playing for a club having a stadium with >40.000 seats, born in a country with >10M inhabitants**

Subject	Predicate	Object
?player	currentclub	?club
?player	clubnumber	11
?player	countryofbirth	?country
?club	capacity	>40000
?country		>10000000

Click on a column header to filter results. Results: 10

Nr.	?player	?country	>40000 >10000000
1	<a href="#">Cicinho</a>	<a href="#">Brazil</a>	80354 187560000
2	<a href="#">Gonzalo Fierro</a>	<a href="#">Colo-Colo</a>	62000 16432674
3	<a href="#">Lukas Podolski</a>	<a href="#">FC Bayern Munich</a>	69901 38536869
4	<a href="#">Mark González</a>	<a href="#">Liverpool F.C.</a>	45362 47432000
5	<a href="#">Michael Thürk</a>	<a href="#">Eintracht Frankfurt</a>	52000 82438000
6	<a href="#">Ramón Morales</a>	<a href="#">Chivas de Guadalajara</a>	72480 107784179
7	<a href="#">Robin van Persie</a>	<a href="#">Arsenal F.C.</a>	60432 16336346
8	<a href="#">Stefano Mauri</a>	<a href="#">S.S. Lazio</a>	82656 58751711

Fig. 6. Form based query builder

### 6.4 Third Party User Interfaces

The DBpedia project aims at providing a hotbed for applications and mashups based on information from Wikipedia. Although DBpedia was just recently launched, there is already a number of third party applications using the dataset. Examples include:

- A SemanticMediaWiki [14][18] installation run by the University of Karlsruhe, which has imported the DBpedia dataset together with the English edition of Wikipedia.
- WikiStory (see Figure 7) which enables users to browse Wikipedia articles about people on a large timeline.
- The Objectsheet JavaScript visual data environment, which allows spreadsheet calculations based on DBpedia data [13].

## 7 Related Work

A second project that also works on extracting structured information from Wikipedia is the YAGO project [16]. YAGO extracts only 14 relationship types, such as *subclassOf*, *type*, *familyNameOf*, *locatedIn* from different sources of information in Wikipedia. One source is the Wikipedia category system (for *subclassOf*, *locatedIn*, *diedInYear*, *bornInYear*), and another one are Wikipedia redirects. YAGO does not perform an infobox extraction as in our approach. For determining (sub-)class relationships, YAGO does not use the full Wikipedia category hierarchy, but links leaf categories to the WordNet hierarchy.

<sup>13</sup> [http://richk.net/objectsheet/osc.html?file=sparql\\_query1.05](http://richk.net/objectsheet/osc.html?file=sparql_query1.05)

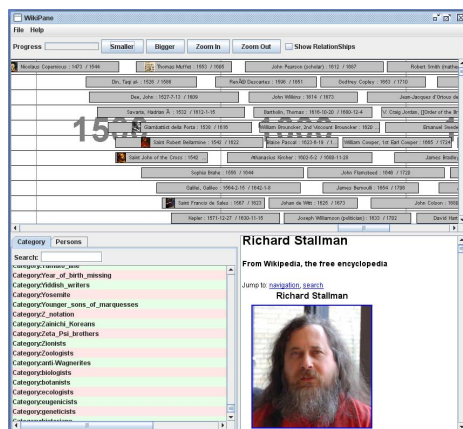


Fig. 7. WikiStory allows timeline browsing of biographies in Wikipedia

The Semantic MediaWiki project [14,18] also aims at enabling the reuse of information within Wikis as well as at enhancing search and browse facilities. Semantic MediaWiki is an extension of the MediaWiki software, which allows you to add structured data into Wikis using a specific syntax. Ultimately, the DBpedia and Semantic MediaWiki have similar goals. Both want to deliver the benefits of structured information in Wikipedia to the users, but use different approaches to achieve this aim. Semantic MediaWiki requires authors to deal with a new syntax and covering all structured information within Wikipedia would require to convert all information into this syntax. DBpedia exploits the structure that already exists within Wikipedia and hence does not require deep technical or methodological changes. However, DBpedia is not as tightly integrated into Wikipedia as is planned for Semantic MediaWiki and thus is limited in constraining Wikipedia authors towards syntactical and structural consistency and homogeneity.

Another interesting approach is followed by Freebase [14]. The project aims at building a huge online database which users can edit in a similar fashion as they edit Wikipedia articles today. The DBpedia community cooperates with Metaweb and we will interlink data from both sources once Freebase is public.

## 8 Future Work and Conclusions

As future work, we will first concentrate on improving the quality of the DBpedia dataset. We will further automate the data extraction process in order to increase the currency of the DBpedia dataset and synchronize it with changes in Wikipedia. In parallel, we will keep on exploring different types of user interfaces and use cases for the DBpedia datasets. Within the W3C Linking Open

<sup>14</sup> <http://www.freebase.com>

Data community project <sup>15</sup> we will interlink the DBpedia dataset with further datasets as they get published as Linked Data on the Web. We also plan to exploit synergies between Wikipedia versions in different languages in order to further increase DBpedia coverage and provide quality assurance tools to the Wikipedia community. Such a tool could for instance notify a Wikipedia author about contradictions between the content of infoboxes contained in the different language versions of an article. Interlinking DBpedia with other knowledge bases such as Cyc (and their use as back-ground knowledge) could lead to further methods for (semi-) automatic consistency checks for Wikipedia content.

DBpedia is a major source of open, royalty-free data on the Web. We hope that by interlinking DBpedia with further data sources, it could serve as a nucleus for the emerging Web of Data.

## Acknowledgments

We are grateful to the members of the growing DBpedia community, who are actively contributing to the project. In particular we would like to thank Jörg Schüppel and the OpenLink team around Kingsley Idehen and Orri Erling.

## References

1. Aberer, K., Cudré-Mauroux, P., Hauswirth, M.: The chatty web: Emergent semantics through gossiping. In: 12th World Wide Web Conference (2003)
2. Auer, S., Lehmann, J.: What have innsbruck and leipzig in common? extracting semantics from wiki content. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 503–517. Springer, Heidelberg (2007)
3. Benjelloun, O., Sarma, A., Halevy, A., Widom, J.: Uldbs: Databases with uncertainty and lineage. In: VLDB 2006 (2006)
4. Tim Berners-Lee. Linked data (2006), <http://www.w3.org/DesignIssues/LinkedData.html>
5. Bizer, C., Cyganiak, R., Heath, T.: How to publish linked data on the web (2007), <http://sites.wiwi.fu-berlin.de/suhl/bizer/pub/LinkedDataTutorial/>
6. Buneman, P., Khanna, S., Tan, W.C.: Why and where: A characterization of data provenance. In: Van den Bussche, J., Vianu, V. (eds.) ICDT 2001. LNCS, vol. 1973, Springer, Heidelberg (2000)
7. Bizer, C.: Quality-Driven Information Filtering in the Context of Web-Based Information Systems. PhD thesis, Freie Universität Berlin (2007)
8. Cui, Y.: Lineage Tracing in Data Warehouses. PhD thesis, Stanford University (2001)
9. Erling, O., Mikhailov, I.: RDF support in the Virtuoso DBMS. volume P-113 of GI-Edition - Lecture Notes in Informatics (LNI), Bonner Köllen Verlag (September 2007) (ISSN 1617-5468)
10. Halevy, A., Etzioni, O., Doan, A., Ives, Z., Madhavan, J., McDowell, L.: Crossing the structure chasm. In: CIDR 2003 (2003)

<sup>15</sup> <http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>

11. Halevy, A.Y., Ives, Z.G., Suciu, D., Tatarinov, I.: Schema mediation in peer data management systems. In: ICDE 2003 (2003)
12. Ives, Z., Khandelwal, N., Kapur, A., Cakir, M.: Orchestra: Rapid, collaborative sharing of dynamic data. In: CIDR 2005 (January 2005)
13. Kementsietsidis, A., Arenas, M., Miller, R.J.: Mapping data in peer-to-peer systems: Semantics and algorithmic issues. In: SIGMOD 2003 (June 2003)
14. Krötzsch, M., Vrandečić, D., Völkel, M.: Wikipedia and the Semantic Web - The Missing Links. In: Voss, J., Lih, A. (eds.) Proceedings of Wikimania 2005, Frankfurt, Germany (2005)
15. Smith, J.M., Bernstein, P.A., Dayal, U., Goodman, N., Landers, T., Lin, K.W.T., Wong, E.: MULTIBASE – integrating heterogeneous distributed database systems. In: Proceedings of 1981 National Computer Conference (1981)
16. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A Core of Semantic Knowledge. In: WWW 2007. 16th international World Wide Web conference, ACM Press, New York
17. Berners-Lee, T., et al.: Tabulator: Exploring and analyzing linked data on the semantic web. In: Proceedings of the 3rd International Semantic Web User Interaction Workshop (2006),  
<http://swui.semanticweb.org/swui06/papers/Berners-Lee/Berners-Lee.pdf>
18. Völkel, M., Krötzsch, M., Vrandečić, D., Haller, H., Studer, R.: Semantic wikipedia. In: Carr, L., De Roure, D., Iyengar, A., Goble, C.A., Dahlin, M. (eds.) WWW 2006. Proceedings of the 15th international conference on World Wide Web, pp. 585–594. ACM Press, New York (2006)
19. Wiederhold, G.: Intelligent integration of information. In: SIGMOD (1993)

# A Semantic Case-Based Reasoning Framework for Text Categorization

Valentina Ceausu<sup>1</sup> and Sylvie Desprès<sup>2</sup>

<sup>1</sup> CRIP 5 - University of Paris 5,  
45, Rue des Saints Pères  
Paris 75006, France

`ceausu@math-info.univ-paris5.fr`

<sup>2</sup> LIPN UMR CNRS 7030 - University of Paris 13,  
99 avenue Jean Baptiste Clément  
93430 Villetaneuse, France

`sylvie.despres@lipn.univ-paris13.fr`

**Abstract.** This paper presents a semantic case-based reasoning framework for text categorization. Text categorization is the task of classifying text documents under predefined categories.

Accidentology is our application field and the goal of our framework is to classify documents describing real road accidents under predefined road accident prototypes, which also are described by text documents. Accidents are described by accident reports while accident prototypes are described by accident scenarios. Thus, text categorization is done by assigning each accident report to an accident scenario, which highlights particular mechanisms leading to accident.

We propose a textual case-based reasoning approach (TCBR), which allows us to integrate both textual and domain knowledge aspects in order to carry out this categorization. CBR solves a new problem (target case) by identifying its similarity to one or several previously solved problems (source cases) stored in a case base and by adapting their known solutions. Cases of our framework are created from text. Most of TCBR applications create cases from text by using Information Retrieval techniques, which leads to knowledge-poor descriptions of cases. We show that using semantic resources (two ontologies of accidentology) makes possible to overcome this difficulty, and allows us to enrich cases by using formal knowledge.

In this paper, we argue that semantic resources are likely to improve the quality of cases created from text, and, therefore, such resources can support the reasoning cycle. We illustrate this claim with our framework developed to classify documents in the accidentology domain.

**Keywords:** semantic description, ontology, text categorization, case-based reasoning, accidentology.

## 1 Introduction

Case-based reasoning (CBR), [1] is a problem solving paradigm which solves a new problem by re-using a collection of already solved problem (called source



cases). This collection represents the case base. Textual CBR, see [2] is an extension of CBR which could be applied in domains where experiences are described by text documents. As many domains produce a large amount of textual data describing problems and their solutions, developing CBR systems able to deal with unstructured or semi-structured text is particularly challenging.

Text documents are unstructured stream of characters, over which only shallow reasoning based on easily observable surface features can be performed.

Thus, cases of TCBR systems are often created by hand or have simplified representations, which can be created by using results of Information Retrieval methods, see [3], [4] or [5].

[6] points out the role of such methods in creating textual cases. Those methods are based on shallow statistical inferences over word vectors, and allow creating a linguistic description of cases, as cases are represented by terms extracted from text. By using Information Retrieval methods, knowledge-poor representations of textual cases are obtained.

This leads to a bottleneck in creating and scaling up TCBR systems, since manual construction of cases often involves inhibitory costs and simplified representations of cases lead to an inefficient reasoning cycle, as little knowledge could be exploited by the cycle.

However, there is a severe gap between the knowledge required for TCBR and the results provided by methods one can perform on textual documents.

Thus, methods, like in particular Information Retrieval, are not sufficient to create knowledge-rich case representations from text. As, among others, [7] points out, the weakness of simple Information Retrieval methods is its lack of exploitation of knowledge about domain objects and relationships.

Since TCBR application is domain specific, descriptions of cases can be improved by using domain ontology. [8] defines an ontology as a formal, explicit specification of a shared conceptualization. Ontology is a formal representation of domain knowledge, providing information about specific objects of the domain and relationships between them. Domain is modeled at conceptual level, in an implementation independent manner. Objects are modeled by concepts, having particular attributes. Relationships between them are modeled as roles, which are binary relations holding between concepts. Each role has a domain and range, both of which are concepts of ontology.

For this work, we assume that an ontology takes into account the linguistic level of entities. Thus, concepts and roles are labeled by terms, which are linguistic manifestation of ontology entities in a specific language (French, English, etc.). Therefore, ontology considered for this work has two levels: a conceptual level, describing domain specific entities (concepts and roles) and a linguistic level, providing linguistic manifestations of those entities in a given language.

Therefore, we claim that such ontology could help creating cases from text. We illustrate this claim by presenting ACCTOS (ACCident TO Scenarios), a TCBR framework integrating ontologies to create cases from text. Cases of ACCTOS are described at formal level, by concepts and roles of semantic resources. By integrating those resources, formal knowledge could be exploited by the reasoning cycle.

## 2 Assigning Accident Reports to Accident Scenarios: A Text Categorization Task

This paper deals with automatic assignment of documents describing real road accident to documents describing road accident prototypes. Road accidents are described by accident reports while accident prototypes are described by accident scenarios.

Accident reports are documents created by the police. They include structured paragraphs describing the context of an accident and people involved in, and natural language paragraphs explaining what happened in the accident. Those paragraphs are written by policemen, with the help of witnesses and people involved in the accident.

Accident scenarios are documents created by researchers in road safety. They are prototypes of road accidents and present in a general way facts and causal relations between different phases leading to a collision. Prevention measures aiming to improve road safety are provided for each accident scenario. A first study led by the department Mechanisms of Accidents of INRETS<sup>1</sup> established a first collection of accident scenarios involving pedestrians.

As the tab. 1 shows, there is a number of differences between accident reports and accident scenarios. Thus, accident reports are created by the police, while

**Table 1.** Accident reports vs. Accident scenarios

	Accident reports	Accident scenarios
created by	policemen	road safety researchers
language	current language	expert language
contains	description of accidents	expert knowledge
structure	semi structured	free text
goal	identify legal responsibility	prevention of road accidents

accident scenarios are created by researchers in road safety. Therefore, accident reports provide descriptions of road accidents written in current language. This means that a notion is often designated by many synonym terms (i.e. person driving a car: *conducteur*, *chauffeur*, *automobiliste (driver)*). Accident scenarios are written in expert language, and the same term is always used to designate a notion (i.e. person driving a car: *conducteur (driver)*).

Assigning an accident report to an accident scenario is twofold: from a domain specific point of view, it allows us to identify particular mechanisms leading to accident; from a linguistic point of view, it allows us to create a bridge between the languages of two different communities (researchers and policemen) of the same domain.

We consider accident scenarios as predefined text categories, as they describe prototypes of road accidents. Therefore, assigning an accident report to an accident scenario is a text categorization task. Moreover, preventions measures are

<sup>1</sup> Institut National de Recherche sur les Transports et leur Sécurité.

provided for each accident scenario. Thus, an accident scenario and his prevention measures can be seen as a problem description (a particular prototype of accident) and his solution (measures proposed in order to avoid this particular prototype of road accident). By consequent, we developed a textual case-based reasoning framework in order to carry out this categorization task. On the following we present this framework.

### 3 ACCTOS: A TCBR Framework for Text Categorization

ACCTOS is a textual case-based reasoning frame developed to classify textual documents.

#### 3.1 ACCTOS Input/Output Data

The input of the system is a set of accident reports. ACCTOS exploits electronic accident reports, which have been made anonymous by the PACTOL <sup>2</sup> tool. An electronic accident report is a semi-structured document containing structured paragraphs and natural language paragraphs. Structured paragraphs specify a number of variables describing: people and vehicles involved in accident, accident context and accident environment. Natural language paragraphs describe what happened in the accident according to several points of view: police (synthesis), people involved (declarations) and witnesses (testimonies).

*The output of the system* is set of assignments, where each assignment is composed of a couple *accident report*, *accident scenario* and a trust assessment.

#### 3.2 Architecture of ACCTOS

ACCTOS adopts a CBR approach. CBR solves a new problem (target case) by exploiting a collection of already solved problems (source cases). The CBR reasoning cycle consists of following phases:

- target case elaboration: creates the target case (problem to solve);
- case retrieval: identifies a number of source cases similar to the target case;
- case adaptation: adapts solutions of source cases (identified by the previous phase) in order to propose a solution for the target case;
- memorization phase: enrich the case base, by adding the target case and his solution.

ACCTOS implements two phases of the CBR reasoning cycle: target case elaboration and case retrieval. To present the architecture of ACCTOS, we use a division into modules, where each of the module addresses a different phase of the reasoning cycle (see Fig. [11](#)).

***The need for formal knowledge to create cases from text.*** Target cases of ACCTOS are created from accident reports. Source cases are created from accident scenarios.

<sup>2</sup> Centre d'Etudes Techniques de l'Equipement de Rouen.

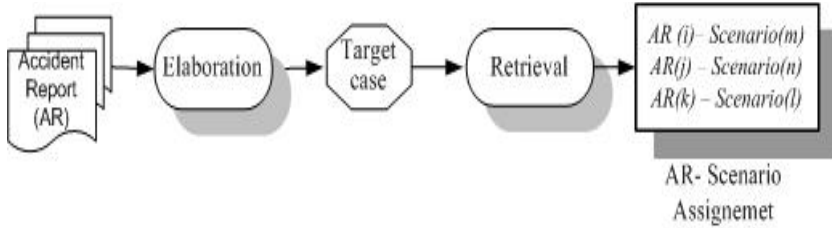


Fig. 1. System architecture

Creating cases from text is a difficult task, as text is unstructured data. To cope with this difficulty, we use semantic resources allowing us to create knowledge-rich representations of cases. As the framework exploits documents created by two communities, we integrate two semantic resources, modeling the accidentology domain according to each community. Thus, the *expert ontology* describes the domain from an expert point of view, while the *facts ontology* describes the domain from a police point of view. Both ontologies are expressed in OWL, [9].

By modeling each community by a semantic resource, it becomes possible to reflect the dynamic of the community. For instance, the expert ontology can be enriched when new scenarios are created by experts in road safety.

### 3.3 Representation of Cases

We proposed a model to represent cases of ACCTOS. According to this model, a case is described by two types of elements: global variables and agents. Global variables specify the number of agents involved in accident, the environment in which the accident occurred - such as main road or secondary road - and context of the accident (by day, in intersection, etc. ).

A human involved in accident and his vehicle represent an agent (see tab 2).

By using agents, it becomes possible to cope with difficulties related to metonymy between the human involved in accident and his vehicle (i.e. *vehicle stops* vs. *driver stops*). It also allows us to treat the particular case of pedestrian. Each agent is defined by his two components - the human H and the vehicle V - and by his evolution in accident. Each component of an agent is designated by a domain term (ie: driver, car) and has several attributes (ie: age is an attribute of Human). Agent evolution is specified by a set of relations describing interac-

Table 2. Components of an agent

Agent	Humain	Vehicle	Attributes	Evolution
Agent 1	piéton (pedestrian)	no vehicle	age: 35	traverser; courir (crossing; running)
Agent 2	(conducteur) (driver)	Véhicule (car)	age: 60	circuler; tourner (circulate; turning to)

tions between his own components and also between the agent and other agents involved in accident.

### 3.4 Creating Source Cases by Using the Expert Ontology

Cases of case base are called source cases and have a two parts: the problem and his solution. A set of accident scenarios is used to build the initial case base of the system. The accident scenario represents the *Problem*; measures of preventions assigned to the scenario represent the *Solution*. For this work, the solution part of source cases is ignored, as the adaptation phase of the reasoning cycle is not implemented.

The *expert ontology* (see [10]) supports the description of source cases. This ontology was built from scratch, by using a corpus composed of accident scenarios and expert knowledge. It models concepts of accidentology and relations holding between them. Concepts are structured in three main classes: concepts describing the *human*, the *vehicle* and the *environment*. Each concept is named by a term and has different attributes.

Concepts are connected by *is-s* relations and roles. *is-a* relations build the hierarchy of domain concepts. Roles describe interactions between concepts and are named by domain specific verbs (i.e. *CirculerSur(Véhicule, Infrastructure)* (CirculateOn(Vehicle,Infrastructure))).

We developed an editor in order to create a source case from an accident scenario. The editor integrates the expert ontology and allows us to describe each accident scenario by a set of concepts and roles of the ontology. The editor also allows us to assign an importance coefficient to each concept or role. For each source case, those coefficients are established by experts. By integrating this ontology, we create source cases having homogeneous descriptions and we can describe them by using formal knowledge (concepts and roles).

### 3.5 Target Case Elaboration by Using Semantic Traces

The goal of this phase is to create a target case from an accident report. We create cases from text by using *semantic traces*. On the following we introduce semantic traces and we present the approach proposed to identify semantic traces from text.

**Semantic Traces: Definition.** Let  $C$  be a corpora and  $O$  an ontology of the same domain. As entities of  $O$  have linguistic descriptions, it becomes possible to identify within  $C$  terms similar to those naming entities of  $O$ . If  $e$  is an entity of  $O$ , we define a semantic trace of  $e$  as a term  $t$  of the  $C$  corpora which is similar to the label of  $e$ . In other words, semantic traces are terms of  $C$  which are similar (from a lexical point of view) to those naming the entities of  $O$ .

Discovering semantic traces is based on the following working hypothesis: if synonymy is not considered, then any entity is named by using the same set of characters, called the core set. Consequently, terms which are named by a set of characters close to the core set represent either the same or a similar entity.

Terms expressed by a set of characters completely different to the core set are referring potentially different entities.

Once identified within the corpus, these terms can be labelled by entities of the ontology. A semantic trace of a concept  $c$  (role  $r$ ) is discovered each time a term is labelled by the concept  $c$  (the role  $r$ ).

**Discovering Semantic Traces.** We have proposed a two steps approach to discover semantic traces within corpora. The first step extracts terms by using an Information Retrieval method. The second one uses string similarity metrics to label extracted terms by ontology entities.

**Terms' extraction using Information Retrieval.** Terms are extracted from corpus by using an Information Retrieval method, based on lexical patterns, see [11]. We define a lexical pattern as a particular combination of part-of-speech categories. For instance *Noun, Preposition, Noun* or *Verb, Preposition, Noun* are lexical patterns. In order to identify instances of patterns, the corpora is tagged using TreeTagger, see [12], which makes part-of-speech information available. This method consists in defining a set of lexical patterns able to extract potentially valid terms from corpora. Then, a pattern recognition algorithm which we implemented retrieves word regroupings matching lexical patterns<sup>3</sup>, see tab. 3. We defined two categories of lexical patterns in order to discover

**Table 3.** Lexical patterns and instances

Pattern	Instance	Note
Noun, Preposition, Noun	ceinture de sécurité (seat belt)	domain term
Noun, Noun	passage piéton (cross road)	domain term
Verb, Preposition	diriger vers (direct to)	verb relation
Verb, Preposition, Noun	venir de i (come from i)	noise

semantic traces: *nominal lexical patterns* are associations of part-of-speech categories which do not include a verb and *verbal lexical patterns* are associations of part-of-speech categories including a verb.

Nominal lexical patterns highlight domain terms, so instances of those patterns could be semantic traces of concepts. Verbal lexical patterns highlight domain relationship expressed by verbs, so instances of those patterns could be semantic traces of roles. The pattern recognition algorithm is applied at sentence level and automatically generates two sets of lexical pattern instances.

**Using an ontology to pass from a linguistic description to a formal description of cases.** The method described in this section allows identifying terms of corpus by using a basic Information Retrieval method. While creating cases from text, this method could provide a description of cases, as significant terms can be extracted. However, this is a linguistic description, as only terms

<sup>3</sup> Examples of this paper are translated in English, although they are extracted from a French corpus experimentation.

are identified. On the other hand, terms can be considered as linguistic manifestations of concepts or roles, see [13]. Hence, if a domain ontology is available, it becomes possible to label those terms by entities of the ontology. This allows us to enrich description of textual cases, by using concepts and roles by which terms were labelled. In the next section we describe the labelling of terms by entities of ontology.

**Semantic labelling of terms.** As the previous section shows, the pattern recognition algorithm identifies instances of nominal and verbal lexical patterns. Instances of nominal patterns could be labelled by concepts, as they could highlight domain terms. Instances of verbal patterns highlight relations of the domain, therefore they can be labelled by roles.

To label a nominal instance, the set of concepts is considered. A string similarity coefficient, see [14] is used to calculate the similarity between instances and terms naming ontology concepts. Each instance will be labelled by the concept whose label (term naming the concept) maximizes the value of this similarity, if the maximum similarity is above a threshold value. Otherwise, the instance is labelled as *inconnu*, (unknown). Each nominal instance labelled by a concept represents the semantic trace of this concept.

Instances of verbal patterns are labelled in a similar way, by considering the set of roles modelled by the ontology. Each verbal instance labelled by a role represents the semantic trace of this role.

### 3.6 Using Semantic Traces to Elaborate Target Cases of ACCTOS

The goal of this phase is to create the target case. The target case is created from text by discovering semantic traces entities modelled by the *facts ontology*, see [15]. This ontology was created from a corpora of about 250 accident reports, by using the *Terminae* tool, see [16]. It models concepts of accidentology and relations holding between them according to a police specific point of view. This ontology points out linguistic particularities of this community, thanks to his conceptual and linguistic level.

Each target case is represented according to the model presented in the section 3.3 and is created from an accident report. An accident report is a semi-structured document, composed of specific structures and natural language paragraphs. Specific structures provide data about: people and vehicles involved in accident, accident context and accident environment. Natural language paragraphs provide descriptions of the accident, according to several points of view (people involved, witnesses). Target cases are created by exploiting both specific structures and natural language paragraphs of an accident report, as shown in the following.

**Identification of global variables.** Values of global variables are identified by automatic procedures exploiting the structure of accident reports.

**Identification of agents.** To describe an agent involved in accident we need to:

- identify terms naming his components;
- identify values of his attributes;
- identify his evolution.

Terms naming component of an agent and values of his attributes are also identified by automatic procedures exploiting the structure of accident reports.

In order to enrich the description of target cases, terms naming components are labelled by concepts of facts ontology, by using the two steps approach presented in section 3.5. This labelling is always possible, as the ontology was created from accident reports. Moreover, it allows us to pass from a linguistic description of components, to a formal one, as we can see in tab 4. Further, this formal description allows us to identify evolutions of agents, as it follows.

**Table 4.** Semantic labelling of instances

Type of instance	Instance	Ontology entity
Nominal	jeune piéton (young pedestrian)	<i>piéton SubConceptOf(Personne)</i> <i>(pedestrian SubConceptOf(Person))</i>
Verbal	circuler sur (circulate on)	<i>circuler(Véhicule, Infrastructure)</i> <i>(circulate(Vehicle, Infrastructure))</i>

**Identification of agents’ evolution.** Evolutions of agents are expressed by a set of verbs appearing in natural language paragraphs (synthesis, declarations and testimonies) of accident reports. The evolution of an agent is identified by discovering traces of facts ontology roles within these paragraphs.

Semantic traces of roles are discovered by using the approach presented in section 3.5. Hence, two lexical patterns are defined: *Verb* and *Verb, Preposition*. Instances of those patterns are identified within paragraphs (previously annotated by TreeTagger). Then, those instances are labelled by roles of facts ontology.

Semantic traces identified consists in a set  $R$  of verbs which are similar (from a lexical point of view) to verbs naming roles of facts ontology, see (II).

$$Traces_{evolution} = \{t_1, t_2, \dots, t_n | t_i \text{ is a semantic trace}\} \tag{1}$$

In order to identify evolution of agents, each semantic trace is replaced by the corresponding role. By doing so, we end up with a set of roles describing evolutions of all agents involved in accident, see (2).

$$Roles_{evolution} = \{r_1, r_2, \dots, r_n | r_i \in Roles_{RTO}\} \tag{2}$$

where  $Roles_{RTO}$  is the set of facts ontology roles.

Let  $a_i$  be an agent whose components are described by concepts  $H$  and  $V$ . This agent should identify, among roles of  $Roles_{evolution}$ , those describing his own evolution. To do so, agent  $a_i$  query the facts ontology in order to get roles



of this ontology having  $H$  or  $V$  as domain. As consequence, a set  $Roles_{(H,V)}$  is obtained, see (3).

$$Roles_{(H,V)} = \{r_1, r_2, \dots, r_n | r_i \in Roles_{RTO} \text{ having H or V as domain}\} \quad (3)$$

The evolution of agent  $a_i$  is given by the intersection of the two sets:  $Roles_{(H,V)}$  and  $Roles_{evolution}$ , see (4)

$$Evolution_{(H,V)} = Roles_{(H,V)} \cap Roles_{evolution} \quad (4)$$

The evolution of each agent is expressed by a set of facts ontology roles, whose traces were identified within natural language paragraphs, see fig. 2.

### 4 Semantic Retrieval

The retrieval phase aims to retrieve source cases similar to the target case. Already solved problems similar to the target case are identified. Therefore, a solution can be proposed to the target case by adapting solutions of those problems. As both target cases and source cases of ACCTOS have semantic descriptions, we propose a retrieval approach supported by the alignment of the experts and facts ontology.

We have proposed an aligned approach, described in [17]. The alignment is given by a similarity function  $Sim(e_e, e_f)$  which allows us to estimate similarity between entities (concepts or roles) of experts ontology ( $e_e$ ) and entities of facts ontology ( $e_f$ ).

Let  $T$  be a target case. Two steps are needed to retrieve similar source cases. (1) *The first step is based on case base indexation.* Global variables are used to index the case base. Values of global variables of the target case are taken into account to identify a set of source cases. The result is a set of source cases having the same context as the target case and involving the same number of agents. (2) *A voting process* is used to improve this first selection. The vote is done by each target case agent to express the resemblance degree between himself and agents of a source case. A note is given by each target case agent to every source case. This note is given by taking into account components of agents and their evolutions. A first similarity measure proposed is given by:

$$Sim(a_i, a_j) = SimComponent(a_i, a_j) + SimEvolution(a_i, a_j) \quad (5)$$

if  $SimComponent(a_i, a_j) \neq 0$ , otherwise  $Sim(a_i, a_j) = 0$ , where  $a_i$  is an agent of the target case and  $a_j$  is an agent of a source case, and  $SimComponent(a_i, a_j)$  expresses resemblances between components of two agents, and is given by :

$$SimComponent(a_i, a_j) = ch_j * sim(H_i, H_j) + cv_j * sim(V_i, V_j) \quad (6)$$

where  $ch_j$  and  $cv_j$  are importance coefficients established for the source case, and values of  $sim(H_i, H_j)$  and  $sim(V_i, V_j)$  are given by the alignment of the two resources.

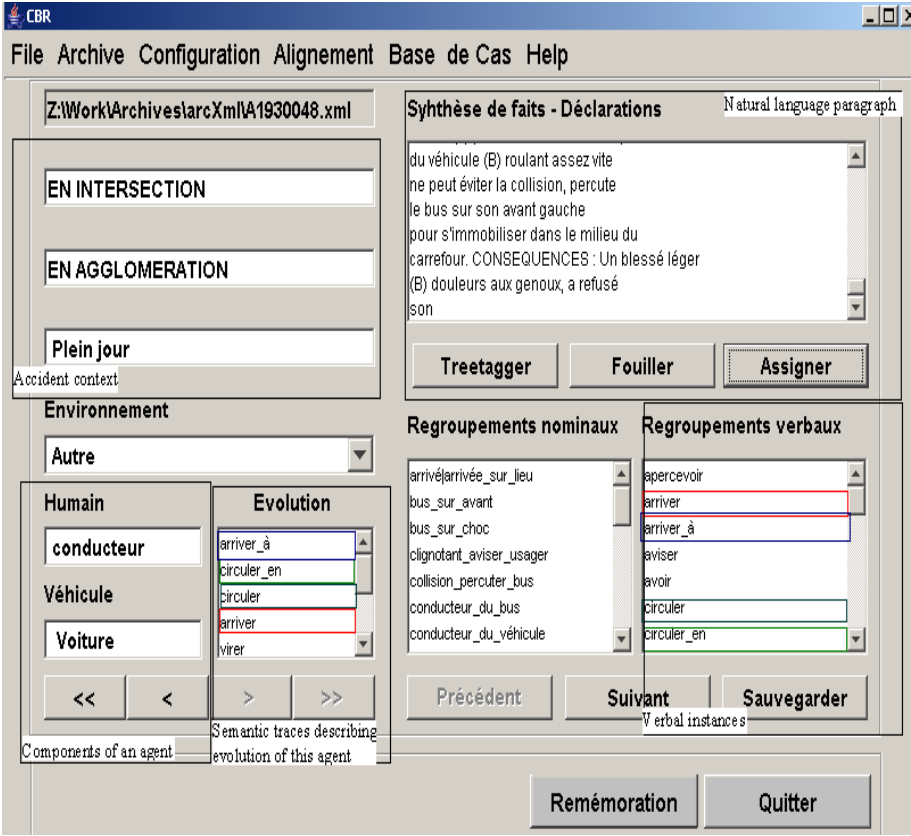


Fig. 2. Identification of evolution

Evolution similarity expresses resemblances between evolutions of two agents :

$$SimEvolution(a_i, a_j) = \frac{\sum_r c_r * sim(rSource_r, rTarget_r)}{\sum_r c_r} \tag{7}$$

where coefficients  $c_r$  expresses the importance of  $rSource_r$  role for the considered source case. Values of  $sim(rSource_r, rTarget_r)$  are given by alignment of the two resources.

Each agent of the target case evaluates his resemblance to agents of the source case by using the presented approach. A similarity vector is obtained. The note  $note_i$  given by the  $agent_i$  to the source case is the maximum value of this similarity vector. Based on notes given by agents, the similarity between the target case and a source case is estimated by the average value:

$$Sim(target, source) = \frac{\sum_{i=1}^{N_a} note_i}{N_a} \tag{8}$$

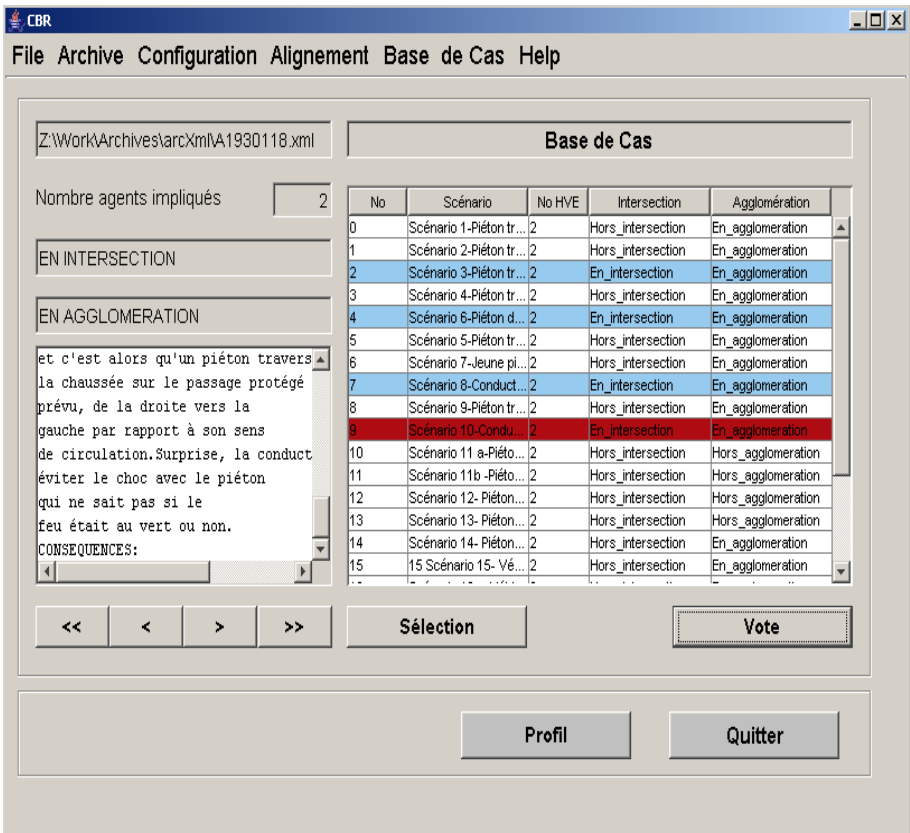


Fig. 3. Case retrieval

where  $note_i$  is the note granted by the agent  $agent_i$ , and  $N_a$  is the number of agents of the considered target case. Case base indexation allows a fast identification of source cases that are similar to the target case. By voting, the most similar cases are selected among the cases retrieved by the first selection. The retrieval process is driven by the description of source cases whose importance coefficients are taken into account by similarity measures. Fig. 3 shows cases selected by case base indexation (light gray) and by vote (dark gray).

## 5 Conclusion and Future Work

This paper presents a semantic case-based reasoning framework for text categorization. Cases of the framework are created from natural language documents provided by two different communities: accident reports written by the police and accident scenarios created by road safety researchers.

Semantic resources are used to cope with heterogeneity and difficulties related to case elaboration from natural language documents. Two ontologies are used to create source and target cases of the system. By integrating semantic resources, we can create knowledge-rich descriptions of cases, as cases are described by concepts and roles of two different ontologies. The advantage is that this knowledge can be used by the reasoning cycle. Hence, the retrieval phase is supported by aligning the expert and the facts ontology.

The development of the framework is finished. There now remains to evaluate his results and to identify different ways to improve them. As for now, an expert evaluation of the system is ongoing. This evaluation is carried out in collaboration with road safety experts, able to validate *accident report*, *accident scenarios* assignments provided by ACCTOS. This validation will allow us to evaluate the precision of ACCTOS results.

## References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations and system approaches. *AICom - Artificial Intelligence Communications* 7(1), 39–59 (1994)
2. Lamontagne, L., Lapalme, G.: Raisonnement à base de cas textuel: état de l'art et perspectives futures. *Revue d'intelligence artificielle* 16(3), 339–366 (2002)
3. Wiratunga, N., Koychev, I., Massie, S.: Feature selection and generalisation for retrieval of textual cases. In: *Proceedings of the 7-th European Conference on Case-Based Reasoning* (2004)
4. Gupta, K., Aha, D., Sandhu, N.: Exploiting taxonomic and causal relations in conversational case retrieval. In: *Proceedings of the Sixth European Conference on Case-Based Reasoning* (2002)
5. Bergmann, R.: On the use of taxonomies for representing case features and local similarity measures. In: *Proceedings of the 6th German Workshop on Case-Based Reasoning* (1998)
6. Bruninghaus, S., Ashley, K.D.: The role of information extraction for textual cbr. In: Aha, D.W., Watson, I. (eds.) *ICCBR 2001. LNCS (LNAI)*, vol. 2080, pp. 74–89. Springer, Heidelberg (2001)
7. Lenz, M.: Textual cbr and information retrieval - a comparison. In: *Proceedings of the 6th German Workshop on Case Based Reasoning* (1998)
8. Gruber, T.: A translation approach to portable ontology specifications. *Knowledge Acquisition*, 199–220 (1993)
9. Smith, M., Welty, C., McGuinness, D.: Owl web ontology language guide. Technical report, W3C, W3C Proposed Recommendation (2004)
10. Desprès, S.: Contribution à la conception de méthodes et d'outils pour la gestion des connaissances. In: *Habilitation à diriger des recherches*, Université René Descartes (2002)
11. Seguela, P.: Adaptation semi-automatique d'une base de marqueurs de relations sémantiques sur des corpus spécialisés. In: *Terminologie et Intelligence Artificielle* (1999)
12. Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In: *Proceedings of the International Conference on New Methods in Language Processing* (1994)

13. Ville-Ometz, F., Royauté, J., Zasadzinski, A.: Filtrage semi-automatique des variantes de termes dans un processus d'indexation contrôlée. In: Proceedings of Colloque International sur la Fouille de Textes (2004)
14. Cohen, W., Ravikumar, P., Fienberg, S.: A comparison of string distance metrics for name-matching tasks. In: IJCAI 2003. Proceedings of the International Joint Conference on Artificial Intelligence, Workshop on Information Integration on the Web pages (2003)
15. Ceausu, V., Després, S.: Towards a text mining driven approach for terminology construction. In: Proceedings of the 7th International conference on Terminology and Knowledge Engineering (2005)
16. Biébow, B., Szulman, S.: A linguistic-based tool for the building of a domain ontology. In: Proceedings of the International Conference on Knowledge Engineering and Knowledge Management (1999)
17. Ceausu, V., Després, S.: Alignement de ressources sémantiques à partir de règles. In: EGC 2007. Dans la revue RNTI (Revue des Nouvelles Technologies de l'Information), numéro spécial (2007)

# Purpose-Aware Reasoning about Interoperability of Heterogeneous Training Systems

Daniel Elenius, Reginald Ford, Grit Denker, David Martin, and Mark Johnson

SRI International, Menlo Park, California, USA  
firstname.lastname@sri.com

**Abstract.** We describe a novel approach by which software can assess the ability of a confederation of heterogeneous systems to interoperate to achieve a given purpose. The approach uses ontologies and knowledge bases (KBs) to capture the salient characteristics of systems, on the one hand, and of tasks for which these systems will be employed, on the other. Rules are used to represent the conditions under which the *capabilities provided* by systems can fulfill the *capabilities needed* to support the roles and interactions that make up each task. An Analyzer component employs these KBs and rules to determine if a given confederation will be adequate, to generate suitable confederations from a collection of available systems, to pre-diagnose potential interoperability problems that might arise, and to suggest system configuration options that will help to make interoperability possible. We have demonstrated the feasibility of this approach using a prototype Analyzer and KBs.

## 1 Introduction

Much has been achieved in coaxing disparate resources to work together synergistically, but our ambitions often exceed the means at our disposal. Initiatives to lash together military training and testing systems are among the most ambitious. Since the 1980s, the means have evolved from dedicated engineered interfaces, to generalized interoperability middleware and protocols, and more recently to service-oriented architectures (SOA) and improvisational all-comer data access strategies such as the U.S. DoD Netcentric Data Strategy (NCDS) and the NATO Network Enabled Capability (NNEC). However, universal improvisational “plug and play” is still beyond our grasp. Successful interoperability is typically achieved only after lengthy planning, and it is not uncommon for apparent successes to exhibit subtle unacceptable anomalies.

For a nontrivial set of resources that have been developed independently to meet the unique needs of their sponsors/owners, it is impossible in general to answer the unbounded question “are these resources interoperable with one another”? However, when qualified as “are resources  $R_1, \dots, R_n$  interoperable with each other for purposes  $P_1, \dots, P_n$ ”, the question becomes manageable. This paper describes the application of Semantic Web technologies to enable automated

“purpose-aware” reasoning about interoperability, as part of the Open Netcentric Interoperability Standards for Training and Testing (ONISTT) program<sup>1</sup>

ONISTT is developing (a) ontologies to express the capabilities needed to perform mission-related tasks, (b) ontologies to express the capabilities available from prospective resources for executing those tasks, and (c) an automated reasoner/analyzer that can determine if the collective capabilities of some subset of candidate resources can satisfy the needs of a specific target mission. If multiple subsets can satisfy the needs, the analyzer ranks the relative goodness of each subset (with respect to relatively simple metrics). Because “goodness” of fit among resources is not always a simple “yes” or “no”, the analyzer may give a qualified answer, leaving it to human judgment to say whether the level of interoperability is “good enough” or “the best obtainable.” The analyzer can also suggest system configuration options.

In ONISTT proof of concept demonstrations, prototype KBs were populated with declarative information about the interoperability needs of particular training events, and the specific capabilities provided by operational and training/testing resources. Several alternative compositions of resources were presented to the prototype analyzer software. The analyzer correctly evaluated potential compatibilities and conflicts among the resources.

## 2 Background

The training community distinguishes between *live*, *virtual*, and *constructive* training systems. In a live training system, real personnel and vehicles are augmented with instrumentation such as GPS trackers and firing simulators, so that combat situations can be trained as realistically as possible without the need to fire real ammunition. Virtual systems – used to train crews of tanks, aircraft, etc. – also involve real personnel, and the controls they use are often very close to those on the real vehicle or article, but their view of their surroundings is through computer-generated images. Constructive systems are done completely on a computer, with a human controller who decides where troops go, and so on.

For larger training exercises, there is often a need to connect different training systems, and the systems can be of all three kinds (i.e. live, virtual, and constructive). We refer to this as LVC training, and the set of systems used is called a *confederation*.

The problem with LVC training is that different training systems are usually not built to be used together. Although the “technical interoperability” of exchanging data among systems is often achieved, there are many ways in which “substantive interoperability” can fail to occur. We have found it useful to distinguish between four levels of interoperability, which we derived from the Levels of Conceptual Interoperability Model (LCIM) [\[1\]](#):

<sup>1</sup> ONISTT is supported by the office of the Deputy Under Secretary of Defense/Readiness/Readiness and Training Policy and Programs (DUSD/R/RTPP). The views expressed in this paper are those of the authors and not necessarily those of DUSD/R/RTPP.

- Network interoperability. Common networking stack and medium. Systems can physically exchange digital messages.
- Syntactic interoperability. Common syntactic structure of messages.
- Semantic interoperability. Systems have a common understanding of the meaning of concepts used in communication.
- Behavior interoperability. Systems are compatible with regard to the actions they take on receiving messages, and the circumstances under which they send messages.

Many interoperability solutions have been implemented or proposed for training and similar systems [2,3,4]. However, none of these solutions encompass the whole range of interoperability problems. In particular, they usually only specify standards for networking and syntax, but fall short of a comprehensive semantics, and fail to address behavioral problems [1].

The end result is that one often does not know whether two systems will be truly interoperable until it has been tried. Much of the knowledge about interoperability and its problems resides in the minds of the engineers responsible for the systems. Currently, before any major LVC training exercise, a so-called BOGSAT (Bunch of Guys Sitting Around a Table) is summoned to work out interoperability issues. [5] This process is error-prone, costly, and time-consuming. Furthermore, the outcome of the process may not be stored for reuse in the future, leading to redundant efforts.

### 3 The ONISTT Approach

In order to enable improvised training events on short notice, we need to automate a significant portion of the planning and setup of these events.

The solution we propose is based on Semantic Web technologies. Our approach is summarized in Figure 1. First (1), we develop what we call *referents* for all the training environments, tasks, infrastructures, and systems that are relevant to the problem. By “referent,” we mean the most accurate and complete information available about the entity in question. Referents can include semi-formal models such as UML diagrams. These referents are then formalized into OWL ontologies (2). We discuss the ONISTT ontologies in more detail in Section 4. On the basis of these ontologies, the human planner defines an event and proposes a partial or full confederation for the event (3). We have developed a plugin to Protégé [6] to facilitate the formalization of the event. Then a piece of software called simply “the Analyzer” uses the data from the planner and the ontologies to verify the given confederation or generate a verified confederation, based on domain-specific rules and general reasoning technology (4). *Verified* means simply that the confederation passes all the interoperability tests that apply for the given purposes and circumstances. The Analyzer is described in more detail in Section 5. The Analyzer either returns a verified confederation (5a) and *configuration artifacts* (to be explained in following sections), or notifies the planner of what went wrong (5b), the more common situation. The Analyzer can report



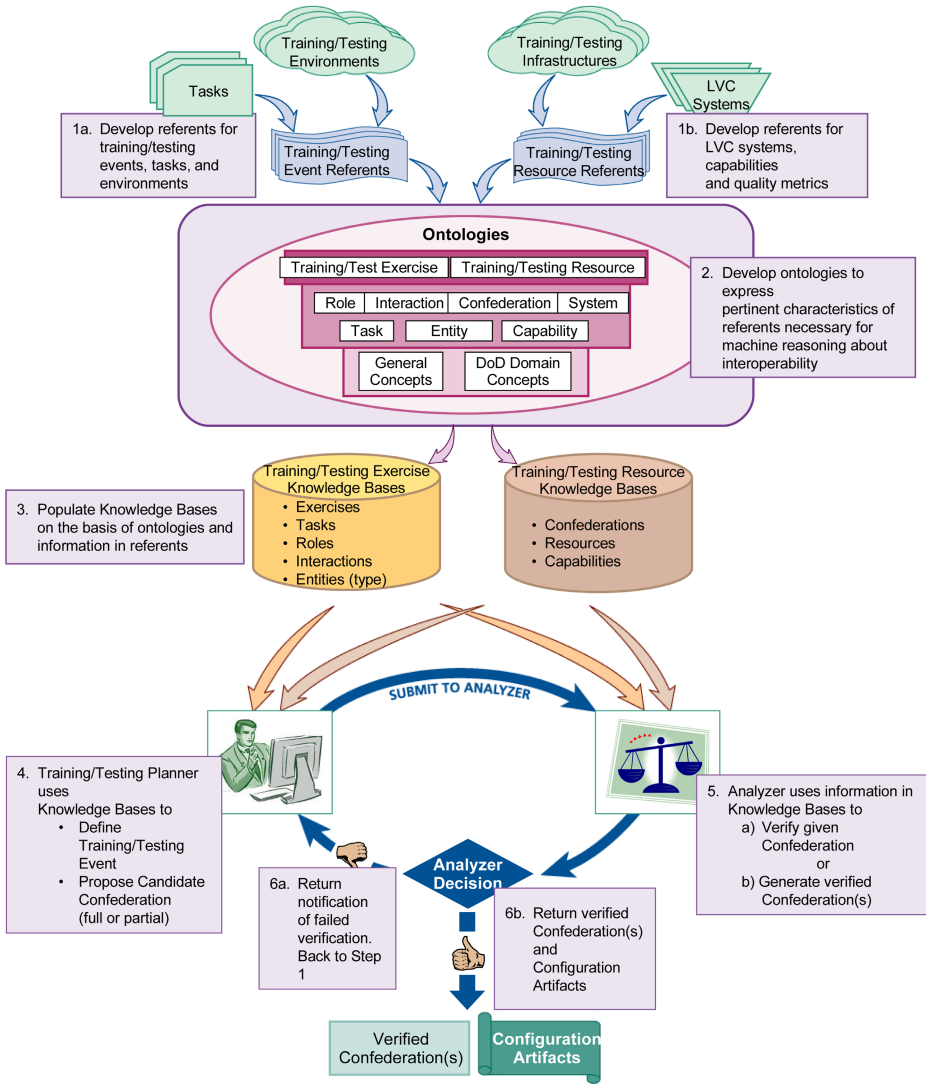


Fig. 1. The ONISTT approach and methodology

anything from minor warnings to major roadblocks to a successful event. At this point the planner can take corrective action and submit a new proposal to the Analyzer, starting the process over. Alternatively, the planner may choose to live with the problems that the Analyzer found. Our goal is that the output of this process should be at least as good as the output of a traditional BOGSAT, while being cheaper, less time-consuming, and more reusable.

## 4 Ontologies

### 4.1 Ontology Language and Structure

We use OWL to express our ontologies. The primary reasons for selecting OWL are pragmatic rather than technical (e.g., we do not make use of an OWL DL subsumption-based reasoner, see Section 5). For our approach to be successful, other groups will need to adopt it, and mature ontology engineering tools will be required. We have created some prototype ontologies, but in the long run, we cannot ourselves encode all the knowledge pertaining to this domain. Therefore, we need a language and tools that other groups can readily pick up and use. The DoD and others are converging on OWL for ontology expression. Although fully mature OWL engineering environments are not yet available, open-source and commercial tools are growing in number and capability.

Our ontology set is highly modularized, relying heavily on the OWL import mechanism. The top-level ontology, `onistt.owl`, has a number of relatively “naked” concepts and some properties between them. Each concept is elaborated in one or more special ontologies. To describe the full richness of the training domain, we also need ontologies of training systems, communication standards, virtual terrain, military vehicles and weapons, and so on. We have made a start at the ontologies we found necessary for the scenarios we have worked with. In total, we have about 60 relatively small prototype ontologies. The intent is that different organizations should be responsible for fleshing out their own ontologies.

### 4.2 Ontology Overview

The ONISTT ontology, as shown in Figure 2 has three complementary parts that compose a *Deployment*:

- An *Exercise* has *Task* objectives (i.e., “purposes”), from which an assemblage of needed capabilities is derived.
- A *Confederation* is a collection of *Resources* whose individual capabilities may be composed to satisfy the sum of the capabilities needed.
- A set of *Assignments* match the capabilities provided by individual confederation resources with specific capabilities needed to conduct the exercise.

Sections 4.3 to 4.5 explain how Exercise, Confederation, and Assignment properties and concepts are tailored so that the Analyzer, as described in Section 5, can satisfy the objectives identified in Section 3. Although the ontology design reflects the intended use of our tool suite in exercise planning, we think that similar purpose-resource matching ontologies could be developed for other domains. Also, many of the imports are prototype generic ontologies we developed for technical domains, for example the spatial ontologies discussed in Section 4.4.

### 4.3 Purpose Ontologies

The ONISTT “purpose” ontologies are shown on the left in Figure 2. The purpose of a training *Exercise* is to improve the proficiency of military forces in

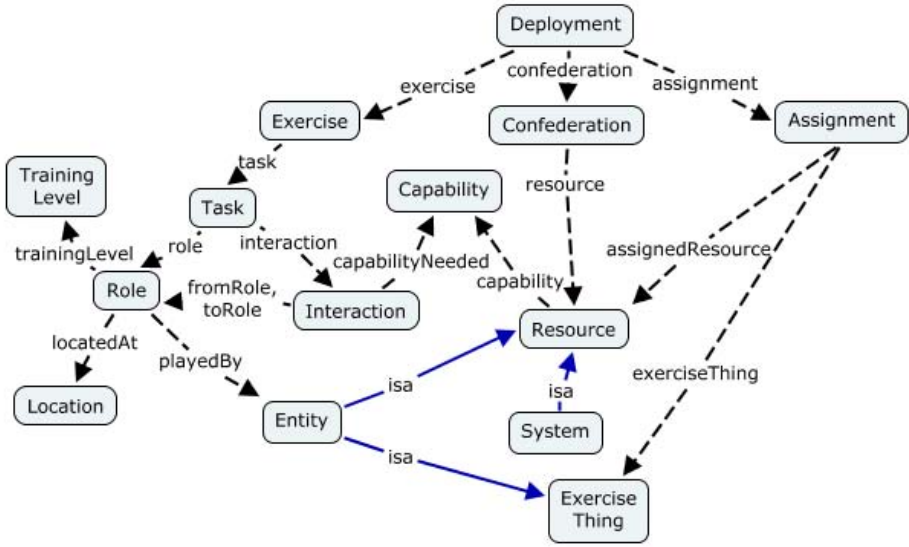


Fig. 2. Top-level ONISTT ontology concepts

the conduct of one or more *Tasks*. Joint training tasks (e.g., Joint Close Air Support) are defined in the Universal Joint Task List (UJTL). Responsibility for executing a task is distributed among several *Roles* (e.g., forward observer). A *Role* is *playedBy* an *Entity*. The UJTL describes friendly force roles. However, a training exercise must provide its own opposing force, and so surrogate hostile force roles are included in ONISTT KBs.

Our ontology includes a prototype taxonomy of *Role* subclasses and their associated properties and restrictions. For example, the *playedBy* property of the *AirborneOrdnanceDeliveryRole* has *allValuesFrom* and *someValuesFrom* restrictions that require individuals to belong to the *AirPlatform* class.

Each role in a task is responsible for performing certain actions. A complete description of tasks would include actions that can be assigned to a single role. However, the scope of ONISTT is interoperability, and therefore the only actions defined in the ontology are *interactions* among roles. The initiator of an interaction is designated by the *fromRole* property, and the other end is the *toRole*. Each interaction is associated with one or more needed *Capabilities*. Interactions between opposing forces are often the most critical to planning a training exercise because they require the support of special training capabilities, for example communications and simulations needed to effect a weapons engagement. However, training environments often differ from real operational environments in ways that require some interactions even among friendly force roles to be supported by auxiliary training system resources.

In defining an exercise, the planner may designate different *TrainingLevel* objectives for different roles. The particular interactions that are required between roles, and the required qualities and characteristics of the interactions, may

depend on the training level. An exercise scenario defines other constraints on roles, such as *Location*.

#### 4.4 Resource Ontologies

The ONISTT “resource” ontologies, which specialize the *Resource* class shown in the middle of Figure 2, are designed to support KBs that describe (1) the capabilities of individual resources, and (2) the bundling of resources.

Since most systems and other assets used in training exercises are multipurpose and adaptive, ONISTT ontologies are designed to describe resources fully, apart from any intended application. The Analyzer software selects facts from the “resource” KBs to determine whether a particular combination of resources is capable of performing a particular set of tasks in a particular context, as described in Section 4.5.

Most resources in LVC training are not accessible atomically but are bundled with other resources. The highest-level collection of resources in our ontologies is the *Confederation*. A Confederation is typically a loose and temporary aggregation of *Resources* – systems, infrastructure, and other assets – that have a more stable, though not necessarily fixed, identity. Resources can have multiple *subresources*, which in turn have subresources, to an arbitrary number of levels. Dependencies among subresources constrain whether they can be used in a mix-and-match fashion or are coupled together.

The main subclasses of Resource are *Entity* and *System*. An entity is an exercise participant, for example an F/A-18 aircraft. Training systems are often very complex, with multiple subsystems, which are recorded in KBs as subresources whose range is also the System class. An example is live training instrumentation that tracks the position of exercise entities, performs weapons effect simulation when entities engage each other, records data for After Action Review (AAR), and so on.

Each Resource has one or more *Capabilities*. One subclass of interest to the ONISTT domain is *Representation Capability*, for example the ability of constructive modeling and simulation (M&S) to simulate an F/A-18 entity. Another subclass is *Communication Capability*, for example the ability of a training system to communicate with other systems using the Test and Training Enabling Architecture (TENA). Another is *Terrain Capability*, for example the ability to determine line of sight between two earth locations.

Quality and type characteristics are specific to Resource types. For example a Representation Capability has a *motionModel* property. A *Constructive Representation Capability* is restricted to motion models of the *Simulated Motion Model*. One of the quality characteristics of a simulated motion model is a boolean property indicating the physical realism of a simulated entity’s motion in turns (smooth or jerky). A few relatively simple quality metrics were used for the initial feasibility demonstration. One of the major challenges for ONISTT is expressing qualities and characteristics of capabilities in a standard way so that the capabilities provided by resources can be compared with the capabilities needed [78,9].

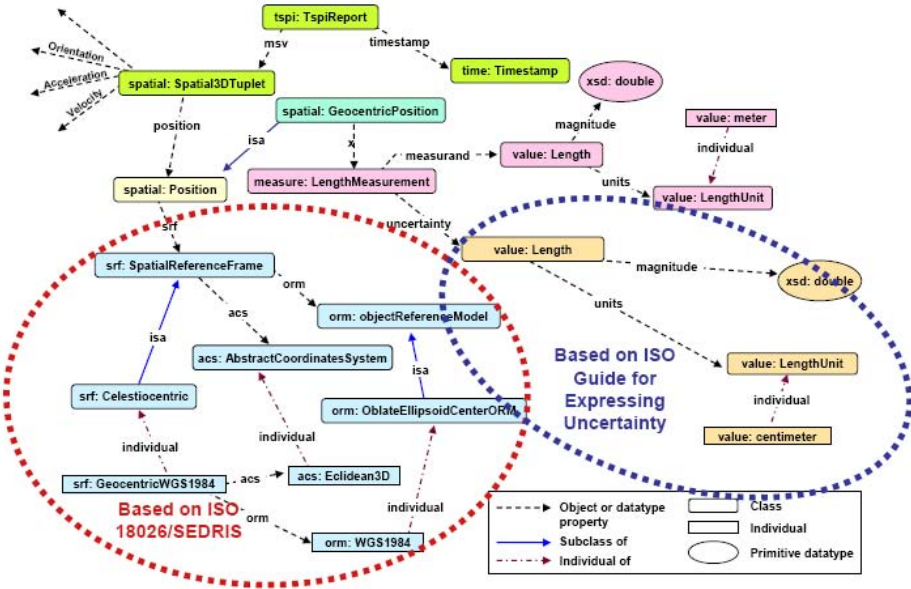


Fig. 3. Slice of the TSPI report ontology

An example of standard definition of capability characteristics and qualities is shown in Figure 3. Many training systems exchange Time-Space Position Information (TSPI). Because the shape of the earth is irregular, it is difficult to express geographic coordinates precisely and unambiguously, and many mathematical schemes have been developed. A recipient of TSPI must know the spatial reference frame of the data to interpret it correctly. The ISO 18026 Spatial Reference Model standardizes geographic reference frames. We translated the ISO 18026 spatial reference frames most commonly used in training systems to an ontology. If an interaction requires two systems to exchange TSPI data, the Analyzer can examine if they use the same reference frame or, if not, whether they have a capability to translate between the two frames. If translation is required, the Analyzer can determine if the translation will be perfect, or if certain attributes, such as line-of-sight calculation, may be distorted.

We developed approximately 20 prototype KBs describing individual systems and capabilities to support the feasibility demonstration. A full ONISTT operational capability will require hundreds. Fortunately, a comprehensive description of each resource and capability is not necessary. The level of detail needed in KBs is limited to facts that are directly relevant to assessing interoperability among resources.

### 4.5 Assignment Ontologies

The “Assignment” ontology, as shown on the right in Figure 2, connects the “purpose” and “resource” ontologies. The objective of exercise planning is to

specify a valid *Deployment* that defines a *Confederation* of resources to meet the needs of a particular *Exercise*. An individual Exercise is a set of individual Tasks, Roles, Interactions, and Entities selected from KBs developed in accordance with the ontologies explained in Section 4.3. An individual *Confederation* is a pool of systems and other resources that are selected from KBs developed in accordance with Section 4.4. Individual *Assignments* assign one or more Confederation Resources to each Entity defined in the Exercise. The Analyzer software determines whether the assigned resources have all the requisite capabilities and capability qualities and characteristics needed to support all Interactions among all Entities.

## 5 Analyzer

We mentioned in Section 3 the use of Analyzer software to provide feedback on the suitability of a given confederation. Here, we describe the implementation of this software, which brought up some issues of more general concern in using Semantic Web technologies in the real world.

While the ontologies provide the background knowledge of the problem domain, the Analyzer provides the computational side of the automation. Its job is to look at the information provided and draw conclusions according to a set of *rules*.

We wanted to frame the operation of the analyzer as a problem of logical deduction, as this would give us a clear semantics of what the analyzer does. We considered (and tried) several different possibilities.

The most natural approach would be formulate the operation of the Analyzer as an OWL subsumption check. This would allow us to use any OWL DL reasoner right out of the box. However, we found it impossible to formulate the problem in this way, mainly because OWL DL is very restrictive with the use of quantifiers and variables.

Another approach was to use OWL augmented with SWRL [10] rules. However, SWRL also proved insufficient to express the operation of the Analyzer. This is explained in more detail below

A third approach was to translate the OWL KB to First-Order Logic (FOL) and axiomatize the operation of the Analyzer in FOL. We tried this (with the SNARK<sup>2</sup> theorem prover), but we found this to be too slow and sensitive to small changes in the problem formulation (as is often the case with applications of FOL theorem proving).

In the end, we decided to write the Analyzer as procedural code. We needed a tight integration of procedural code (the Analyzer) and declarative content (the KB). Prolog is a natural choice in this kind of situation. A large fragment of OWL, called Description Logic Programs (DLP) [11], can be readily translated to Logic Programs [12] (the logical underpinnings of Prolog). Prolog can also be used as a programming language for writing procedural code. In particular, we chose to use

---

<sup>2</sup> <http://www.ai.sri.com/snark/>

XSB Prolog<sup>3</sup> in order to avoid the well-known problems that ordinary Prolog has with recursive structures such as equivalent classes or properties.

The Analyzer implementation consists of a pair of software components: A *Translator* that translates from the OWL+SWRL KBs into XSB Prolog, and the *Analyzer Core* that runs domain-specific tests on the information in the knowledge base.

## 5.1 Translator

Fortunately, there is already a “standard” way [11] to translate a fragment of OWL DL, called DLP, to Logic Programs. It is also straightforward to extend this translation to handle SWRL rules, because these rules are just Horn clauses, directly expressible in Logic Programs (and thus in Prolog). We will describe some of the salient features of our translation approach in the following.

A naive translation to Prolog (as done by for example the `dlpconvert` tool, which is part of the KAON2 toolset<sup>4</sup>) encodes OWL classes as unary Prolog predicates, OWL properties as binary predicates, and OWL axioms to Prolog rules.

However, we have chosen a different encoding, with several advantages that will be explained shortly. Our target Prolog statements include only two predicates, *inst/2* and *value/3*, with the intuitive meaning that *inst(x,y)* means that *x* is an instance of *y*, and *value(x,y,z)* means that *x* has value *z* on property *y*<sup>5</sup>. In this encoding, all class, property, and instance names appear as Prolog *terms*. We may say that these entities are *reified*, as they appear as objects in the target language.

This encoding allows more types of queries to be answered. For example, we may ask for all classes that John is an instance of, `?- inst(John,X)`, with answers like `X = person; X = animal; X = thing`. Note that the answers are not just asserted instance relationships, but also all relationships that can be inferred according to the semantics of the DLP+SWRL fragment. Another type of query is to ask for all known or inferred property values of an instance, `?- value(john,Prop,Val)`, and get answers like `Prop = sibling Val = Dave; Prop = sibling Val = Alice; Prop = age Val = 23`.

This approach gives us more expressive power in formulating queries. Not just instances, but also classes and properties, become objects that we can refer to in queries. For example, we can use a class as the first or third argument to the value predicate. This is usually called classes-as-instances, or just reification, and is a feature of OWL Full. The jury is out on whether this is actually a desirable feature in a language. One view is that a perceived need for this feature indicates that something is wrong with the ontology. This may be true in a “pure” OWL

<sup>3</sup> <http://xsb.sourceforge.net/>

<sup>4</sup> <http://owltools.ontoware.org/>

<sup>5</sup> We could have encoded everything using the value predicate by making “type” a property. However, we opted for a more intuitive encoding that avoids special properties like `rdf:type`.



DL ontology that is designed with OWL DL reasoning in mind. However, for the kinds of queries we do, we have found that classes-as-instances is quite a useful feature in a few select places in our ontologies.

## 5.2 Analyzer Core

Once the OWL+SWRL knowledge base has been translated, it can be loaded into our Prolog engine, and used by the Analyzer Core.

The Analyzer Core is a piece of code, currently a few hundred lines, that is also written in XSB Prolog. While the translated knowledge base uses XSB in a declarative way by just stating the facts, the Analyzer Core is a procedural program that queries the knowledge base in various places. This is a very flexible approach to integration of programming and reasoning.

More specifically, the operation of this component is as follows. It looks at each pair of Roles that have an Interaction between them. It then compares the Capabilities needed for that type of Interaction with the Capabilities provided by the Resources that are used to represent the Role. The resources may be sufficient, insufficient, or somewhere in between. In the real world, the latter is usually the case. Therefore, the Analyzer generates warnings for conditions that may be problematic but not fatal. Each condition is encoded as a rule in the Analyzer.

Let us look at one of these conditions, *unrealistic\_motion*. This condition occurs when we have an interaction requiring line of sight (LOS), for example, a DirectFire interaction, *and* the fromRole is represented by a virtual system (such as a tank simulator). Under these conditions, we require that the toRole has smooth motion modeling. An example of where this is not true is when the toRole is represented by a constructive system where units move between waypoints without smoothing out the curves (i.e., they have infinite angular acceleration). This causes the users of the virtual system to see units “jumping around” in an unrealistic way on their 3D displays. This leads to a loss of realism, and is also a “fair fight” issue, since it is hard to target units that are moving in a way that is physically impossible. The Prolog encoding of this rule is as follows:

```
% I - Interaction
% FRes - set of resources that represent fromRole
% TRes - set of resources that represent toRole

realisticMotion(I,FRes,TRes) :-
    realisticMotion_cond(I,FRes), !, realisticMotion_constraint(TRes).

realisticMotion(_I,_FRes,_TRes).

realisticMotion_cond(I,FRes) :-
    inst(I,'interaction:LOS_Interaction'),
    member(FS,FRes),
    value(FS,'onistt:subresource',FSs),
    value(FSs,'system:representationCapability',FRC),
    inst(FRC,'system:VirtualRepresentationCapability').
```



```

realisticMotion_constraint(TRes) :-
member(TS,TRes),
value(TS,'onistt:subresource',TSs),
value(TSs,'system:representationCapability',TRC),
value(TRC,'system:motionModel',MMT),
value(MMT,'motion:smoothMotion',true).

```

If the `realisticMotion` procedure fails, then the condition occurs (i.e., if we do not have `realisticMotion`, then we have unrealistic motion). The top-level rule is an implication that must be satisfied; *if* `realisticMotion_cond` holds, *then* `realisticMotion_constraint` must also hold.

Several things are worth pointing out in this example.

- The rule is stated in a *positive* way – unless we can prove the absence of a warning condition, we assume that the condition holds. This means that a warning condition cannot be avoided simply because of lack of information.
- We see that the Analyzer queries the knowledge base in many places, that is, it invokes the *value* and *inst* predicates. Each of these calls can involve arbitrary DLP reasoning. For example, the `onistt:subresource` property is transitive, so the value calls using this property will return a transitive closure. This is what we meant by a flexible integration of programming and reasoning, above.
- Different rules have different sets of arguments. In this case, we need the Interaction, the from-resources, and the to-resources. Other rules are slightly different. Also, some rules have return values. For example, the rules to check whether two roles can communicate also return the communication path (we call this a *configuration artifact*).

Other examples of conditions are lack of training system or tactical communications (where needed) or uncorrelated terrain data, which can cause problems like tanks hovering in the air, or airplanes flying through mountains.

The set of rules to check for problematic conditions is meant to be extensible. Ideally, one should be able to specify these rules in a declarative way – for example in SWRL – so that they can easily be inspected, edited, and so on. This was also our original approach. However, we found that the expressiveness of SWRL was not sufficient to encode the rules. To see why this is so, first consider the requirements for the Analyzer. The Analyzer is basically a *function* that takes one argument, the Deployment to check, and returns the results of the analysis, that is, warnings for some of the interactions in the Deployment, and configuration artifacts. These results could be encoded as ontological objects (e.g., a Warning class, with subclasses for different types of warnings), and we would then like the Analyzer to create and return the appropriate instances. However, SWRL rules cannot create new instances, or indeed return any type of structured data, unless the data is part of the “input” arguments to the rule. We *can* do this in Prolog, by using compound terms, but SWRL is Datalog (i.e. function-free), and thus it does not have compound terms. One option that we

plan to explore for the purpose of making the Analyzer rules more transparent is Functional RuleML<sup>6</sup>

## 6 Evaluation

Our work on the ontologies as well as the analyzer has been driven by a number of use cases [13]. Each use case consisted of an exercise and several different confederations (and assignments) for that exercise. Working up the use cases forced us to develop the ontologies of the systems and tasks involved to a considerable level of detail. The cases were inspired by real-world exercises and systems with which we had previous experience, and were designed to provide a realistic level of complexity. Some of our use cases were such that we knew what the results should be, and we could verify that the analyzer came up with those results. We also developed new and more complex scenarios, where we did not know the answers beforehand. The analyzer returned useful warnings and configuration artifacts, such as mediated communication paths. We have not yet tried the analyzer on a real-world exercise ahead of time. Finding an appropriate exercise to work with is high on our agenda.

There is much to gain by applying Semantic Web technology to our domain, in terms of automation, reduction of costly labor, and new functionality. At the same time, the approach requires a large one-time investment in ontology development before it can be widely adopted. It is our contention that the benefits are large enough to motivate the costs, especially since the ontologies can be reused in many different contexts.

## 7 Conclusions

We have described a novel approach by which software can assess the ability of a confederation of heterogeneous systems to interoperate to achieve a given purpose. This approach uses ontologies and KBs to capture the salient characteristics of systems, on the one hand, and of tasks for which these systems will be employed, on the other. Rules are used to represent the conditions under which the *capabilities provided* by systems can fulfill the *capabilities needed* to support the roles and interactions that make up each task. An Analyzer component employs these KBs and rules to determine if a given confederation will be adequate, to generate suitable confederations from a collection of available systems, to pre-diagnose potential interoperability problems that might arise, and to suggest system configuration options that will help make interoperability possible.

Representing the capabilities of systems and reasoning about interoperability are notoriously difficult problems, in their general forms. Solutions to these problems have potential value in a number of domains and applications. In military settings, such as complex training exercises, it is often a top priority to minimize

---

<sup>6</sup> <http://www.ruleml.org/fun/>

the engineering effort, and maximize the flexibility, associated with the deployment of “improvised” systems of systems. The work described here demonstrates a promising way forward. A key enabler of this approach is the explicit representation of purpose (i.e., tasks, roles, interactions, and the capabilities required for their fulfillment). We have found that assessing interoperability for a given purpose is considerably more manageable than general, unconstrained forms of the interoperability problem.

## References

1. Tolk, A., Turnitsa, C.D., Diallo, S.Y.: Implied ontological representation within the levels of conceptual interoperability model. *International Journal for Intelligent Decision Technologies (IDT)* 2 (to appear, 2008)
2. Distributed Interactive Simulation Committee of the IEEE Computer Society: IEEE standard for distributed interactive simulation – application protocols, IEEE Std 1278.1a-1998 (1998)
3. Simulation Interoperability Standards Committee of the IEEE Computer Society: IEEE standard for modeling and simulation (M&S) high level architecture (HLA)-framework and rules (2000) IEEE Std 1516-2000
4. Hudgins, G.: DoD ranges interoperability reuse achievable through the Test and Training Enabling Architecture, TENA. In: *International Telemetering Conference*, pp. 23–27 (2006)
5. Johnson, M., Ford, R., Shockley, J., Giuli, R., Oberg, S., Beebe, M.: Integration of CCTT and JCATS in an LVC exercise. In: *Simulation Interoperability Workshop, 04E-SIW-066* (2004)
6. Knublauch, H., Fergerson, R., Noy, N., Musen, M.: The Protégé OWL plugin: An open development environment for semantic web applications. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) *ISWC 2004*. LNCS, vol. 3298, pp. 229–243. Springer, Heidelberg (2004)
7. RPG Special Topic: Fidelity. Technical report, Defense Modeling Simulation Office (DMSO) (2000)
8. Davis, P.K., Anderson, R.H.: Improving the composability of DoD models and simulations. Technical report, RAND National Defense Research Institute (2003)
9. Kasputis, S., Oswalt, I., McKay, R., Barber, S.: Semantic descriptors of models and simulations. In: *Simulation Interoperability Workshop, 04F-SIW-070* (2004)
10. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosz, B., Dean, M.: SWRL: A Semantic Web Rule Language Combining OWL and RuleML (2004)
11. Grosz, B.N., Horrocks, I., Volz, R., Decker, S.: Description logic programs: Combining logic programs with description logic. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) *ISWC 2003*. LNCS, vol. 2870, Springer, Heidelberg (2003)
12. Loyd, J.W.: *Foundations of Logic Programming*, 2nd edn. Springer, Heidelberg (1987)
13. Ford, R., Hanz, D., Elenius, D., Johnson, M.: Purpose-Aware Interoperability: The ONISTT Ontologies and Analyzer. In: *Simulation Interoperability Workshop, 07F-SIW-088*, Simulation Interoperability Standards Organization (2007)

# A Collaborative Semantic Web Layer to Enhance Legacy Systems

Alfio Gliozzo<sup>1</sup>, Aldo Gangemi<sup>1</sup>, Valentina Presutti<sup>1</sup>, Elena Cardillo<sup>2</sup>, Enrico Daga<sup>2</sup>,  
Alberto Salvati<sup>2</sup>, and Gianluca Troiani<sup>2</sup>

<sup>1</sup>Laboratory for Applied Ontology, ISTC-CNR, Rome, Italy

<sup>2</sup>URT-CNR, Rome, Italy

**Abstract.** This paper introduces a framework to add a semantic web layer to legacy organizational information, and describes its application to the use case provided by the Italian National Research Council (CNR) intraweb. Building on a traditional web-based view of information from different legacy databases, we have performed a semantic porting of data into a knowledge base, dependent on an OWL domain ontology. We have enriched the knowledge base by means of text mining techniques, in order to discover on-topic relations. Several reasoning techniques have been applied, in order to infer relevant implicit relationships. Finally, the ontology and the knowledge base have been deployed on a semantic wiki by means of the WikiFactory tool, which allows users to browse the ontology and the knowledge base, to introduce new relations, to revise wrong assertions in a collaborative way, and to perform semantic queries. In our experiments, we have been able to easily implement several functionalities, such as expert finding, by simply formulating ad-hoc queries from either an ontology editor or the semantic wiki interface. The result is an intelligent and collaborative front end, which allow users to add information, fill gaps, or revise existing information on a semantic basis, while keeping the knowledge base automatically updated.

## 1 Introduction

A legacy information system can be defined as any information system that significantly resists modification and evolution. Legacy systems are affected by problems such as lack of documentation, obsolete hardware and cost of maintenance software. On the other hand, most of the systems currently in place in large institutions and companies belong to the aforementioned category. Therefore, the new trend is to develop methodologies to allow the information to migrate from legacy systems to more flexible data structures that enable interoperability, reusability, and integration with the current Semantic Web (SW) technologies.

In this paper we propose a SW-based solution for the problem above, and we describe its application to the use case provided by the Italian National Research Council (CNR) intraweb. The main goal of this project is to develop a SW layer on top of the databases and web publishing systems that are currently in place at the CNR.

CNR is the largest research institution in Italy, employing around 8000 permanent researchers, organized into departments and institutes. Its 11 departments are focused

on the main scientific research areas. Its 112 institutes spread all over Italy, and are subdivided into research units, which are characterized by different competences, research programmes, and laboratories.

The overall structure of the CNR is then rather complex: departments express a “research demand”, while institutes perform a “research supply”. The activity of planning and organization of such a huge institution is then strictly related to that of matching the research demand and the research supply. It can be performed only by having in mind a global picture of the interrelations between the entities in such a huge network, an operation almost impossible without the semantic facilities provided by the recent ICT technology. As a matter of fact, only recently research units from different institutes and departments have slowly started some synergies, and at the cost of lengthy meetings and substantial push from a new set of 83 management units, called “progetti” (frameworks). 749 further units, “commesse” (“research programmes”) have been created in order to direct local projects and synergies, and 704 researchers lead them as chief scientists. Each research programme is structured into local workpackages that channel research funds to institutes.

The complexity of this structure has proved to increase the potentiality for synergies within the CNR, but also the effort for its maintenance, monitoring, channelling of external funds and requests, etc. An adequate support for extracting and matching the competence-related knowledge that is scattered within local research units appears more and more relevant. In this context, information sharing and interoperability is crucial from a project management perspective. For example, in order to achieve a particular subgoal of a research project, it is often necessary to look for external qualified and highly specialized human resources, while having a clearer picture of the competences spread in the various departments of CNR would allow to avoid the use of external resources, since the probability of finding the desired profile among the internal members of the organization is very high.

SW technologies contain viable solutions to overcome the problems above, so that we have developed a prototype system that allows data migration from legacy databases to a wiki portal. Our system employs SW technologies such as OWL ontologies, reasoning systems, text mining tools, and ontology-driven wiki site management.

The general architecture of our system is described in Section 1, while the remaining sections analyze each component. Section 2.1 describes the migration process we implemented to unify the information spread into the different legacy systems in place at CNR, while Section 2.2 describes the domain ontology we developed and its further population. Section 2.3 focalizes on the text mining component we developed to induce `on_topic` relations among instances of the ontology, while Section 2.4 illustrates the deployment of the so obtained ontology into the wiki portal. Pros and cons of the proposed approach are described in the evaluation section, while Section 4 concludes the paper illustrating the new application scenarios opened by this work.

## 2 General Architecture and Information Workflow

As introduced in the previous section, the main goal of the information workflow presented in this paper is to enhance accessibility and interoperability of the information

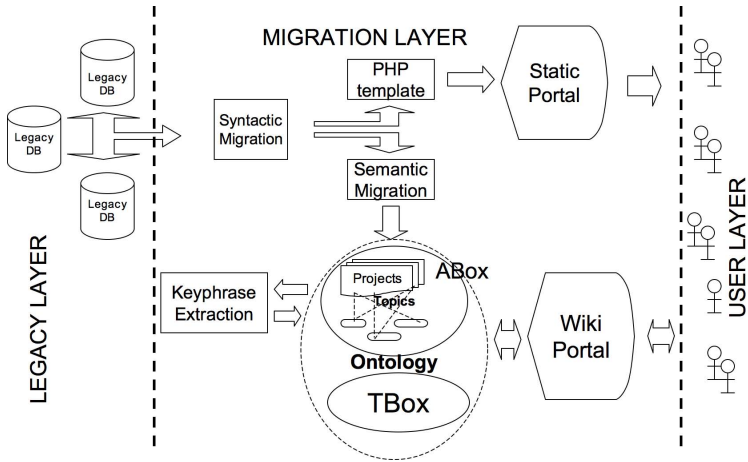


Fig. 1. The CNR semantic intraweb workflow and architecture

spread in different legacy systems, making it reusable by final users, and allowing them to interact with the resulting knowledge in a collaborative way. This section broadly describes the methodology and the algorithms we implemented to this aim. Details of each component will be illustrated in the appropriate subsections.

The overall system architecture is represented in Figure 1. The first component of our system performs a syntactic migration of the information spread into different CNR databases. The databases contain administrative and financial data, research organization data, and personal data of CNR employees. For privacy reasons, in this paper we focus on the semantic intraweb created for research organization data, which also supports a preliminary *expert finding* system. The result of this operation is a unified view of the overall CNR activities, including people involved, departments, research projects, and so on. This information is directly put at disposal of final users by means of an intraweb portal, presenting it by means of dynamically generated web pages. Details of this operation are described in Section 2.1.

Syntactic integration only provides a static and partial view of the information contained into the database. Basic semantic operations, such as expert finding and similarity reasoning, cannot be performed intellectually or with simple queries, given the huge amount of data actually contained into the databases. To this aim, we have performed a semantic migration, with the main objective of porting and then distilling knowledge from the information released by the previous integration. We have obtained a structurally richer representation, and on top of it reasoning and other cognitive operations can be performed. In particular, we have developed an ad-hoc domain ontology describing the information contained in the databases at a semantic level (see the TBox from Figure 1), and we have automatically populated it by implementing a semantic migration process that transfers the information from views on the database entries, to appropriate OWL-RDF code. Details of this step are illustrated in Section 2.2.

As a matter of fact, since a large part of the CNR data is composed of textual material (e.g. abstracts of research projects, descriptions of work done, internal reports, etc.), the benefits of semantics in this phase alone are limited to a more explicit, rigorous

maintenance of information contained in databases. However, this is not yet something that “makes a difference” to final users, and can convincingly support the need for a real migration to the semantic (intra)web.

Therefore, we have followed the direction of adopting automatic text mining techniques to further enrich the structure of the knowledge base (and eventually of the ontology). In particular, we have acquired `on_topic` relations, by adopting automatic key phrase extraction techniques. This operation is described in detail in Section 2.3. Based on this richer structure, the inferencing capabilities provided by OWL and SPARQL reasoners have been key to a substantial enrichment of the knowledge base (see Table 1).

Finally, we have adopted a wiki-based approach to deploy the knowledge base into a wiki site available to the final users. To this aim, we have exploited WikiFactory [4], an environment that can automatically generate a wiki portal mirroring an existing ontology. In addition, WikiFactory allows the final user to modify the ontology and the knowledge base (e.g. introducing new classes or properties, modifying the existing ones, introducing new individuals and property values, etc.) by simply modifying the generated wiki pages.

## 2.1 Syntactic Integration of Legacy Systems

All the components adopted for the syntactic integration are based on web services able to provide information from different knowledge sources into multiple standard formats, such as XML, RSS, SOAP etc. Each different legacy system can be accessed by means of an ad-hoc web service, providing information into a standardized XML format. These formats are transformed by applying appropriate templates. The system matches the XML retrieved from the web service to the template by means of XSLT datasheets, and returns the information organized in another (XML, RDF or OWL) format. To this aim, the system adopts different technologies (mainly Java and IBM Web DataBlade).

This simple strategy is used to provide information to external systems (for example, a web site of a CNR institute through RSS), to allow system integration, to retrieve knowledge expressed in RDF-OWL, and finally to build HTML pages for the current intraweb portal in place at CNR, already accessed by thousands of users.

The syntactic migration and integration of CNR data does not allow to reason over the knowledge contained in those data, because it only addresses data manipulation without any explicit assumption on the semantics (either in the linguistic or logical sense) of those data. For example, the reason why a certain table from a database is ultimately transformed into a certain part of a HTML page is not explicit (no logical semantics), and the associations between the terms used across the records of the databases are implicit (no linguistic semantics).

In order to add some semantics to the CNR data, we have adopted a twofold strategy: on one hand, an OWL conversion (see Section 2.2) of legacy data has been made by creating a template for each class from the CNR ontology, which is filled for each instance extracted from a database. On the other hand, a text mining-based enrichment of semantic relations among terms from textual records (see Section 2.3) has been also performed, and then formalized in OWL. This twofold strategy provides logical and linguistic semantics to a substantial amount of CNR data.



## 2.2 Representing and Reasoning on Legacy Information in OWL-RDF

In order to make the semantic migration effective, we have firstly developed an OWL ontology describing the CNR scientific organization. The ontology engineering process was rather simple as we took advantage from the existing XSD schemata defined to produce a set of HTML templates as presented in the previous section. Based on those templates, we produced a formal description of the domain ontology in OWL(DL).

Figure 3 shows the TBox of the ontology that we developed to describe the CNR scientific organization. It encodes the relations between individuals of classes such as Researcher, City, Department, Research Programme, Institute and Framework. The actual expressivity exploits a fragment of OWL(DL): cardinality restrictions, property range and domain, disjointness, transitive and symmetric properties, etc.

Then we have populated the ontology by exporting RDF code from the syntactic module described in section 2.1. The second column of Table 1 reports the size of the ontology collected after the simple migration from legacy data. After basic reasoning, 3148 individuals and 14695 property values have been created in the CNR knowledge base. In the next section, we will explain why these data increase so dramatically after applying learning techniques to unstructured data.

As a tool for the ontology lifecycle and reasoning over the large ABoxes created after reasoning with topics extracted via NLP and LSA (see next section), after some testing, we have decided to use TopBraid Composer, a commercial software based on the open source development platform Eclipse, providing advanced visualization and querying tools, as well as efficient and substantially bug-free interaction between the Pellet reasoner, the storage mechanism, and the interface.

## 2.3 Acquiring On-Topic Relations by Applying Text Mining Technologies

One of the main limitations characterizing typical OWL ontologies (at least for OWL1.0) is their weakness in modelling semantic proximity among concepts. For example, names of persons and organizations typically belong to different taxonomies, and just in a few cases they are actually related. On the other hand, semantic proximity is very well modeled by adopting geometrical models, such as the Vector Space Model [8] or contextual similarity techniques [2], elaborated in the Information Retrieval and Computational Linguistics areas.

Geometrical models for semantic proximity are at the basis of successful applications like search engines, while at the same time they constitute one of their bigger limitation. In fact, even if search engines are very powerful in providing information somehow related to the query, they are clearly not able to go deeper than that. For example, logic constraints cannot be imposed on the type of objects we are interested in during search, making the retrieval technology clearly inadequate for the new needs of the Web 2.0 and ultimately the semantic web.

<sup>1</sup> <http://www.loa-cnr.it/ontologies/CNR/CNR.owl>

<sup>2</sup> <http://www.topbraidcomposer.com/>

<sup>3</sup> <http://www.eclipse.org/>

<sup>4</sup> <http://www.mindswap.org/2003/pellet/>



**Table 1.** Ontology size before and after the reasoning and text mining procedures (original Italian class names are translated)

<b>ABox</b>	<b>before</b>	<b>after</b>
Department	11	11
Institute	112	112
ChiefScientist	704	704
Framework	83	83
Programme	749	749
Workpackage	1166	1166
Topic	499	3148
City	66	66
<b>Total individuals</b>	<b>3393</b>	<b>6042</b>
<b>owl:PropertyValue</b>	<b>14695</b>	<b>158441</b>
<b>TBox</b>	<b>before</b>	<b>after</b>
owl:Class	11	11
owl:DatatypeProperty	33	33
owl:FunctionalProperty	10	10
owl:InverseFunctionalProperty	7	7
owl:ObjectProperty	28	48
owl:Ontology	1	1
owl:TransitiveProperty	2	2
owl:SymmetricProperty	0	10

On the other hand, logical models, such as those represented by domain ontologies, are characterized by approximately opposite properties. They allow us to easily perform semantic queries, for example by reasoning over OWL-RDF models, or by imposing a semantics over a query language like SPARQL, but they typically rely on manually designed descriptions written in some formal language, which are typically very costly and not available on a large scale.

In the context of several research projects, we are following the direction of fusing empirical and logical approaches for knowledge representation, trying to integrate assessed text processing and information retrieval techniques with traditional knowledge engineering methods. The outcome of this integration would provide a much more powerful framework for knowledge representation, integration and acquisition to be used as a basic infrastructure for the SW.

To accomplish this goal, we followed the direction of acquiring `on_topic` relations to create new links among instances in the ontology. On topic relations are automatically induced via text mining, by analyzing the textual material connected to the instances in the ontology. We added a new class in the ontology, called `topic`, whose instances are different key-phrases extracted from documents, and we related them to entities in the ontology, such as research programmes, institutes, researchers, etc.

Extracting keyphrases from documents is a very well established technique in Natural Language Processing (NLP). In the literature, several methodologies have been proposed, ranging from applying supervised learning techniques [10], to pattern based approaches. Key-phrases are in general noun compounds, usually composed of 2 or 3 words, and can be identified by specifying syntactic patterns. Statistical measures are in

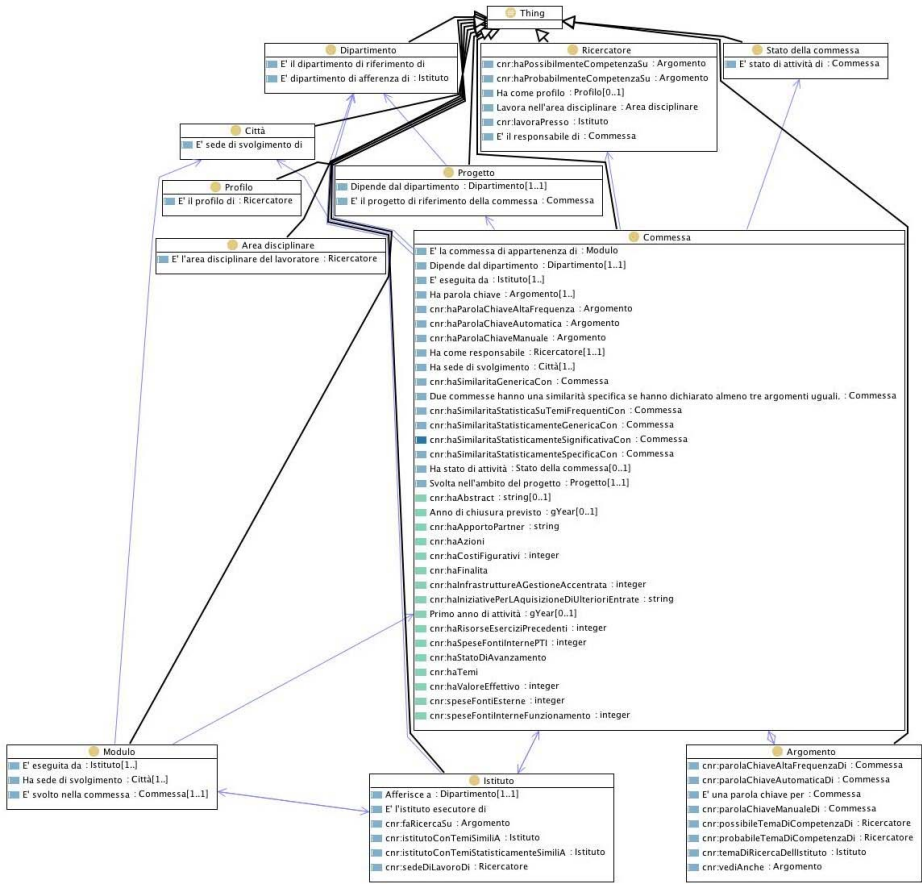


Fig. 2. TBox of the CNR ontology

general adopted to measure the internal coherence among words of the same terms and the distributional properties of the term as a whole inside documents in a corpus.

For the purposes of this work we implemented a novel approach for term extraction, based on Latent Semantic Analysis (LSA). Our approach identifies first a set of candidate terms from the whole document collections, by applying pattern-based approaches on the output of a Part of Speech tagger (e.g. all the sequences composed by Noun + determiner + nouns are candidate Italian terms). Then it filters out incoherent terms by estimating the mutual information between the compound words for each term (e.g. “new economy” is a term, while “new English” is not a term). Finally, it represents documents and terms in the LSA space, a geometrical space in which the similarity between them can be estimated by taking into account second order relations. Keyphrases for each document are then selected by looking for all the neighboring terms of the document vector in the LSA space. This process is illustrated by figure 4. Further details on the LSA technique adopted are reported in [3].

**Resource Form**

Name: ICT.P04.019\_5675

**Annotations**

rdfs:label

Intraweb semantico: gestione avanzata dell'informazione in organizzazioni complesse

**Other Properties**

cnr:commessaDiAppartenenzaDi

ICT.P04.019.001

ICT.P04.019.002

cnr:dependeDalDipartimento

Dip-ICT

cnr:segueDa

Ist-ISTC

cnr:haAbstract

Un intraweb è un web che comprende i nodi HTTP di una intranet. In molte organizzazioni gli intraweb sono un mezzo privilegiato per la gestione della conoscenza (Corporate knowledge management).  
 Le tecnologie semantiche possono essere sfruttate sui documenti presenti in un intraweb grazie alle dimensioni contenute del corpus, la disponibilità di modelli di riferimento definite.  
 Come caso di studio, si intende costruire l'IntraWeb Semantico (IWS) per la gestione evoluta della conoscenza del CNR. IWS è un sistema basato su informazioni retrieval avanzato, tecniche di elaborazione del linguaggio naturale, machine learning, ingegneria ontologica e linguaggi progettati per il semantic web.  
 Le funzionalità comprendono: gestione dei contenuti, potenziamento del motore di ricerca terminologico con componenti morfologici, multi-lingua, modellazione dei log e risorse linguistico-semantiche, integrazione di servizi, supporto alla decisione su documenti digitalizzati, mappatura di documenti su un modello prototipico, creazione di basi di conoscenza, modellazione di linee-guida per contratti e workflow, creazione di know-how comunitario (ex. semantic wiki).

cnr:haAnnoDiChiusuraPrevisto

2008

cnr:haFinalita

L'obiettivo di questa commessa è sviluppare una sofisticata piattaforma di gestione semantica dell'informazione contenuta all'interno di una intranet, e in prospettiva nel web.  
 Componenti di questa architettura sono moduli individuali di immediata applicazione che progressivamente arricchiscono un motore di ricerca e un sistema di gestione del contenuto di nuova generazione.

Fig. 3. An Instance of the Research Program class

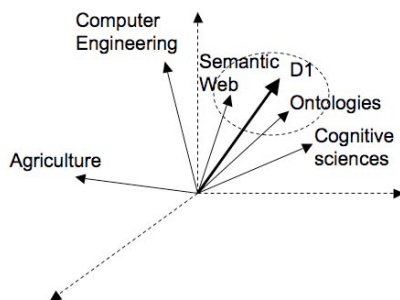


Fig. 4. Terminology Extraction in the LSA space

## 2.4 WikiFactory: A Collaborative Environment for Knowledge Representation, Maintenance and Upgrading

As a final step of our workflow, we used the obtained ontology and its associated knowledge base in order to build a web portal based on a semantic wiki platform. To this aim, we exploited WikiFactory [4]. WikiFactory is a server application that takes as input an OWL ontology (including individuals and facts) and automatically deploys a semantic wiki-based portal. Wiki applications share the same basic philosophy of open editing and provide a simple text-based syntax for content editing. Currently, the most popular semantic wiki is Semantic MediaWiki [12] (a MediaWiki [6] extension), which enables semantic features such as defining categories, relations and articles, which corresponds to OWL classes, properties, and individuals, respectively. Semantic MediaWiki allows

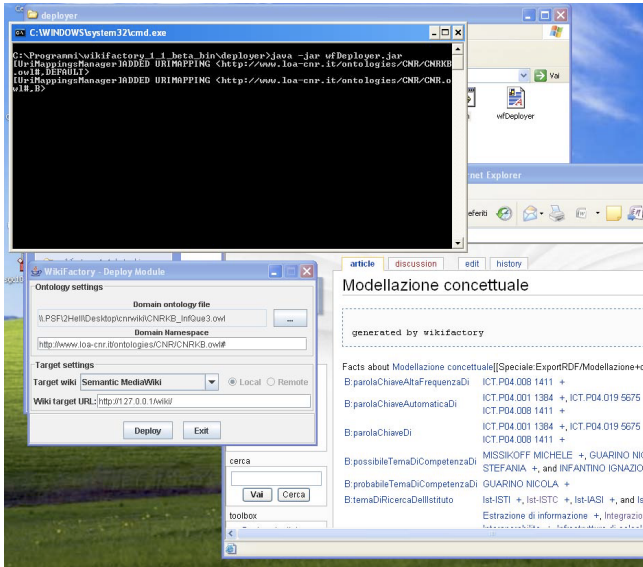


Fig. 5. Deployment on WikiFactory

users to define queries by supporting a subset of SPARQL [7]. Although Semantic MediaWiki could be used in order to import our ontology and associated knowledge base into a semantic wiki site, we decided to use WikiFactory, because it provides additional features that are key to our case study requirements:

- It maintains the synchronization between the underlying ontology and the wiki content: this means that users can navigate and evaluate the ontology, and directly modify it from the wiki pages.
- For each wiki page, WikiFactory provides users with suggestions on the usage of semantic relations: users are not supposed to know all defined relations and the way to apply them, they can rely on the “light” reasoning capability of WikiFactory that include in each page all applicable semantic relations.
- It enables users to handle simple restrictions: each suggested relations is associated with a link that enables users to express some restriction on that relation.

For the sake of our case study, WikiFactory synchronization capability and user support for applicable relations has been particularly useful. Figure 6 shows an example of the visualization made possible by the wiki deployment.

## 2.5 A Remark on Related Work

It’s not our intention to suggest here best practices on how to extract legacy data from relational databases: there exists a large literature on this topic, even in the semantic web proper (see e.g. [9], which also contains a large review of related work). Our focus is here on what to do with the extracted legacy data, and how to use NLP, semantic

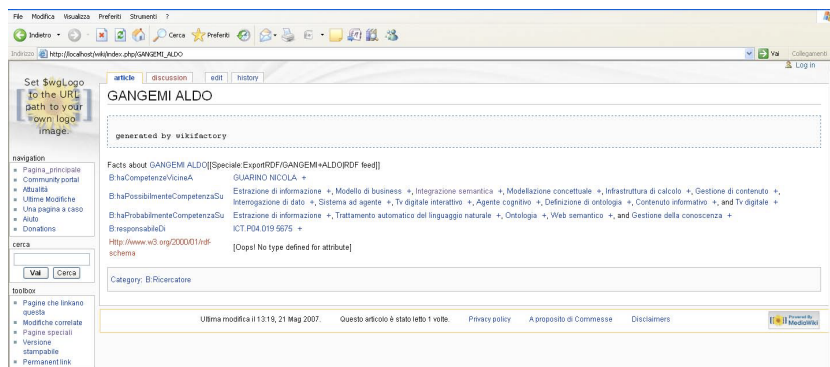


Fig. 6. Visualization of an instance of the class researcher on the wiki

reasoning, and collaborative semantic tools to improve and enrich those data. Within our scope, there is related research, which differentiates from ours in scope and techniques used. For example, [5] focuses on using semantics for expert discovery, but does not apply it to a large organizational intraweb. [1] focuses on using a semantic wiki for organizational knowledge, but does not deal with knowledge enrichment as we do.

### 3 Evaluation

The evaluation of such a complex system is not trivial. Generally speaking, a conclusive judgment on the usefulness of the proposed technology could only be asserted on the basis of user satisfaction on the real use of the front end (the semantic wiki in our case). At the present stage, our application is entering the community usage, and we have formal plans on how to perform a user study in the next months.

For the sake of this work, we have evaluated the different modules independently, by adopting both quantitative and qualitative criteria. In particular, we have evaluated the accuracy of the keyword extraction system, and some functionalities, such as expert finding, and we can express a qualitative judgment on them.

#### 3.1 Benefits from Semantic Migration

Semantic migration of legacy data augments sensibly the range of possibilities of the original information system, since turning it to a knowledge base enables consistency checking, inferences and semantic queries. On the other hand, the pure migration strategy adopted to populate the ontology provides little more than the original material, since only weak inferences can be performed, i.e. inverse relations that are materialized based on the ontology properties, and some SPARQL CONSTRUCT materializations that can e.g. be carried out to associate research programmes to manually inserted topics, and these to the chief scientists leading the programmes. This is an example of CONSTRUCT queries executed on the legacy knowledge alone (the original Italian vocabulary is translated here). The first query constructs owl:PropertyValues between

researchers and the topics they probably have competence on (since they are responsible for the programme for which that topic has been manually inserted). The second constructs owl:PropertyValues that assert the similarity of competence between any two researchers that have probably competence on at least two common topics.

```
CONSTRUCT { ?r hasProbablyCompetenceOn ?k }
WHERE {
  ?r responsibleFor ?c .
  ?c hasManuallyInsertedTopic ?k .}

CONSTRUCT { ?r1 hasSimilarCompetenceAs ?r2 }
WHERE {
  ?r1 hasProbablyCompetenceOn ?k1 .
  ?r2 hasProbablyCompetenceOn ?k1 .
  ?r1 hasProbablyCompetenceOn ?k2 .
  ?r2 hasProbablyCompetenceOn ?k2 .
  FILTER (?k1 != ?k2) .
  FILTER (?r1 != ?r2) .}
```

Unfortunately, manually inserted topics are just a few (see Table 1), and they hardly co-occur in different programmes, so that very few property values have been inferred between researchers and topics, or between similarly competent researchers. This finding suggests that whenever a legacy database mostly contain string-based, non-structured data, migrating them to semantic technologies does not necessarily starts a virtuous circle of knowledge enrichment.

### 3.2 Key-Phrase Extraction

Extracting terminology from domain specific texts is a very well assessed technique in Natural Language Processing [11], and the present state-of-the-art algorithms for this task are highly accurate, achieving precision in general higher than 0.8. The key-phrase extraction problem is more complex, as it requires to associate relevant terms (i.e. keywords) to documents. For this purpose, we have randomly sampled a set of 30 instances from the class `ResearchProgramme`, and we asked a domain expert to manually create a “Gold Standard” set of `ON_TOPIC` associations connecting them to appropriate keywords [5]. The result is a list of keywords associated to each programme, as illustrated in Figure 2. Then we compared the Gold Standard annotations with those provided by the automatic system, measuring (micro)precision and recall, obtained as the total number of (manually assigned) relations that have been actually matched by the automatic system respectively divided by the total number of assignments made by the system, and by the human, obtaining precision 0.66 and recall 0.51. These results are very encouraging, since the agreement measured on the same task is around 70%. In addition, our system is totally unsupervised and can be easily ported to different tasks and domains, making our technology for extracting `on_topic` relations widely applicable at very low costs.

<sup>5</sup> We concentrated on the class `ResearchProgramme` since a lot of text is usually contained in the `ABSTRACT` and `GOALS` fields of each individual, as illustrated by Figure 2.

ID COMM.	KEYWORD 1	KEYWORD 2	KEYWORD 3	KEYWORD 4	KEYWORD 5
938	DIVERSITA' GENETICA	RESISTENZA A STRESS	ALBERO FORESTALE	MARCATORE MOLECOLARE NEUTRALE	CARATTERE ADATTIVO
973	FONDALE MARINO	RILIEVO ACUSTICO	MACCHINA OPERATRICE	PERCEZIONE SONORA	CRESCITA DI FILM SOTTILE
999	DIFETTIVA EUROPEA	QUADRO LEGISLATIVO	TARGET DI QUALITA'	QUALITA' DELL'ARIA	LEGISLAZIONE ITALIANA
1004	ECOSISTEMI	POPOLAZIONE ITTICA	DISPOSITIVO ELETTROACUSTICO	ATTIVITA' DI DIVULGAZIONE	COLONNA D'ACQUA
1054	ESTRAZIONE DI PARAMETRI	REALIZZAZIONE INFRASTRUTTURA	GRIGLIA COMPUTAZIONALE	TELERILEVAMENTO	ELEBORAZIONE DI IMMAGINI
1066	SONDE FLUORESCENTI	PRODUZIONE DI ANTICORPI	CARATTERIZZAZIONE BIOCHIMICA	KIT DIAGNOSTICI	SALUTE UMANA
1085	RILASCIO DI GLUTAMMATO	MUTAZIONE PUNTIFORME	CELLULE GLIALI	MORBO DI PARKINSON	NEURONI DOPAMINERGICI

Fig. 7. Gold Standard Keyphrases associated to different programmes

### 3.3 Reasoning on the Acquired Knowledge

Going back to the sample CONSTRUCT queries shown in Section 3.1, once many more topics are available and are associated to the research programmes described by the texts from which the topics have been extracted, the result of those queries becomes meaningful. In fact, since all research programmes are now associated with many topics, most researchers can be now said to be associated with several topics, as inferrable from a new version of the first CONSTRUCT query, which populates the `hasProbablyCompetenceOn` property values through the statistically inferred topics:

```
CONSTRUCT { ?r hasProbablyCompetenceOn ?k }
WHERE {
?r responsibleFor ?c .
?c hasHighFrequencyStatisticallyInferredTopic ?k .}
```

Since researchers have now more topics, which they can be competent upon, researchers can be associated on a richer competence similarity basis, thus originating a potential social network, needed to achieve the expert finding task. For example, by firing a revision of the second CONSTRUCT query:

```
CONSTRUCT { ?r1 hasSimilarCompetenceAs ?r2 }
WHERE {
?r1 hasProbablyCompetenceOn ?k1 .
?r2 hasProbablyCompetenceOn ?k1 .
?r1 hasProbablyCompetenceOn ?k2 .
?r2 hasProbablyCompetenceOn ?k2 .
FILTER (?k1 != ?k2) .
FILTER (?r1 != ?r2) .}
```

As an example, one of the authors has discovered his own social competence network, made of Aldo Gangemi, Nicola Guarino, Michele Missikoff, Mario Mango Furnari, who are actually the chief scientists that work on semantic web and semantic integration at CNR. The measure of the population performed is found in the third



column of Table 1: from 499 to 3148 topics, and from 14695 to 158441 property values. The precision of the results, based on the first evaluation sessions on a sampling of topics that are closely related to authors' competence, is very good.

## 4 Conclusion and Future Work

In this paper we have described the application of semantic web technologies to the problem of enhancing knowledge extracted from organizational legacy systems. We integrated the information from different databases that describe the scientific organization of CNR into an OWL ontology, and enriched the resulting knowledge base by automatically learning on\_topic relations based on the analysis of the textual fields from the databases. Finally, we performed reasoning on the top of the learnt knowledge, largely expanding the ontology by inferring new relations through the materialization of SPARQL queries. In particular, we concentrated on the expert finding problem, by inducing competence proximity relations between researchers.

The resulting knowledge base has been deployed in a collaborative environment, generated by the WikiFactory tool, allowing communities of users to access the knowledge base and to modify the information there contained, for example by cleaning data, correcting wrong property values inferred on top of automatically learnt topics from the key-phrase extraction system, by adding new instances and relations among them, etc.

This paper has focused on the first stage of the *competence finder* project at Italian National Research Council, a research programme involving several departments and research units. As future work, we will further study the integration of topicality information into structured knowledge bases, for example by formalizing continuous distances within the ontology as a way to reason over semantic proximity. We are also going deeper in integrating text processing and ontologies, for example by applying named entity recognition and text categorization tools on the textual fields. Another parallel development direction is to make user studies on the collaborative environments for annotating and validating the results of automatic text processing techniques. To this aim, we are further developing the WikiFactory platform, concentrating on its compatibility with more sophisticated OWL constructs. Finally, we are going to propose the overall architecture presented in this paper as a general industrial solution for knowledge management, being the verticalization effort limited to the engineering of new domain ontologies, and leaving the remaining components as domain independent.

## Acknowledgments

We are grateful to the members of the NeOn consortium who contributed to the NeOn vision being funded by the European Commission 6th IST Framework Programme. Further information on NeOn is available on <http://www.neon-project.org>.

## References

1. Auer, S., Riechert, T., Dietzold, S.: Ontowiki - a tool for social, semantic collaboration. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)



2. Dagan, I.: Contextual word similarity. In: Dale, R., Moisl, H., Somers, H. (eds.) *Handbook of Natural Language Processing*, pp. 459–476. ch. 19. Marcel Dekker Inc. (2000)
3. Gliozzo, A.: *Semantic Domains in Computational Linguistics*. PhD thesis, University of Trento (2005)
4. Iorio, A.D., Presutti, V., Vitali, F.: Wikifactory: An ontology-based application for creating domain-oriented wikis. In: Sure, Y., Domingue, J. (eds.) *ESWC 2006*. LNCS, vol. 4011, pp. 664–678. Springer, Heidelberg (2006)
5. Kraines, S., Guo, W., Kemper, B., Nakamura, Y.: Semantic web technology for expert knowledge sharing and discovery. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006*. LNCS, vol. 4273, Springer, Heidelberg (2006)
6. The MediaWiki Website, Viewed on (November 30, 2006)  
<http://www.mediawiki.org/wiki/MediaWiki>
7. Eric Prud'hommeaux and Andy Seaborne. SPARQL query language for RDF (2005)
8. Salton, G., McGill, M.H.: *Introduction to modern information retrieval*. McGraw-Hill, New York (1983)
9. Tang, J., Zhou, C., Chen, H., Wang, Y.: From legacy relational databases to the semantic web: an in-use application for traditional chinese medicine. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006*. LNCS, vol. 4273, Springer, Heidelberg (2006)
10. Turney, P.: Learning algorithms for keyphrase extraction. *Information Retrieval* 2(4), 303–336 (2000)
11. Velardi, P., Navigli, R., Cucchiarelli, A., Neri, F.: *Ontology Learning from Text: Methods, Evaluation and Applications*. In: *Evaluation of OntoLearn, a Methodology for Automatic Learning of Domain Ontologies*, IOS Press, Amsterdam (2005)
12. WikiMedia. “Semantic MediaWiki”, Viewed on (November 30, 2006)  
<http://meta.wikimedia.org/wiki/SemanticMediaWiki>

# HealthFinland

## —Finnish Health Information on the Semantic Web

Eero Hyvönen, Kim Viljanen, and Osma Suominen

Semantic Computing Research Group (SeCo),  
Helsinki University of Technology (TKK), Laboratory of Media Technology  
University of Helsinki, Department of Computer Science  
firstname.lastname@tkk.fi  
<http://www.seco.tkk.fi/>

**Abstract.** This paper shows how semantic web techniques can be applied to solving problems of distributed content creation, discovery, linking, aggregation, and reuse in health information portals, both from end-users' and content publishers' viewpoints. As a case study, the national semantic health portal HEALTHFINLAND is presented. It provides citizens with intelligent searching and browsing services to reliable and up-to-date health information created by various health organizations in Finland. The system is based on a shared semantic metadata schema, ontologies, and ontology services. The content includes metadata about thousands of web documents such as web pages, articles, reports, campaign information, news, services, and other information related to health.

## 1 Introduction

Health information on the web is provided by different independent organizations of varying levels of trustworthiness, is targeted to both laymen and experts, is available in various forms, and is written in different languages. The difficulty of finding relevant and trustworthy information in this kind of heterogeneous environment creates an obstacle for citizens concerned about their health. Portals try to ease these problems by collecting content into a single site [1]. Portal types include *service portals* collecting a large set of services together into a localized miniature version of the web (e.g., Yahoo! and other “start pages”), *community portals* [2] acting as a virtual meeting place of a community, and *information portals* [3] acting as hubs of data. This paper discusses problems concerning information portals when publishing health information on the web for the citizens. We consider both the publishers' and the end-users' viewpoints. A distributed semantic web content publishing model has been developed for health organizations, based on a shared metadata schema, ontologies, and ontology services, by which the content is created cost-effectively by independent content producers at different locations. Our system aggregates the content and makes it semantically interoperable to be reused in different applications without modifying it.

To test and demonstrate the approach, we have created an operational prototype of the national semantic health information portal “HEALTHFINLAND—Finnish Health

---

<sup>1</sup> <http://www.w3.org/2001/SW/>

Information of the Semantic Web<sup>2</sup>. The content for the prototype (ca. 6000 web documents) was created by the National Public Health Institute (KTL)<sup>3</sup>, the UKK Institute<sup>4</sup>, the Finnish Institute of Occupational Health<sup>5</sup>, the national Suomi.fi citizen's portal<sup>6</sup>, and the Ministry of Justice<sup>7</sup>, and new organizations are joining in.

In the following, problems of finding and producing health information on the web are first outlined. After this the content creation model of HEALTHFINLAND and the portal itself are presented.

## 2 Problems of Mediating Health Information

A citizen searching for health information on the web faces many challenges:

1. *Content discovery.* The discovery of relevant content is difficult because it often requires prior knowledge of the administrative organization providing the contents.
2. *Outdated and missing linkage.* After finding a piece of interesting information, it is often tedious and difficult to find related relevant web resources. Furthermore, when useful links are given on a web page, they outdate quickly. When new information is entered in a site or old information changed or removed, the links in existing pages cannot be updated automatically but refer to older information, or even non-existing information.
3. *Content aggregation.* Satisfying an end-user's information need often requires *aggregation* of content from several information providers, which is difficult if heterogeneous content is provided by several independent web sites. For example, if a baby is born in your family, relevant information related to the situation may be provided by health care organizations, social organizations, the church, legal administration, and others.
4. *Quality of content.* The trustworthiness of the information on the web pages varies. In many cases it is difficult know whether a content is based on scientific results or laymans' opinions and rumors, or whether it is motivated by commercial interests.
5. *Matching end-users' expertise level.* There are lots of medical information available that is targeted to experts rather than ordinary citizens. Providing and finding the information on the right level of user expertise is a challenge that is very evident in the medical domain where, e.g., the terminology used by doctors and content providers is very different from the terminology used by citizens in expressing their needs and interests.

From the viewpoint of the health organizations, creating health information to citizens is problematic in many ways:

---

<sup>2</sup> <http://www.seco.tkk.fi/applications/terveysuomi/>

<sup>3</sup> <http://www.ktl.fi/>

<sup>4</sup> <http://www.ukkinstituutti.fi/>

<sup>5</sup> <http://www.ttl.fi/>

<sup>6</sup> <http://www.suomi.fi/>

<sup>7</sup> <http://www.finlex.fi/>

1. *Duplicated work.* Several organizations create overlapping content, which is in many cases a waste of time and money and confusing to the end-user. For example, in Finland the governmental citizen portal Suomi.fi has a section for governmental health information containing material partly overlapping with those available through the sites of the Finnish Centre of Health Promotion, and the health pages of the national broadcasting company YLE. These organizations share the goal of providing free health information to citizens and are not competing with other. In our vision, similar content should in such situations be created only once and reused rather than recreated by others.
2. *Difficulty of reusing content.* Content in portals is usually annotated for the purpose of presenting it in a particular portal and for the particular purpose of the organization managing the portal. This makes it difficult and expensive for other organizations to reuse content across portals even if the portal owners were willing to do this. For example, in our case, a newspaper would be willing to publish links to the governmental HEALTHFINLAND portal to enrich their health related news articles, and the portal would definitely like to promote its health information to the readers of the online newspaper. However, a cost-effective way to do this with minimal changes in current content management systems (CMS) is needed.
3. *Internal and external link maintenance.* The problems of maintaining links up-to-date is costly and tedious from the site maintenance viewpoint, especially when dealing with links to external sites to which the maintainer has no control.
4. *Indexing (annotation) problems.* Finding the right keywords and other metadata descriptions for web pages and documents is difficult and time consuming for information producers. The vocabularies used, such as MeSH<sup>8</sup>, UMLS<sup>9</sup> or SNOMED CT<sup>10</sup>, are very large and require expertise to use.
5. *Quality control.* There are several quality issues involved when publishing health information: 1) Quality of the content creation process (e.g. regular reviews and updates of published material). 2) Quality of the content itself (e.g., errors in the medical subject matter, is the content readable and written for the correct audience). 3) Quality of additional information on pages (e.g., it is advisable to show the date of publication on each page). 4) Quality of the metadata. For example, one indexer may use only few general keywords while another prefers a longer detailed list, which leads to problems of unbalanced metadata.

Much of the semantic web<sup>11</sup> content will be published using semantic portals<sup>12</sup> based on web standards such as RDF<sup>12</sup> and OWL<sup>13</sup>. In MUSEUMFINLAND<sup>14</sup> <sup>15</sup>, a semantic web model and portal was created in the cultural domain for distributed semantic content creation <sup>16</sup>, aggregation, and provision to end-users using semantic search and browsing services. This approach has been shown to be applicable in different domains

<sup>8</sup> <http://www.nlm.nih.gov/mesh/>

<sup>9</sup> <http://umlsinfo.nlm.nih.gov>

<sup>10</sup> <http://www.snomed.org/snomedct/>

<sup>11</sup> <http://www.w3.org/2001/sw/>

<sup>12</sup> <http://www.w3.org/RDF/>

<sup>13</sup> <http://www.w3.org/TR/owl-features/>

<sup>14</sup> <http://www.museosuomi.fi/>

[17], and it was also applied to HEALTHFINLAND. In the following, we show how HEALTHFINLAND develops the idea of semantic portals further and applies it in practice to create a national publication channel for health information targeted to citizens.

### 3 Overview of the HEALTHFINLAND Approach

In traditional web publishing, content creators publish web pages and link them together independently from each other. Content management systems (CMS) and portals are used to aggregate related material collections within one site, and to provide local search and linking services. Linking between sites is usually done manually. Search engines are used to provide content aggregation services on the global cross-site level.

In HEALTHFINLAND we wanted to create a new kind of collaborative distributed content creation model for publishing health information on the web in order to solve the problems listed in section 2.

The first idea of the model is to minimize duplicate redundant work and costs in creating health content on the national level by producing it only once by one organization, and by making it possible to reuse the content in different web applications by the other organizations, not only in the organization's own portal. This possibility is facilitated by annotating the content locally with semantic metadata based on shared ontologies, and by making the global repository available by a semantic portal and by open APIs for creating mash-up web services. This is a generalization of the idea of “multi-channel publication” of XML, where a single syntactic structure can be rendered in different ways, but on the semantic metadata level and using RDF: semantic content is reused through *multi-application publication*.

The second key idea behind HEALTHFINLAND is to try to minimize the maintenance costs of portals by letting the computer take care of semantic link maintenance and aggregation of content from the different publishers. This possibility is also based on shared semantic metadata and ontologies. New content relevant to a topic may be published at any moment by any of the content providers, and the system should be able to put the new piece of information in the right context in the portal, and automatically link it with related information.

The third major idea of HEALTHFINLAND is to provide the end-user with intelligent services for finding the right information based on her own conceptual view of health, and for browsing the contents based on their semantic relations. The views and vocabularies used in the end-user interface may be independent of the content providers' organizational perspective, and are based on layman's vocabulary that is different from the medical expert vocabularies used by the content providers in indexing the content.

Figure 1 depicts an overview of the HEALTHFINLAND system. The content providers on the left produce web pages, documents, and other resources of interest along their organizational interests as before for their own purposes (“primary applications” in the figure). However, the content is annotated by using a shared metadata schema and ontologies for the others to use, too. Selected content is then harvested into a global knowledge base (center of the figure) to be reused in “secondary applications”. In this paper, we focus on one application in particular, the semantic portal HEALTHFINLAND that provides citizens with information services to the global health information repository. We

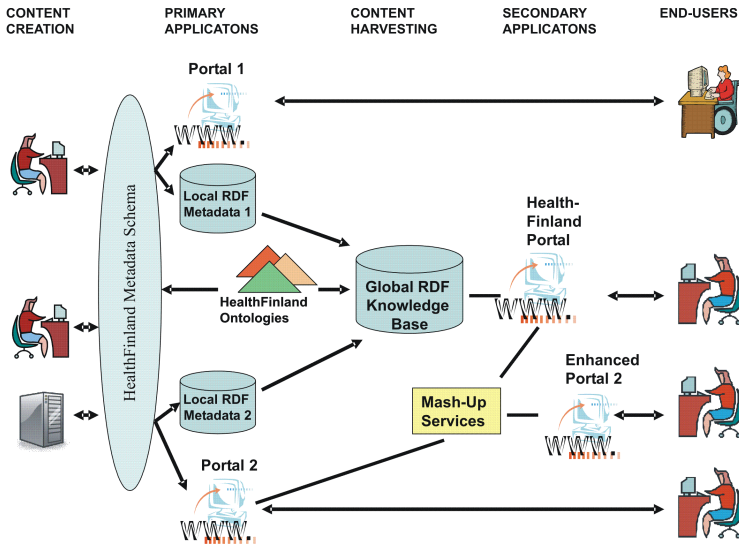


Fig. 1. An overview of the HEALTHFINLAND content creation and reuse process

will also briefly show how external organizations can reuse the semantic content cost-effectively with semantic mash-up service components called “floatlets” in the spirit of Google Maps<sup>15</sup> and AdSense<sup>16</sup>, but generalized on the semantic level. The figure depicts an enhanced portal “Portal 2” in which the content of the primary application is enriched by, e.g., semantic recommendation links to content pages in HEALTHFINLAND. In the following, the metadata schema and ontologies used in the system are first outlined.

## 4 Ontological Infrastructure

The ontological infrastructure of HEALTHFINLAND consists of two major components:

- 1) A metadata schema, i.e. an annotation ontology that specifies what elements are used for describing the web documents to be included in the system, and what kind of values the elements (properties) can take. The metadata schema is shared by all organizations creating the content and ensures *syntactic interoperability* of the content.
- 2) A set of ontological vocabularies whose concepts are used to fill in values of the metadata schema. The ontologies are also shared by the organizations, and their usage ensures *semantic interoperability* of the content.

### 4.1 Metadata Schema

The HEALTHFINLAND portal requires the web documents used in the system to be described in a uniform and machine-understandable manner. A metadata schema specifies

<sup>15</sup> <http://maps.google.com>

<sup>16</sup> <http://www.google.com/adsense/>

a set of fields (properties) which are used for presenting information about each document. The values of the metadata fields are either human-readable text (e.g., title), structured strings (e.g., publication date) or shared, explicitly identified ontological concepts (e.g., the subject classification). Some fields are obligatory and some fields may exist more than once. In addition to being a formal specification of what is required from the content producers, the schema can be used for, e.g., automatically generating a user interface for creating metadata conforming to the schema, and for automatic content validation and feedback generation before publishing the content in the portal [8].

**Table 1.** HEALTHFINLAND Metadata Schema. Mandatory fields are marked in **bold**. Cardinalities are presented in the column C.

	Name	QName	C	Value type	Value range
General metadata	<b>Identifier</b>	<b>dc:identifier</b>	1	URI	
	Locator	ts:url	0..1	URL	
	<b>Title</b>	<b>dc:title</b>	1 <sup>a</sup>	Free text	Non-empty string.
	<b>Abstract</b>	<b>dcterms:abstract</b>	1 <sup>a</sup>	Free text	Non-empty string.
	<b>Language</b>	<b>dc:language</b>	1..*	String	RFC 3066
	<b>Publication time</b>	<b>dcterms:issued</b>	1	String	W3CDTF (ISO 8601)
	Acceptance time	dcterms:dateAccepted	0..*	String	W3CDTF (ISO 8601)
	Modification time	determs:modified	0..*	String	W3CDTF (ISO 8601)
	<b>Publisher</b>	<b>dc:publisher</b>	1..*	Instance	foaf:Organization
Creator	dc:creator	0..*	Instance	foaf:Organization, foaf:Person or foaf:Group	
Content classification	<b>Subject</b>	<b>dc:subject</b>	1..*	Concept	YSO, MeSH and HPMulti Ontologies
	<b>Audience</b>	<b>dcterms:audience</b>	1..*	Concept	Audience Ontology
	<b>Genre</b>	<b>ts:genre</b>	1..*	Concept	Genre Ontology
	<b>Presentation type</b>	<b>dc:type</b>	1..*	Concept	DCMI Type vocabulary
	<b>Format</b>	<b>dc:format</b>	1	String	IANA MIME types
	<b>Medium</b>	<b>dcterms:medium</b>	1	Concept	Medium Ontology
	Spatial coverage	determs:spatial	0..*	String or concept	DCMI Point, DCMI Box or Location Ontology
Relations	Temporal coverage	determs:temporal	0..*	String or concept	W3CDTF, DCMI Period or Time Ontology
	Part of	dcterms:isPartOf	0..*	Document	URI
	Rights	dc:rights	0..*	Free text or document	URI or textual description
	Source	dc:source	0..*	Free text or document	URI (e.g., ISBN) or bibliographical reference
	Reference	dcterms:references	0..*	Free text or document	URI (e.g., ISBN) or bibliographical reference
Translations	Translation of	ts:isTranslationOf	0..*	Document	URI
	Format of	determs:isFormatOf	0..*	Document	URI

<sup>a</sup> Multilingual values are allowed, but only one value in each language.

The metadata schema (Table 1) is based on the Dublin Core Element Set<sup>17</sup>, along with refinements introduced in DCMI Terms<sup>18</sup>. In addition, to allow a more detailed description of the required metadata, we have introduced three extensions to Dublin Core elements: 1) The *dc:type* field has been refined with a *ts:genre* field<sup>19</sup> to distinguish between the technical type of the document (presented using DCMI Type vocabulary) and the content genre, such as *News item*, *Organizational information* and *Research* (described in our Genre ontology). 2) The *dc:identifier* is extended with an (optional) *ts:url*

<sup>17</sup> <http://dublincore.org/documents/dces/>

<sup>18</sup> <http://dublincore.org/documents/dcmi-terms/>

<sup>19</sup> Namespace *ts* refers to the Finnish name TerveSuomi of HEALTHFINLAND.



**Table 2.** Examples of how metadata is presented in RDF/XML and XHTML

	RDF/XML	XHTML
Free text	<code>&lt;dc:title&gt;Rokotteiden hävittäminen&lt;/dc:title&gt;</code>	<code>&lt;meta name="DC.title" content="Rokotteiden hävittäminen" /&gt;</code>
String	<code>&lt;dc:language&gt;&lt;dcterms:RFC3066&gt;&lt;rdf:value&gt;fi&lt;/rdf:value&gt;&lt;/dcterms:RFC3066&gt;&lt;/dc:language&gt;</code>	<code>&lt;meta name="DC.language" scheme="DCTERMS.RFC3066" content="fi" /&gt;</code>
Concept	<code>&lt;dc:subject rdf:resource="http://www.yso.fi/onto/yso/p123" /&gt;</code>	<code>&lt;link rel="DC.subject" href="http://www.yso.fi/onto/yso/p123" /&gt;</code>

field to distinguish between non-accessible identifiers and document locators. 3) The final extension is the *ts:isTranslationOf* field which extends the *dcterms:isVersionOf*, and is used for presenting the relation between language translations of documents. The metadata schema is specified in detail in [9].

The metadata in HEALTHFINLAND is intended to be presented using RDF, conforming to the recommendations for expressing Dublin Core in RDF [10]. A subset of the metadata can also be embedded in (X)HTML pages using META and LINK elements based on the Dublin Core recommendation [11]. The HTML embedded metadata solution has some limitations, because not all relevant documents are in HTML format and advanced RDF metadata structures, such as defining an instance with a certain URI, can not be done using the HTML META and LINK tags. Therefore, the RDF presentation is recommended. Examples of how metadata is expressed in RDF and HTML is shown in Table 2.

The RDF and/or HTML embedded metadata is published for the HEALTHFINLAND portal by making it available on a public WWW server where it can be accessed regularly by the HEALTHFINLAND metadata harvester which fetches the content from the content providers to a centralized metadata repository (cf. Figure 1). During the harvesting, 1) the content is transformed into RDF (if originally presented in HTML), 2) missing values are replaced with default values when possible, and 3) the RDF is validated against the metadata schema and other validation rules. Each metadata producer gets a report of warnings, errors and other problems that were encountered during harvesting and validating the content. If some parts or all of the metadata is unacceptable due to serious errors, the metadata is discarded until necessary corrections are made. Otherwise, the metadata is added to and published in the HEALTHFINLAND portal.

## 4.2 Ontologies

Semantic interoperability in HEALTHFINLAND is obtained by using a set of shared ontologies for filling in the values of the metadata schema. The ontologies include a Medium Ontology containing resources for representing different media types (Web page, CD, DVD, etc.), an Audience Ontology representing categories of people, such as genre groups, professional groups, risk groups, and age groups, a Place Ontology containing geographical places (e.g., Finland, Helsinki, etc.) in a part-of hierarchy, a Genre Ontology for genre types (news, game, etc.), DCMI type ontology containing media types (text, sound, video, etc.), and a Time Ontology. In the future, custom made organizational vocabularies can also be used, provided that they are linked with the HEALTHFINLAND ontologies.



The most important ontologies in HEALTHFINLAND are the three *core subject domain* ontologies that are used for describing the subject matter of web contents:

1. The Finnish General Upper Ontology (YSO)<sup>[20]</sup> that includes approximately 20 000 concepts. The YSO ontology was created by transforming the General Finnish Thesaurus YSA<sup>[21]</sup> into RDF/OWL format using the Protégé editor<sup>[22]</sup> and by manually crafting the concepts into full-blown rdfs:subClassOf hierarchies [12]. YSA is widely used in Finland for indexing various kinds of content, e.g. in libraries.
2. The international Medical Subject Headings (MeSH) which includes approximately 23 000 concepts. The Finnish translation of MeSH, FinMeSH, was developed by the Finnish Medical Society Duodecim<sup>[23]</sup> and was acquired for HEALTHFINLAND as a database. The vocabulary was transformed into the SKOS Core format<sup>[24]</sup> without changing the semantics of the vocabulary or its structure [13].
3. The European Multilingual Thesaurus on Health Promotion<sup>[25]</sup> (HPMULTI), which included a Finnish translation. HPMULTI contains approximately 1200 concepts related specifically to health promotion. HPMULTI was transformed into RDF SKOS in the same way as FinMeSH.

All three ontologies were needed to cover the subject matter of the portal properly. YSO is broad but too general w.r.t. detailed medical content. On the other hand, MeSH contains lots of useful medical concepts and is widely used in the health sector, but is focused on clinical healthcare. HPMULTI complements the two vocabularies by focusing on health promotion terminology.

## 5 Distributed Semantic Content Creation

A major challenge in the distributed content creation model of HEALTHFINLAND is how to facilitate the cost-effective production of descriptive, semantically correct high-quality metadata. In HEALTHFINLAND three ways of creating metadata are considered and supported: 1) Enhancing existing web content management systems (CMS) with ontology mash-up service components for producing semantic metadata. 2) Using a browser-based metadata editor for annotating web content. 3) Automatic conversion of metadata. These approaches are outlined below.

### 5.1 Enhancing an Existing CMS with Ontology Services

Most content providers in HEALTHFINLAND use a CMS for authoring, publishing and archiving content on their website. A typical CMS supports creation of textual metadata about documents, such as title and publication time, but not ontological annotations. This would require that the system has functionalities supporting ontology-based

<sup>20</sup> <http://www.seco.tkk.fi/ontologies/ys/>

<sup>21</sup> <http://www.vesa.lib.helsinki.fi>

<sup>22</sup> <http://protege.stanford.edu>

<sup>23</sup> <http://www.duodecim.fi>

<sup>24</sup> <http://www.w3.org/2004/02/skos/core/>

<sup>25</sup> <http://www.hpmulti.net/>

annotation work, e.g., concept search for finding the relevant concepts (identified with URIs), concept visualization for showing the concept to the user, and storing concepts along other information about the documents. The CMS should also be able to export the metadata, preferably in RDF format, to be used by semantic web applications.

Currently, ontologies are typically shared by downloading them, and each application must separately implement the ontology support. To avoid duplicated work and costs, and to ensure that the ontologies are always up-to-date, we argue that one should not only share the ontologies, but also the *functionalities* for using them by centralized mash-up component services. Such services, e.g. Google Maps, have been found very useful and cost-effective in Web 2.0 applications for integrating new functionalities with existing systems.

We have applied the idea of using mash-ups to provide ontology services for the content producers of HEALTHFINLAND by creating the ONKI Ontology Server framework<sup>[24]</sup> [14]. ONKI provides ontological functionalities, such as concept searching, browsing, disambiguation, and fetching, as ready-to-use mash-up components that communicate asynchronously by AJAX<sup>[27]</sup> (or Web Service technologies) with the shared ontology server. The service integration can be easily done by changing only slightly the user-interface component at the client side. For example, in the case of AJAX and HTML pages, only a short snippet of JavaScript code must be added to the web page for using the ONKI services.

The main functionality addressed by the ONKI UI components is concept finding and fetching. For finding a desired annotation concept, ONKI provides text search with semantic autocompletion [15]. This means that when the annotator is typing in a string, say in an HTML input field of a CMS system, the system dynamically responds after each input character by showing the matching concepts on the ONKI server. By selecting a concept from the result list, the concept's URI, label or other information is fetched to the client application.

Concept browsing can also be used for concept fetching. In this model, the user pushes a button on the client application that opens a separate ONKI Browser window in which annotation concepts can be searched for and browsed. For each concept entry, the browser shows a *Fetch concept* button which, when pressed, transfers the current concept information to the client application. ONKI supports multilingual ontologies, has a multilingual user-interface, supports loading multiple ontologies, and can be configured extensively.

ONKI is implemented as a Java Servlet application running on Apache Tomcat<sup>[28]</sup>. It uses the Jena semantic web framework<sup>[29]</sup> for handling RDF content, the Direct Web Remoting (DWR)<sup>[30]</sup> library for implementing the AJAX functionalities, the Dojo JavaScript toolkit<sup>[31]</sup>, and the Lucene<sup>[32]</sup> text search engine.

<sup>26</sup> <http://www.seco.tkk.fi/services/onki/>

<sup>27</sup> [http://en.wikipedia.org/wiki/Ajax\\_\(programming\)](http://en.wikipedia.org/wiki/Ajax_(programming))

<sup>28</sup> <http://tomcat.apache.org/>

<sup>29</sup> <http://jena.sourceforge.net/>

<sup>30</sup> [http://en.wikipedia.org/wiki/DWR\\_\(Java\)](http://en.wikipedia.org/wiki/DWR_(Java))

<sup>31</sup> <http://dojotoolkit.org/>

<sup>32</sup> <http://lucene.apache.org/>

## 5.2 Browser-Based Metadata Editor

Some HEALTHFINLAND content providers can not add mash-up ontology support to their CMS due to technical or economical reasons. Furthermore, some content providers do not even have a CMS or they may not have access to the CMS that contains the content, e.g., if the content originates from a third party. To support metadata productions in these cases, we have created a centralized browser-based annotation editor SAHA [8] for annotating web pages. SAHA adapts automatically to different metadata schemas. In this case the HEALTHFINLAND schema is used. The schema element fields in SAHA can be connected with ONKI ontology service components, providing concept finding and fetching services to the annotator, as discussed above.

## 5.3 Automatic Conversion

The third content producing method in HEALTHFINLAND is automatic conversion of original data to HEALTHFINLAND metadata. This method is currently used in cases where metadata exists in a CMS, but it is in an incompatible format, does not contain ontological annotations (URIs) and/or some minor information is missing in the metadata. Because the HEALTHFINLAND metadata schema is strongly based on Dublin Core and because many content providers in Finland use thesauri (e.g., the Finnish General Thesaurus YSA and the Medical Subject Headings MeSH), the content can in many cases be transformed fairly accurately into ontological form. For example, some juridic content produced by the Finnish Ministry of Justice is harvested for HEALTHFINLAND. The metadata, targeted originally for the governmental Suomi.fi portal, uses a Dublin Core based metadata schema (JHS 143 recommendation [16]) and is automatically translated into the HEALTHFINLAND metadata format.

# 6 Intelligent Services to the End-User

The HEALTHFINLAND user interface is based on the faceted browsing (a.k.a. view-based search) paradigm [17][18], which has been found useful in our earlier semantic portals, such as [5][17], and in other systems, such as SWED<sup>33</sup> and Multimedien<sup>34</sup>.

A challenge in publishing health-related information in a citizens' semantic portal is the gap between the citizens' information needs and the professional conceptualizations and terminology used in medical vocabularies. To bridge this gap and to enable an intuitive facet-based user interface for the portal, we constructed the search facets by using a card sorting method [19] to elicitate how users tacitly group and organize concepts in the health domain. The new user-centric facets organize the material from citizens' point of view, and they are mapped by the portal to concepts in the medical ontologies.

The HEALTHFINLAND portal, like typical semantic portals, provides the end-user with two basic services: 1) a search engine based on the semantics of the content and 2) dynamic linking between pages based on the semantic relations in the underlying knowledge base. The main facets of the portal are Topic, Life event, Group of people,

<sup>33</sup> <http://www.swed.org.uk/swed/index.html>

<sup>34</sup> <http://e-culture.multimedien.nl/demo/search>

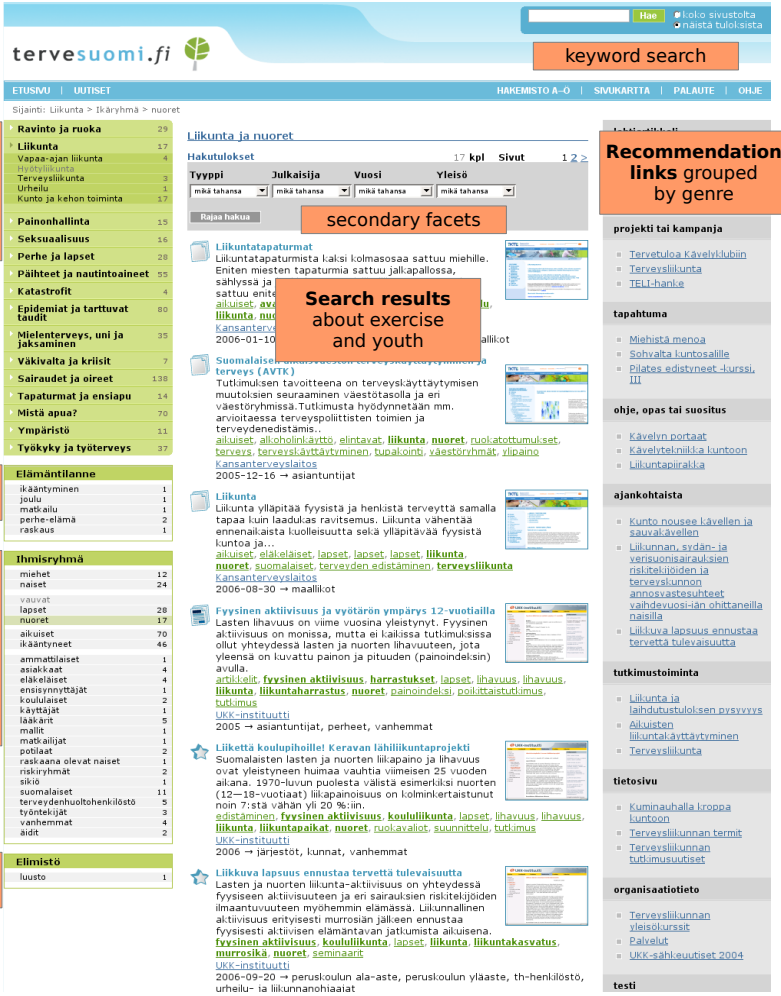


Fig. 2. Portal user interface with semantic search, browsing and recommendations

and Body part. The facets can be seen in the left column in Figure 2. In addition, secondary drop-down facets for constraining the search with a set of additional choices are provided for Genre, Publisher, Publication year and Audience.

Keyword searches can be initiated at any time and can be combined with category browsing. Traditional keyword search functionality has been semantically enhanced by targeting not only content titles, descriptions and body text but also the facet categories and underlying ontology concepts, including non-preferred concept labels. Thus, synonyms and abbreviations can be used in keyword searches provided they are known in the ontology.

The portal also provides recommendation links at several stages: 1) individual content items (pages) are linked to related material, 2) search result listings provide “best

picks”, and 3) concept pages link to related content. Recommendations are generated using ontological knowledge and are grouped according to genre (e.g. statistics, research activities, news items, laws) or language (e.g. similar content in English).

One problem with a portal approach and distributed content creation in general is that when search results are provided as traditional hyperlinks, users are forced to navigate between different web sites that each have their own navigation systems and styling. Also, providing recommendation links across sites is challenging. The HEALTHFINLAND portal will integrate selected content items that have been retrieved from affiliated websites directly into the portal interface, providing seamless navigation and recommendation links in the proper context of the content page. Our solution requires that the content is marked up using a small amount of RDFa syntax<sup>35</sup>, which helps the metadata harvester extract the body content of suitable web pages, skipping navigation elements and styling.

The HEALTHFINLAND portal also incorporates an alphabetical index of concepts as well as a concept browser that can be used to browse the subject ontology and for concept-based search of the content.

The portal is implemented as a Java Servlet application running on Apache Tomcat. It is built using the Tapestry framework<sup>36</sup> and uses Jena for RDF functionality. Search and recommendation functionality has been implemented using the Lucene search engine, which has been enhanced to handle category and concept queries.

## 7 Discussion

This paper addressed the problems of the citizen end-users (cf. Section 2) as follows: 1) Content finding is supported by cross-portal semantic search, based on concepts and facets rather than keywords. 2) The problem of outdated and missing links is eased by providing the end-user with semantic recommendations that change dynamically as content is modified. 3) Content aggregation is facilitated by end-user facets that collect distributed but related information from different primary sources. 4) Quality of content is maintained by including only trustworthy organizations as content producers. 5) End-users’ expertise level is taken into account by the metadata element “Audience”. Separation of end-user vocabularies from indexing vocabularies makes it possible for the citizen to search and browse content using layman’s vocabulary although the content is indexed by using professional medical terminology.

At the same time, the problems of content providers (cf. Section 2) are eased, too: 1) Duplication of content creation can be minimized by the possibility of aggregating cross-portal content. 2) Reusing the global content repository is feasible, as demonstrated by the semantic portal HEALTHFINLAND. By using mash-up techniques, external applications, such as the primary applications of Figure 1, can reuse the content provided by secondary applications, such as HEALTHFINLAND. 3) Internal and external link management problems are eased by the dynamic semantic recommendation system of the portal and the content aggregation mechanisms. 4) The tedious content indexing task is supported cost-effectively by shared ontology service components for

<sup>35</sup> <http://www.w3.org/TR/xhtml-rdfa-primer/>

<sup>36</sup> <http://tapestry.apache.org/>

mash-ups. 5) Metadata quality can be enhanced by providing indexers with ontology services by which appropriate indexing concepts can be found and correctly entered into the system.

The content creation model presented is based on a shared metadata schema and ontologies as in [20]. However, the idea of sharing ontologies through ontology service components for mash-ups is new. The user interface is based on the faceted search paradigm [17][18], but integrated with semantic web ontologies and reasoning with semantic recommendations [21], as in [22]. A new feature of the system is the separation of end-user facets from indexing ontologies [23][19], which is crucial in the medical domain. The card sorting approach [19] was found useful in accomplishing this.

This work is a part of the national semantic web ontology project FinnONTO<sup>37</sup> 2003–2007, funded mainly by the National Funding Agency for Technology Innovation (Tekes) and the Ministry of Social Affairs and Health. The HEALTHFINLAND project is co-ordinated by the National Health Institute in Finland. We thank E. Hukka, M. Holi, P. Lindgren, and J. Eerola for co-operation.

## References

1. Sidoroff, T., Hyvönen, E.: Semantic e-government portals - a case study. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
2. Staab, S., et al.: Semantic Community Web Portals. In: Proceedings of the 9th International World Wide Web Conference, Amsterdam, The Netherlands, Elsevier (2000)
3. Reynolds, D., Shabajee, P., Cayzer, S.: Semantic Information Portals. In: Proceedings of the 13th International World Wide Web Conference on Alternate track papers & posters, ACM Press, New York (2004)
4. Maedche, A., Staab, S., Stojanovic, N., Struder, R., Sure, Y.: SEMantic portaAL — the SEAL approach. Technical report, Institute AIFB, University of Karlsruhe, Germany (2001)
5. Hyvönen, E., Mäkela, E., Salminen, M., Valo, A., Viljanen, K., Saarela, S., Junnila, M., Kettula, S.: MuseumFinland. MuseumFinland – Finnish museums on the semantic web 3(2), 224–241 (2005)
6. Hyvönen, E., Saarela, S., Viljanen, K., Mäkela, E., Valo, A., Salminen, M., Kettula, S., Junnila, M.: A semantic portal for publishing museum collections on the web. In: Proceedings of ECAI/PAIS 2004, Valencia, Spain (2004)
7. Käsää, T., Hyvönen, E.: A semantic view-based portal utilizing Learning Object Metadata. In: Mizoguchi, R., Shi, Z., Giunchiglia, F. (eds.) ASWC 2006. LNCS, vol. 4185, Springer, Heidelberg (2006)
8. Valkeapää, O., Alm, O., Hyvönen, E.: Efficient content creation on the semantic web using metadata schemas with domain ontology services (system description). In: Proceedings of the ESWC 2007, Springer, Berlin (2007)
9. Suominen, O., Viljanen, K., Hyvönen, E., Holi, M., Lindgren, P.: TerveSuomi.fi:n metatietomäärittely (Metadata schema for TerveSuomi.fi), Ver. 1.0 (26.1.2007) (2007), <http://www.seco.tkk.fi/publications/>
10. Dublin Core Workgroup: Expressing qualified Dublin Core in RDF/XML (2002), <http://dublincore.org/documents/dcq-rdf-xml/>

<sup>37</sup> <http://www.seco.tkk.fi/projects/finnonto/>

11. Dublin Core Workgroup: Expressing Dublin Core in HTML/XHTML meta and link elements(2003), <http://dublincore.org/documents/dcq-html/>
12. Hyvönen, E., Valo, A., Komulainen, V., Seppälä, K., Kauppinen, T., Ruotsalo, T., Salminen, M., Ylisalmi, A.: Finnish national ontologies for the semantic web—towards a content and service infrastructure. In: DC 2005. Proceedings of International Conference on Dublin Core and Metadata Applications (2005)
13. van Assem, M., Malaise, V., Miles, A., Schreiber, G.: A method to convert thesauri to skos. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)
14. Hyvönen, E., Viljanen, K., Mäkelä, E., et al.: Elements of a national semantic content infrastructure—case Finland on the semantic web. In: Proceedings of the ICSC 2007, IEEE Computer Society Press, Los Alamitos (2007)
15. Hyvönen, E., Mäkelä, E.: Semantic autocompletion. In: Mizoguchi, R., Shi, Z., Giunchiglia, F. (eds.) ASWC 2006. LNCS, vol. 4185, Springer, Heidelberg (2006)
16. JHS workgroup: JHS 143: Asiakirjojen kuvailun ja hallinnan metatiedot (2004), <http://www.jhs-suositukset.fi/suomi/jhs143>
17. Pollitt, A.S.: The key role of classification and indexing in view-based searching. Technical report, University of Huddersfield, UK (1998), <http://www.ifla.org/IV/ifla63/63polst.pdf>
18. Hearst, M., Elliott, A., English, J., Sinha, R., Swearingen, K., Lee, K.P.: Finding the flow in web site search. CACM 45(9), 42–49 (2002)
19. Suominen, O., Viljanen, K., Hyvönen, E.: User-centric faceted search for semantic portals. In: Proceedings of the ESWC 2007, Springer, Berlin (2007)
20. Hyvönen, E., Salminen, M., Kettula, S., Junnila, M.: A content creation process for the Semantic Web, Proceeding of OntoLex 2004: Ontologies and Lexical Resources in Distributed Environments, May 29, Lisbon, Portugal (2004)
21. Viljanen, K., Käsälä, T., Hyvönen, E., Mäkelä, E.: Ontodella—a projection and linking service for semantic web applications. In: Bressan, S., Küng, J., Wagner, R. (eds.) DEXA 2006. LNCS, vol. 4080, Springer, Heidelberg (2006)
22. Hyvönen, E., Saarela, S., Viljanen, K.: Application of ontology based techniques to view-based semantic search and browsing. In: Proceedings of the First European Semantic Web Symposium, Heraklion, Greece, Springer, Berlin (2004)
23. Holí, M., Hyvönen, E.: Fuzzy view-based semantic search. In: Mizoguchi, R., Shi, Z., Giunchiglia, F. (eds.) ASWC 2006. LNCS, vol. 4185, Springer, Heidelberg (2006)



# Spatially-Augmented Knowledgebase

Dave Kolas and Troy Self

BBN Technologies  
1300 N. 17<sup>th</sup> St., Suite 400, Arlington, VA 22209  
{dkolas, tself}@bbn.com

**Abstract.** As an increasing number of applications on the web contain some elements of spatial data, there is a need to efficiently integrate Semantic Web technologies and spatial data processing. This paper describes a prototype system for storing spatial data and Semantic Web data together in a SPatially-AUGmented Knowledgebase (SPAUK) without sacrificing query efficiency. The goals are motivated through use several use cases. The prototype's design and architecture are described, and resulting performance improvements are discussed.

## 1 Introduction

With the advent of social networking sites, wikis, and other web environments that fall under the umbrella of web collaboration technologies, exposing the data behind web sites in machine-readable formats is becoming ever more popular. Much of the information linked and shared across the Web becomes more useful when combined with its spatial context. Crime statistics, real-estate information, and restaurant reviews are examples of information that is more useful when consumed from a spatial perspective. Using Web 2.0 techniques, web sites commonly referred to as “mash-up” sites are able to display information spatially. For example, one site may overlay crime statistics on a map using Google Maps<sup>1</sup> while another site displays houses for sale on Google Maps. In both cases, the combination of data and capabilities is predefined by the mash-up site and is only used for display purposes.

Semantic Web technologies, such as the Resource Description Framework (RDF) and the SPARQL Protocol and RDF Query Language (SPARQL) are beginning to eliminate this limitation. The graph structure of RDF along with the graph query capabilities of SPARQL make them ideal candidates for representing and searching the ever-changing, interlinked, flexible data of the Web, which is not easily done using a traditional relational database [1].

RDF databases, sometimes called triplestores, offer significant advantages over traditional structured databases for Semantic Web data [2], but are not optimized for spatial information such as geographic coordinates. In this paper, we describe a Spatially AUGmented Knowledgebase (SPAUK) that provides the high-performance graph query capabilities needed for searching webs of data, without sacrificing the spatial indexing and processing capabilities necessary for performing searches

---

<sup>1</sup> <http://maps.google.com>



involving spatial extents and operators. Here we describe our motivations and example use cases for the augmented knowledge base as well as the design and implementation results. Finally, we discuss the status of the prototype and future direction.

## 2 Motivation

While RDF and modern triple stores are efficient at storing and querying data linked across multiple sources of information, they are poor performers when it comes to spatial processing. The current standard for storing spatial data generally involves using an object-relational database augmented with spatial capabilities, such as Oracle Spatial. While this approach has proven effective within a predominantly spatial environment, the object-relational model lacks the flexibility of RDF and triple-stores that make them attractive for searching linked data across multiple sources. The goal of SPAUK here is to provide efficient storage and query of spatial data without sacrificing the flexibility and graph search ability of RDF and triplestores.

### 2.1 Use Cases

#### Query Mash-ups

Online communities, specifically social networking sites, have led to a surge in available data about relationships between people. In many cases, a person will own an identity on several sites and provide location information about where they live or work. The graph structure of RDF makes it natural for representing the information distributed across these sites. Combining the graph query capabilities enabled by RDF with efficient spatial processing allows us to search for people based on profile data from multiple online identities, filtered within a particular spatial boundary. As a developer, I may be organizing a working group and wish to find other developers near me with similar interests. A graph search supplemented with spatial information allows me to search for all employees of companies located within 2 miles of my company who are developers on SemWebCentral<sup>2</sup> and have listed their employers on their Facebook<sup>3</sup> account. Given location information about local coffee shops, I can also search for a coffee shop centrally located between us where we can meet. Searches like these require the ability to link and search information from multiple online sources while bounding the query and results within spatial constraints. Without spatial query techniques, graph queries like these may waste time processing all coffee shops, all SemWebCentral users, or all Facebook users before testing the location information to determine if they match the query.

#### Spatial Annotation

Web sites exist that allow users to submit reviews about all kinds of topics, including movies, books, and restaurants. In the latter case, the geospatial information is important when it is time to search reviews. A user will most likely only be interested in reviews of restaurants within a particular boundary or near a particular event.

---

<sup>2</sup> <http://www.semwebcentral.org>

<sup>3</sup> <http://www.facebook.com>

A knowledge base that allows efficient storage and search of interlinked data along with geospatial data enables an application that allows users to annotate restaurants on a map, review the restaurant, and provide details about the restaurant by linking to other data or reviewers on the Web. Such an application would allow a search for good restaurants near a particular conference.

## 2.2 Query Types

In order to support the combination of semantic and spatial data, one must consider several different types of queries. The work [3] of Egenhofer on spatial query languages based on SQL defined three types of queries:

- Queries about spatial properties
- Queries about non-spatial properties
- Queries about both spatial and non-spatial properties

Applying these straightforward concepts to a Semantic Web system yields three analogous query classes:

- Queries about spatial properties
- Queries about ontological properties
- Queries about both spatial and ontological properties

Among the spatial properties that can be queried over, several types of spatial queries have been identified:

- Queries about the spatial properties of an individual
- Queries that relate individuals to a known location (point and range queries)
- Queries that relate individuals to one another (spatial join, nearest neighbor queries)
- Queries that spatially aggregate individuals

We will now explore each of these types of spatial queries individually, as applied to a storage mechanism that also supports ontological data.

The simplest type of spatial queries is queries for the location of a known object: “*Where is the location of Jimmy’s Pizza Parlor?*” This type of query is essentially straightforward data retrieval, and does not necessarily require any specialized spatial processing. As such, a semantic system could support these queries without modification.

The second type of spatial queries relates individuals to a known location. This location could be another specific object in the knowledge base, i.e.: “*Which gas stations are within 1 mile of Jimmy’s Pizza Parlor?*” or an absolute location, i.e.: “*Which gas stations are within 1 mile of 38°N, 77°W?*”. Naturally, these queries must be crossed with ontological inference as well: “*Which restaurants are within 1 mile of Gus’s Gas?*” where ‘restaurants’ must include entities defined not specifically as restaurants, but those defined as Pizza Parlors, Sub Shops, etc., also.

The third type of spatial queries relates individuals to one another. This class includes both spatial joins and nearest neighbor queries. For example, “*Where can I go to buy bananas, milk, and a drill within a 2 mile radius?*” involves not only the

spatial join between the individual places, but also the ontological inference of the types of stores that sell the items in question.

Our spatial semantic knowledge base must be able to support all of these types of queries, and combinations thereof, efficiently.

### 3 Related Work

Significant research has gone into creating various types of efficient spatial index structures. These index structures are generally used as a supplemental index to an object-relational database. Adding the supplemental indices allows the object-relational databases to significantly increase their performance with respect to spatial queries. The indices are attached to a column or columns defined as a spatial datatype.

A wide variety of useful spatial index structures exist, each with its own positive and negative characteristics. Most fall within a small number of major families, however. These are R-trees [4], quadtrees [5], and grid files [6]. Since we will not be attempting to enhance these index structures in any way, our discussion in this area will focus on which is appropriate to attach to a semantic knowledgebase.

### 4 Design

The primary goal of SPAUK is to provide efficient spatial processing for spatial semantic systems. We can leverage the significant work that has gone into optimizing database systems for spatial data processing. These systems typically employ a supplementary spatial index to provide efficient spatial queries. As such, we chose to design SPAUK as a semantic knowledgebase capable of supporting supplementary spatial (and other) indices.

A secondary goal was to design a system such that the addition of spatial processing to the system is as transparent as possible to the user. This means that from a client's point of view, all of the data, both the semantic data and the spatial data, is still presented as a graph. To do this, the knowledge base presents itself as a standard SPARQL endpoint. This allows any clients capable of interfacing via the SPARQL protocol to utilize SPAUK.

Thus, the design must present one conceptual graph to its clients, and queries over this graph must be divided appropriately into sub-queries which can be answered by the various parts of the knowledgebase. Spatial parts of the query, including locations and spatial relationships, must be sent to the spatial index and query processor. Non-spatial components of the query must be sent to the underlying triplestore. Results must be combined from the two parts to form a coherent answer. Moreover, data which is inserted must find its way into the appropriate parts of the knowledge base.

#### 4.1 Interface

As noted before, SPAUK's external interface utilizes the SPARQL protocol for query access. However, mapping queries that include spatial instances and relationships to

SPARQL is not necessarily straightforward. There are many possible ways that one could use SPARQL for spatial data, and the ideal way has yet been attained [7]. For our prototype, we stayed within the bounds of SPARQL as it is currently defined. While this did not necessarily provide the cleanest possible spatial-semantic query interface, it did allow us to utilize other semantic web software without modification.

In order to do this, we needed to define a set of classes and properties to represent objects, attributes, and relationships that the knowledgebase could understand. Numerous candidate representations already exist. GeoRSS is a good choice for representing spatial extents because it is simple, it already has an RDF syntax, and it is based on the Open Geospatial Consortium's standard for representing spatial extents, Geography Markup Language (GML) [8]. Using another representation, such as the spatial portions of a commonly used upper ontology, could have worked just as well. For the spatial relationships, we decided to start with a set of qualitative topological relationships based on the Region Connection Calculus [9]. First, we look at an example of a Gas Station expressed using these concepts:

```
[ ] a gas:GasStation;
    gas:name "Gus's Gas";
    gas:brand gas:Exxon;
    gas:numberOfPumps "8";
    georss:where [
        a gml:Point;
        gml:pos "38 -77"
    ].
].
```

The following is an example of the query, "*Which gas stations are within 1 mile of 38°N, 77°W?*" encoded as described.

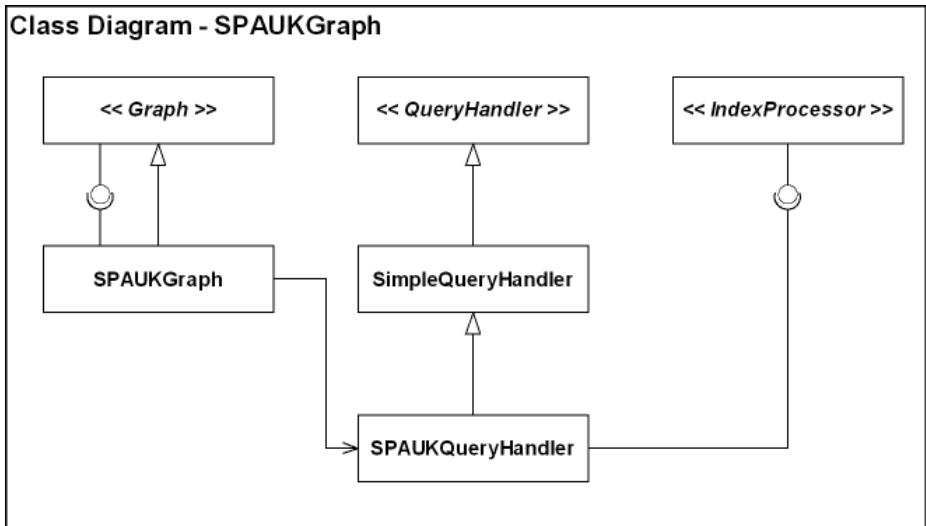
```
SELECT ?x
WHERE {
    ?x a gas:GasStation;
        georss:where ?y.
    ?y rcc:part [
        a gml:Buffer;
        gml:radius "1";
        gml:bufferGeometry [
            a gml:Point;
            gml:pos "38 -77"
        ].
    ].
}
```

This provides an interface for querying, but does not allow for insertion or deletion of triples. Since this is a necessary for our system and is not yet part of the SPARQL specification, we added HTTP interface methods for both insertion and deletion. These methods merely require a set of RDF triples being posted to the appropriate URLs. Together with the SPARQL methods, these methods define the entirety of the external interface of SPAUK.

## 4.2 Architecture

In order to facilitate interoperability and leverage existing semantic web software, the architecture of SPAUK is based on the Jena Semantic Web Framework<sup>4</sup> and Joseki<sup>5</sup>. Utilizing these tools allowed us to focus on the core query-splitting and spatial components of SPAUK.

The basic idea of the architecture is to have a specialized SPAUKGraph implementation of the `com.hp.hpl.jena.graph.Graph` Java interface that deals with the splitting and combining of the information that goes in and out of the knowledgebase. SPAUKGraph deals directly with some set of `IndexProcessors`, which represent the interface to the data stored in 1 or more supplemental indices. We address how the Graph handles queries and insertion below.



**Fig. 1.** Class diagram for the SPAUKGraph and its relation to the IndexProcessors

### 4.2.1 Data Insertion

The underlying triplestore continues to be the master copy of all information in SPAUK. All data inserted is in the form of statements, which are inserted directly into the underlying triplestore.

There is an important dichotomy between the statements in the triplestore and the contents of the supplemental indices. While the data in the triplestore is a graph, the data that goes in the supplemental indices are sets of discrete objects, i.e. spatial extents. This makes knowing when to insert an object into the supplemental index somewhat tricky. The insertion interface sees the statements being added one at a time, and must combine sets of them to form objects to be inserted. We accomplished this through the use of Jena's `InfGraph`. For each type of object that the system must

<sup>4</sup> <http://jena.sourceforge.net/>

<sup>5</sup> <http://www.joseki.org/>

watch for, a rule is added to an InfGraph layer above the underlying triplestore. The head of the rule is a function that connects to the appropriate IndexProcessor to add an object to the index. The rule will not fire until all required components are available. This rule, for example, adds points to the spatial index processor when they are inserted:

```
[point:
  (?x rdf:type gml:Point)(?x gml:pos ?pos) ->
    point(?x, ?pos)
]
```

This scheme allows for the insertion of objects into the indices without concerning us with the transactionality of the data store. In fact, if part of a geometry definition is inserted at some point, and then much later the rest of the definition is inserted, the geometry will be indexed successfully at the later point. However, it does not account for updates or deletions of statements corresponding to indexed objects. As such, indexed objects are treated as immutable within the system. If the location of a restaurant changes, rather than changing the properties of the location object to which the restaurant is attached, the restaurant must be severed from the location and a new location created.

#### 4.2.2 Querying

Querying the combined data storage is the most complicated part of the system. The appropriate parts of the query must be partitioned among the underlying triplestore and the supplemental indices, depending upon which parts are capable of most efficiently answering each piece of the query.

When the SPAUKGraph receives the query, it first splits the query based on the namespaces of the associated with the attached IndexProcessors. For each triple, the namespace of the predicate or the Class (for `rdf:type` statements) is matched against the namespaces associated with the IndexProcessors. If a namespace is not associated with an IndexProcessor, it defaults to association with the underlying triplestore. For instance, in the following query, the portion that must be processed by the spatial index processor has been italicized:

```
SELECT ?x
WHERE{
  ?x a gas:GasStation;
    georss:where ?y.
  ?y rcc:part [
    a gml:Buffer;
    gml:radius "1";
    gml:bufferGeometry [
      a gml:Point;
      gml:pos "38 -77"
    ]
  ].
}
```

Unfortunately, this assigns artificial meaning to the namespaces of which the query processor is aware. This could lead to errors in processing if the users attempted to extend the spatial ontologies in a way that the system did not understand. Since no better way to divide the statements at the query level has yet come to light, this is the method that the SPAUK prototype uses.

Once the query has been divided into appropriate parts, the SPAUKGraph must make a best-effort attempt to determine which part of the query is the most selective. Since it does not have any information about the selectivity of the query parts directly, it must ask the underlying triplestore and the IndexProcessors to approximate the selectivity of their parts of the query as an estimated number of results. Developing an appropriate cost model for this is an area of future work. In the current implementation, if the spatial query processor receives a query for objects within a specified area, it returns the highest possible selectivity and thus is chosen first. In all other cases, the system defaults to allowing the triplestore to bind first. Other possibilities include attempting to execute the different parts in parallel, however this was beyond the scope of our prototype.

An initial subquery is chosen and then executed by either the underlying triplestore or an IndexProcessor as appropriate. The bindings from this subquery are then applied to the other subqueries as they are executed. Since the linkage objects between the spatial and non-spatial portions are bound at this point, it is expected that the selectivity of the remaining bound subqueries should be extremely high.

Though SPAUK supports SPARQL, the SPARQL support is provided exclusively by the ARQ component of Jena. Thus our system deals only with queries in the form of simple graph patterns. This drastically reduces the amount of query processing work that needs to be done; however, there are cases where this design creates SPARQL queries that could not be properly optimized using the index. This could happen if the definition of the range in a range query were split over an OPTIONAL clause. Since these cases are primarily connected to poor query construction, we currently ignore them in the design.

### 4.3 Indices

The spatial index used in the prototype was a simple in-memory gridfile. This is not a particularly sophisticated spatial index; however, the software was designed such that substituting another indexing mechanism should be straightforward.

Ideally, we would like to add either a quadtree or R-tree indexing mechanism to SPAUK. Having both available to spatial applications is ideal, since both have strengths and weaknesses depending on the distribution of the data being stored. Particularly, quadtrees function better than R-trees when data is more evenly spatially distributed, and R-trees function better when data is more spatially clustered. Since there are applications which could potentially make use of both types of indices, the option of both should exist. This is analogous to the spatial index support provided in common spatial relational databases such as Oracle Spatial 10g.

## 5 Results

The SPAUK system was successfully implemented for a subset of the desired problem. Processing was implemented for two major types of spatial geometries: Points

and Polygons defined by an exterior linear ring. Rules were created to detect these geometries and insert them into the index. Two spatial relationships were implemented over these polygons, *connected* and *part*. These allowed us to sufficiently test the query splitting mechanism.

Unfortunately, without the creation of spatial semantic benchmarks, we do not yet have a way to empirically test the performance of the SPAUK system. However, consideration of the prior art in the object-relational database realm and a careful look at the index structures demonstrates that the technique is superior.

Consider attempting to build a system for spatial semantic data without any spatial indexing. A query for all restaurants that are in a 2 mile radius from a given point would clearly be  $O(n)$  in the number of restaurants, since the system would have to compare each and every restaurant's location to the spatial buffer area. However, if a quadtree was used for spatial indexing, we would expect the time to find objects in the radius to be logarithmic.

## 6 Conclusion

While we have not yet done formal analysis of the performance improvement caused by using a supplemental spatial index, examples of the technique in the object-relational database world, simple analysis of the algorithms involved, and preliminary usage of the SPAUK system have shown that the approach is indeed valid. Attaching a semantic GIS client to the SPAUK system provides responsive spatial semantic query capability. We believe that this type of system enables a new class of semantic applications whose full potential cannot yet be conceived. Waldo Tobler's "first law of geography" states, "Everything is related to everything else, but near things are more related than distant things." [10] Since a goal of the Semantic Web is to maximize the meaning of relationships, spatial information processing cannot be ignored.

## 7 Future Work

The first major piece of future work for the SPAUK system will be to fully implement the GeoRSS geometry types and the RCC8 spatial relations. This will provide a fully usable system for experimentation with spatial semantic data storage, and hopefully provide others with a method of building spatial semantic applications when it soon becomes open source.

The second piece of future work involves significantly more formal performance testing. However, this will require several other advancements. First, a benchmark for spatial semantic data must be created. This could very well be an enhancement of the Lehigh University Benchmark (LUBM) [11]. Secondly, SPAUK would need to be attached to a more robust spatial index, such as a persistent R-tree. With these modifications in place, SPAUK will be formally compared to a semantic spatial system in which spatial calculations are performed only as function calls in rules.

Finally, extending the SPAUK implementation with a temporal index or other indices is very desirable. The architecture is built not just for one supplemental index, but for many; hopefully it can provide benefit for a wide variety of application areas.



**Acknowledgements.** We wish to thank Mike Dean and BBN Technologies for their support of this effort and helpful comments on the paper.

## References

1. Connected Services Framework 3.0 Developers Guide. Microsoft (2006)
2. Lassila, O., Hendler, J.: Embracing Web 3. IEEE 11, 90–93 (2007)
3. Egenhofer, M.: Spatial SQL: A Query and Presentation Language. IEEE Transactions on Knowledge and Data Engineering 6, 86–95 (1994)
4. Guttman, A.: R-trees: a dynamic index structure for spatial searching. ACM Press, New York (1984)
5. Finkel, R.A., Bentley, J.L.: Quad trees a data structure for retrieval on composite keys, pp. 1–9. Springer, Heidelberg (1974)
6. Nievergelt, J., Hinterberger, H., Sevcik, K.C.: The Grid File: An Adaptable, Symmetric Multikey File Structure, vol. 9, pp. 38–71. ACM Press, New York (1984)
7. Kolas, D.: Supporting Spatial Semantics with SPARQL (2007)
8. ISO/TC 211/WG 4/PT 19136 Geographic information - Geography Markup Language (GML) (2004)
9. Cohn, A.G., Bennett, B., Gooday, J., Goss, N.M.: Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. GeoInformatica 1, 275–316 (1997)
10. Tobler, W.R.: A Computer Movie Simulating Urban Growth in the Detroit Region. JSTOR 46, 234–240 (1970)
11. Guo, Y., Pan, Z., Heflin, J.: LUBM: A Benchmark for OWL Knowledge Base Systems. Journal of Web Semantics 3, 158–182 (2005)

# Recipes for Semantic Web Dog Food — The ESWC and ISWC Metadata Projects

Knud Möller<sup>1</sup>, Tom Heath<sup>2</sup>, Siegfried Handschuh<sup>1</sup>, and John Domingue<sup>2</sup>

<sup>1</sup> Digital Enterprise Research Institute, National University of Ireland, Galway  
knud.moeller@deri.org, siegfried.handschuh@deri.org

<sup>2</sup> Knowledge Media Institute and Centre for Research in Computing, The Open University, Milton Keynes, UK  
{t.heath, j.b.domingue}@open.ac.uk

**Abstract.** Semantic Web conferences such as ESWC and ISWC offer prime opportunities to test and showcase semantic technologies. Conference metadata about people, papers and talks is diverse in nature and neither too small to be uninteresting or too big to be unmanageable. Many metadata-related challenges that may arise in the Semantic Web at large are also present here. Metadata must be generated from sources which are often unstructured and hard to process, and may originate from many different players, therefore suitable workflows must be established. Moreover, the generated metadata must use appropriate formats and vocabularies, and be served in a way that is consistent with the principles of linked data. This paper reports on the metadata efforts from ESWC and ISWC, identifies specific issues and barriers encountered during the projects, and discusses how these were approached. Recommendations are made as to how these may be addressed in the future, and we discuss how these solutions may generalize to metadata production for the Semantic Web at large.

## 1 Introduction

This paper reports on the efforts made to generate, maintain and deploy structured metadata for both the European and the International Semantic Web Conferences (ESWC and ISWC). In particular, we discuss the experiences gained and lessons learned during ESWC2006 and ISWC2006. However, the work done for those conferences is continued and further refined in the 2007 ESWC in Innsbruck, Austria and the 2007 ISWC in Busan, South Korea. The main contributions of this paper are the in-depth reporting of real deployments of Semantic Web technologies, and analysis of the technical lessons learned. Through analyzing our own experiences we provide a number of recommendations to others pursuing similar projects. Whilst the deployments were carried out by Semantic Web enthusiasts, the lessons learned apply to a wide range of potential deployment scenarios. As the Semantic Web moves from research and development into deployment and adoption, understanding these experiences and their implications becomes increasingly important. Crucially, challenges encountered by

those knowledgeable about the Semantic Web will almost certainly be encountered by those with less experience.

The following section discusses the aims of the projects, and related work. Section 3 characterizes the source data from which the metadata was generated, and discusses the output requirements. Sections 4.1 and 4.2 discuss the approaches taken at ESWC2006 and ISWC2006 respectively, the problems encountered and the solutions applied. In Section 5, we make recommendations for those undertaking such efforts at future conferences. These may also be relevant to any generation of metadata on a larger than personal scale.

## 2 Background and Aims of the Metadata Projects

The ESWC2006 and ISWC2006 metadata projects addressed a number of related aims. Firstly, to generate data for use with existing Semantic Web applications (thereby providing a showcase for such applications) and to further Semantic Web research. Secondly, to evaluate the practicality and feasibility of producing, managing, and deploying Semantic Web data for events such as ESWC2006 and ISWC2006. Only by undertaking such projects can we fully understand these processes. By reporting our experiences we believe we can inform similar projects in the future, whilst also highlighting the challenges of producing, managing and deploying Semantic Web data. Finally we argue that the Semantic Web community has an obligation to carry out such activities, if we are to better comprehend the challenges faced by others who may wish to adopt Semantic Web technologies. “Eating our own dog food” [4] is an essential mechanism by which to gain the appropriate insights.

### 2.1 Related Work

The use of technologies to support technology-related events is not novel. For example, [8] reports on the deployment of IRC backchannels at the ACM 2004 Conference on Computer Supported Cooperative Work (CSCW 2004), as a means to complement existing communication channels at the event.

Given the specific metadata requirements of the Semantic Web compared to other technological advances, the production of structured metadata describing events has greatest prevalence at conferences in the Semantic Web and related fields. For example, [3] reports on the W3Photos project, an initiative to enable the addition of semantic descriptions to photos taken at the Worldwide Web Conferences and related events. At the Worldwide Web Conference in 2006, a version of the conference programme was produced in RDF/XML<sup>1</sup> and deployed together with related services aimed at end users, such as the mSpace Schedule Explorer<sup>2</sup>. Organizers of the 4th International Semantic Web Conference in 2005 made the PiggyBank semantic web browser extension<sup>[6]</sup> available to delegates,

<sup>1</sup> <http://www2006.org/programme/dynamic>

<sup>2</sup> <http://www06.mspace.fm/>

as a means to annotate items related to the conference such as papers presented or local restaurants.

Whilst each of these initiatives is commendable, they share a common limitation due to the restricted scope of the metadata produced. To the best of our knowledge, the metadata projects at ESWC2006 and ISWC2006 represent the first instances of conferences attempting to offer comprehensive semantic descriptions of the event. Furthermore, as a result of this comprehensiveness, we believe that these events have moved significantly closer to integrating metadata production with broader conference workflows than previous events. These integration attempts raised a number of issues and challenges, which will be examined in the remainder of this paper.

### 3 Problem Description

In the course of the metadata efforts for both ESWC2006 and ISWC2006 we faced a number of specific problems and challenges, which will be discussed in this section. These challenges can be divided into two main areas: those arising from the specific characteristics of the input source data, and those arising from the requirements on the output data (the RDF metadata). If we view the metadata efforts as a process, then those challenges define its input and output, respectively. We will describe both in turn in the following sections.

#### 3.1 Data Characteristics

When describing an academic conference, the three object types of greatest interest are people, events, and publications. People may take the role of paper authors, delegates at the conference, and committee members. Events may consist of talks (e.g. paper or poster presentations), conference sessions in which several papers are presented, or entire tracks. Various kinds of non-academic events also occur, such as meals, social events, and even coffee breaks. Publications can consist of full papers and poster/demo papers, plus a bound or electronic volume of the entire conference proceedings. In addition, artefacts such as sets of slides can be of great value, whilst not being formally published. Apart from this core set of data, other kinds of information, such as rooms within the conference venue, or sponsoring organizations, can be relevant.

In the course of producing metadata for both conferences, it became apparent that the source data from which both corpuses were produced had some novel characteristics relative to other Semantic Web data sets we had previously encountered. We predict that these characteristics are common to academic conferences in general, and found that they raised a number of challenges in producing the RDF descriptions of the conferences.

Firstly, data sets related to academic conferences are typically small in volume, compared to existing, established data sets available as RDF, such as DBLP<sup>3</sup>.

<sup>3</sup> <http://www4.wiwiw.fu-berlin.de/dblp/>

Secondly they are heterogenous in nature, covering concepts as diverse as people, places, artefacts, and events. Consequently there are few economies of scale in producing this data, as many different export or conversion tools may be required, relative to large, homogenous datasets from one central source.

Thirdly the input data set typically originates from many different sources, such as conference submission and registration systems, email messages, or text documents, and is traditionally managed by many different people, and each may use different methods to manage the data. In our experience these ranged from spreadsheets, to lists in documents, and HTML pages. This presented a number of challenges, as very few of these systems were already web-based, or designed to publish to public web sites.

### 3.2 Output Requirements

Just as the input data influenced the metadata efforts, so did the requirements on the output data. The metadata was to be used in a Semantic Web context, and so needed to adhere to the principles of linked, Semantic Web data. This made RDF a natural choice of data model. It was also important to use established vocabularies and ontologies. This was done to an increasing extent, starting with the integration of FOAF and SWRC at ESWC2006 and continuing with the iCalendar and BibTeX standards at ISWC2006. Where existing URIs could not be used to identify resources (as was often the case) new URIs were necessary. These were minted following consistent patterns and based on domains within our control. This allow for the configuration of properly de-referenceable URIs.

Many tools that used the conference metadata were targeted at visualisation, and so needed to traverse the RDF graph in all ways and access resources from all kinds of angles. This made the use of inverse properties very useful, if not necessary. Also, inverse functional properties are a good way of getting around the URI problem, and transitive properties can simplify ontology modeling significantly. OWL lite provides these features, and was therefore chosen to model the ontologies.

## 4 Approaches Taken

### 4.1 3rd European Semantic Web Conference (ESWC2006)

The ESWC2006 Semantic Web Technologies project<sup>4</sup> combined the creation and publishing of metadata describing the conference, with deployment of a range of applications (such as a semantic Wiki, photo annotation tool, and semantic search engine) that were intended to enhance the conference for delegates by making use of the ESWC2006 Conference Ontology and associated RDF/XML dataset. Fuller descriptions of these applications are given in [5] alongside the results of an evaluation into how they were received by conference delegates.

---

<sup>4</sup> <http://www.eswc2006.org/technologies>

**Ontology.** Existing event and conference ontologies, such as the AKT Reference Ontology<sup>5</sup>, the Conference ontology by Jen Golbeck<sup>6</sup>, and the eBiquity Conference Ontology<sup>7</sup> were initially surveyed to assess their suitability for use in the ESWC2006 Technologies project. These ontologies were found to lack the expressivity required for the project. Consequently an exercise was carried out to model the Conference domain as the basis for a new ESWC2006 Conference Ontology. The ESWC2006 Conference Ontology has the following top-level classes: Artefact, Call, Event, Place, Role, Sponsorship, all of which (except Sponsorship) are extensively sub-classed to provide a high degree of expressivity. In contrast to other ontologies, the ESWC2006 Conference Ontology explicitly models relationships between people, roles, and events. So for example, the act of *giving a paper at a conference* is modeled in terms of a person holding a role of presenter at a specific *talk event*, with which there may be one or more associated *artefacts*, such as a paper or a slide set.

Wherever possible the ontology sought to reuse existing classes from widely deployed ontologies, instead of starting from scratch. Consequently, the ontology makes use of the Person and ResearchTopic classes from the FOAF<sup>8</sup> and SWRC<sup>[12]</sup> ontologies, respectively. For example, the `eswc:heldBy` property has a domain of `eswc:Role` and a range of `foaf:Person`. It is hosted at<sup>9</sup>, according to the “Best Practice Recipes for Hosting RDF Vocabularies”<sup>[10]</sup>.

**Metadata Creation.** RDF descriptions<sup>[10]</sup> were made available of the ESWC2006 Organizing Committee, the tracks, sessions, talks, and roles that existed at the conference, the papers, posters, and demos presented, rooms in the conference venue, and delegates who opted into the public Semantic Delegates List. Where an Organizing Committee member was responsible for a particular area (such as posters and demos, or workshops), that individual provided the source data for the corresponding RDF descriptions. These data sets were generally small in volume, and came in a range of different formats, such as Excel spreadsheets, tables in documents, and HTML, as this was generally how members of the organizing committee managed the data for which they were responsible. These characteristics influenced how the RDF descriptions were then produced. It was not deemed efficient to automate the production of RDF descriptions where the source data was not already well structured, or in areas such as workshops, tutorials, or demos, where there were relatively few instances. Consequently, the majority of descriptions were produced manually using a generic XML editor. Aside from being very resource intensive this created issues with maintenance, for example when sessions changed venue.

The one area where automation was clearly beneficial was in producing the Semantic Delegates List. On an opt-in basis, additional data was collected from

<sup>5</sup> <http://www.aktors.org/publications/ontology/>

<sup>6</sup> <http://www.mindswap.org/~golbeck/web/www04photo.owl>

<sup>7</sup> <http://ebiquity.umbc.edu/ontology/conference.owl>

<sup>8</sup> <http://xmlns.com/foaf/0.1>

<sup>9</sup> <http://www.eswc2006.org/technologies/ontology>

<sup>10</sup> <http://www.eswc2006.org/rdf>

delegates at registration via the conference registration system. Those who gave their consent were featured in the Semantic Delegates List, an RDF representation of people present at the conference. This gave basic information such as the delegate's name, and a hash of their mailbox URI, in addition to any further information they had supplied such as their homepage URI, the URI of their FOAF file, their workplace homepage, and their areas of interest. The information provided by delegates was exported from the conference registration system as an Excel spreadsheet, and processed by a PHP script to generate RDF according to the FOAF ontology. This strategy proved very effective where specific pieces of information has been given dedicated fields in the registration system. However, due to limitations in the registration system, information about delegates' areas of interest had to be provided into one text field, despite the interests coming from a fixed vocabulary. This presented numerous challenges when parsing the data to produce RDF.

An additional challenge concerning topics of interest centered around how the Areas of Interest for a specific conference are managed and integrated with other data sources. For ESWC2006, the existing topic hierarchy from the SWRC ontology [12] was used when collecting delegates' interests. A preferable approach would be for conference topics to be marked up using SKOS<sup>11</sup> from an early stage, as this topic listing could then be reused within other systems deployed for the conference. Such an approach would also facilitate the creation of mappings between topics occurring at related conferences whilst still allowing local flexibility in how topics are defined and how they evolve.

One objective of the ESWC2006 Technologies project was to bring the same degree of semantic descriptions to workshops, as to the main conference track. This presented significant challenges, in that all workshops maintained separate web sites maintained by different people, and all structured differently. The intended approach was to use GRDDL<sup>12</sup> to generate RDF descriptions of workshop proceedings, participants, and committees. This objective was not met at ESWC2006 due to time constraints, however, we believe it would be a valuable objective for future conferences.

Finally, creation of the metadata for ESWC2006 required the minting of new URIs to identify all relevant entities. A lack of tools to support this process made it difficult to ensure they were being used consistently across all RDF descriptions (for example in the RDF describing full papers, and in the Semantic Delegates List). The most pragmatic solution to this problem involved simply using a consistent syntax for minted URIs.

**Metadata Deployment.** In deploying the RDF descriptions of the conference, close integration of the RDF with the conventional Web site was desired. Consequently, and to aid human management of the data, many separate RDF files were created and deployed on the conference web site<sup>13</sup>. The *FOAF Autodiscovery*

<sup>11</sup> <http://www.w3.org/2004/02/skos/>

<sup>12</sup> <http://www.w3.org/TR/grddl/>

<sup>13</sup> <http://www.eswc2006.org>

technique<sup>[14]</sup> was used to link HTML pages to their corresponding RDF files. The Apache web server running the conference web site was also configured to ensure URIs could be de-referenced, using approaches similar to those outlined in [1].

## 4.2 5th International Semantic Web Conference (ISWC2006)

The approach taken and decisions made during the planning of the ISWC2006 metadata efforts<sup>[15]</sup> were largely influenced by the metadata efforts of ESWC2006. In this section we will therefore concentrate on the aspects where changes or a development between the two conferences took place. Just like for for ESWC2006, the data corpus was made available prior to the conference, and a number of applications who used the data were featured on the metadata website.

**Ontology.** Since the ESWC schema had already proved useful and adequate, its adoption for ISWC2006 was clearly desirable. By using the same schema, both datasets become interoperable, can be queried and visualized in an integrated fashion and will together have greater impact on the community. However, a number of issues were identified which needed to be addressed. The main points were: (i) increased integration of existing standards and vocabularies, (ii) rearrangement of the document concept space, and (iii) fully utilising the possibilities OWL lite offers, such as inverse properties, which aid in data visualization.

Addressing those points did not mean literally changing the existing ESWC ontology, but instead establish a new ISWC ontology. However, instead of starting afresh, this new ontology simply imports the old one, adds a number of new concepts and properties, makes some additional statements about existing concepts and properties and deprecates others. In effect, a dataset conforming to the ISWC ontology is still largely compatible with the ESWC ontology. In the following paragraphs we will discuss some of the changes made in moving from one conference ontology to the next. We think this discussion illustrates very well some typical issues for ontology evolution.

Even though the ESWC ontology integrated FOAF for people-related metadata, it still used custom classes and properties to represent events and publications. For ISWC, we decided to integrate more established standards for those kinds of entities. For event data, the iCalendar format [2] was adopted. Since iCalendar itself does not have an RDF schema, we decided to use the schema suggested in a W3C interest group note<sup>[16]</sup>. For representing publications, we adopted the widely used Bib<sub>T</sub>E<sub>X</sub> schema [11]. A number of implementations in RDF exist, but we decided to use the Bib<sub>T</sub>E<sub>X</sub>-related classes from the SWRC (Semantic Web for Research Communities) ontology<sup>[17]</sup>, due to reasonably wide usage (outside SWRC-related projects, the ontology is also used by Flink [9] and openacademia [7]) and tool support.

<sup>14</sup> <http://rdfweb.org/topic/Autodiscovery>

<sup>15</sup> [http://iswc2006.semanticweb.org/program/tech\\_links.php](http://iswc2006.semanticweb.org/program/tech_links.php)

<sup>16</sup> <http://www.w3.org/TR/rdfcal>

<sup>17</sup> <http://ontoware.org/projects/swrc/>



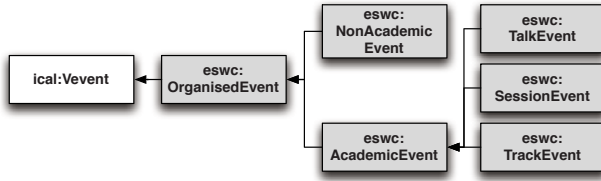


Fig. 1. Integrating the iCalendar Ontology

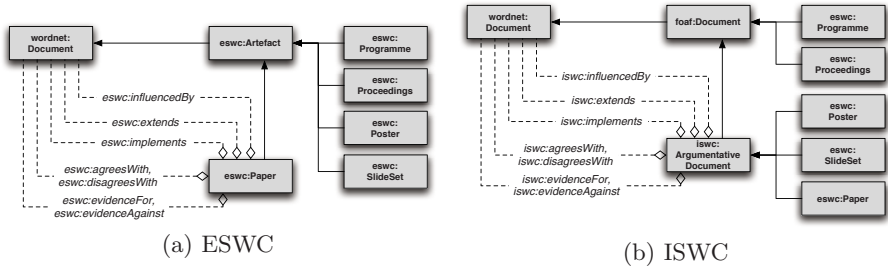
The integration with both iCalendar and Bib<sub>T</sub>E<sub>X</sub> was performed in a straightforward fashion. We chose an appropriate concept from the ESWC ontology as the entry point and established a subclass relationship to a matching concept in the external ontology. Figure 1 gives an example of how this worked with the iCalendar integration. The `eswc:OrganisedEvent` concept, which is the superconcept of all event types in the ESWC ontology was made a subclass of `ical:Vevent`, thereby allowing all ESWC event types to be treated as iCalendar events.

The document concept space in the ESWC ontology is homogeneous — papers, posters, slide sets, proceedings and programmes are all grouped under a common superconcept `eswc:Artefact`. Properties regarding scientific discourse such as `eswc:influencedBy` or `eswc:agreesWith` are defined to apply to instances of `eswc:Paper` only (see Fig. 4.2). For ISWC, we decided to split the document concept space into *argumentative* documents (papers, posters and slide sets) and *non-argumentative* documents (proceedings and programme). This allowed us to redefine the properties listed above so that they apply to all kinds of argumentative documents. Also, we wanted to further strengthen the integration with FOAF and establish a relation to FOAF’s `Document` class. In order to achieve this, we introduced a new class `iswc:ArgumentativeDocument`, which is the superclass of all relevant document classes in the ESWC ontology, and a subclass of `foaf:Document`<sup>18</sup>. All document classes which are deemed non-argumentative are direct subclasses of `foaf:Document`. In addition, we had to introduce new properties such as `iswc:agreesWith` and at the same time deprecate the ones like `eswc:agreesWith` (see Fig. 4.2).

To utilize the expressiveness of OWL lite to a greater degree, the ISWC ontology defined `owl:inverseOf` relationships for a number of properties from the ESWC ontology. Finally, transitivity was used in defining properties that express sub- and super-event relationships (e.g. a talk is a sub-event of a session, is a sub-event of a track).

**Metadata Creation.** The process of creating the metadata for ISWC2006 was affected by the same issues that had already surfaced during ESWC2006. Again, the source data came in a variety of different formats, none of which were particularly suitable for automatic transformation. This reflects the fact that,

<sup>18</sup> Note that, since existing statements from the ESWC ontology are not changed, all document classes are still also subclasses of `eswc:Artefact`!



**Fig. 2.** The Evolution of the Document Concept Space from ESWC to ISWC

even though ESWC and ISWC are Semantic Web conferences, data integration wasn't yet a critical point to consider in organizing them.

The main portion of the generated data was papers, authors and events such as talks and panels. In addition to that, conference and session chairs were represented, as well as those conference attendees who volunteered to provide a FOAF profile of themselves. The source data for both papers and authors was an Excel table provided by the proceedings chair, as well as the actual PDF documents of the papers. Automated generation of structured metadata was partially possible. However, author email addresses and affiliations, as well as paper abstracts still had to be extracted by hand. The source data for the events list was a timetable in HTML, which was sufficiently unstructured to make automatic conversion to RDF difficult.

To ease the creation of RDF, we therefore decided to add an intermediate step. We first generated ordinary, non-RDF Bib<sub>T</sub>E<sub>X</sub> and iCalendar documents. The benefit was that we were now able to edit and maintain the data with dedicated tools that are tailored towards dealing with such data and make this task much less frustrating and error-prone than maintaining the data in a general-purpose text editor. Also, it was now possible to use other tools to automatically generate the desired target RDF (we used a modified version of the customizable `bibtex2rdf` converter<sup>19</sup> and Python scripts made available by the W3C Semantic Web Interest Group<sup>20</sup>).

Not all data could be generated and maintained in this way. Email addresses and affiliations of authors cannot be represented in ordinary Bib<sub>T</sub>E<sub>X</sub>, and so had to be added by hand later. Similarly, links between papers and their talks, links between events (e.g. between individual talks and their session), as well as the various conference and session chairs were added manually.

Table 1 provides an overview of the two datasets that were generated for both conferences. Even though almost identical ontologies were used, some obvious differences can be observed. The ISWC dataset did not contain any workshops, and was thus less comprehensive than that of ESWC. As a result, much fewer instances of `eswc:Role` and `eswc:Artefact` were defined. On the other hand, it

<sup>19</sup> <http://www.l3s.de/~siberski/bibtex2rdf/>

<sup>20</sup> <http://www.w3.org/2002/12/cal/>

**Table 1.** Overview of the ESWC2006 and ISWC2006 datasets

Top Level Class	ESWC2006	ISWC2006
<code>eswc:OrganisedEvent</code>	129	130
<code>eswc:Role</code>	130	48
<code>eswc:Artefact</code>	100	68
<code>foaf:Person</code>	189	301
<code>swrc:Topic</code>	59	0
<code>eswc:Place</code>	8	0
Total Number of Triples	2939	5902

is striking that the ISWC dataset has a much larger number of triples (before inferencing). This is mainly due to a small change in the ontology, which was introduced to improve the immediate usefulness of the data for consuming SW tools: the authors of publications were modelled as complex `foaf:Person` objects with various assertions about them, whereas at ESWC they were represented by a URI that was not further defined. This also explains the rise in person instances from ESWC to ISWC.

**Hosting/Serving the Metadata.** In a move from ESWC2006, we decided to use a database solution for hosting, instead of individual files. The complete dataset first created in the form of RDF documents and then loaded into a Jena RDF store<sup>21</sup> and made available through a Joseki<sup>22</sup> SPARQL server at a public URI<sup>23</sup>. Using a hosting setup based on an RDF store made reasoning capabilities available to all external consumers of the conference data (mainly the tool providers), and, more importantly, made it possible to perform queries over the complete dataset. Again, the webserver was configured to allow dereferencing of resource URIs.

## 5 Recommendations for Semantic Web Dog Food

Having encountered the challenges reported above, work has been undertaken towards their resolution; both in the evolutions seen between ESWC2006 and ISWC2006, and in further changes being made in how the metadata will be produced for ESWC2007 and ISWC2007. However, many issues remain. Consequently we offer the follow recommendations to others undertaking similar projects in the future, and to those planning any deployment of metadata for the Semantic Web.

### 5.1 Process Recommendations

- **Release metadata early**, thereby allowing tools to be developed using real data, and deployed in advance of the conference. This may be of particular

<sup>21</sup> <http://jena.sourceforge.net/>

<sup>22</sup> <http://www.joseki.org/>

<sup>23</sup> <http://128.192.251.191:8080/joseki/iswc>

significance where tools are designed to support delegates in planning their travel to and schedule at the conference. An example is the conference scheduler that was deployed during ISWC2006<sup>24</sup>.

- **Provide sample data** with which developers can work in the period before the final conference data is available. Datasets from previous conferences may be sufficient for this purpose.

Addressing these recommendations requires more integrated and efficient workflows across the entire conference planning process.

## 5.2 Workflow Recommendations

At ESWC2006 and ISWC2006 metadata was produced centrally by one person, but from heterogenous sources. This differs from a number of existing metadata production workflows. For example, Semantic Web-compatible versions of databases such as DBLP (see Sect. 3.1) are produced by exposing existing databases as RDF, enabling a simple workflow and single source to yield large amounts of data. In contrast, production of personal FOAF files has traditionally been carried out by one individual creating RDF by hand, or semi-manually. We argue that production of conference metadata to date has followed a third workflow, of production by one individual with relatively low degrees of automation, but from many sources. This may provide an interesting case study relevant to Semantic Web deployments in general, as not all metadata will be produced on a very large or very small scale, as in the DBLP and FOAF examples. However, we believe that future conference workflows should attempt to distribute production of metadata across many individuals, ensuring production is as automated and integrated with wider conference organization as possible. In addition to increasing efficiency, automating production may also help ensure comprehensiveness of the resulting dataset — something that can be hard to achieve with more manual production. Automation may be aided by the use of common platforms for managing conference information that are capable of publishing data directly to the Semantic Web, or through an intermediate conversion step. On this basis we offer the following recommendations:

- **Reuse** wherever possible. Ontologies, and tools and methodologies for producing data may all be sufficiently generic to be reused. Naturally, the SWC conferences all build on the work that has been done at the preceding conferences, and all reuse vocabularies such as FOAF, iCal or BibTeX.
- **Integrate metadata production into conference workflows** at all possible stages: in the work of the organizing committee members, and in the submission and registration systems. For ISWC2007, this has been applied by using the same submission system throughout (for both conference and all workshops) and ensuring that the system provides all necessary data.
- **Devolve responsibility for metadata production** to authors, delegates, and members of the organizing committee. For example, for ISWC2007, all

<sup>24</sup> <http://schedule.semanticweb.org>

authors of accepted papers are required to add additional data to be used in the metadata corpus.

### 5.3 Technical Recommendations

- **Define clear formats for minted URIs.** This will help ensure consistency in the absence of URI management tools. E.g., starting with ESWC2007, all conference metadata will be hosted at <http://data.semanticweb.org>, and the URIs for all entities adhere to strict format within this domain, such as <http://data.semanticweb.org/conference/eswc/2007/person-59437>.
- **Support the de-referencing of URIs** by configuring web servers appropriately. Servers should ideally support HTTP 303 redirects and Content Negotiation.
- **Provide crawlable and browsable metadata** on the conference web site that is well integrated with conventional HTML content.
- **Provide a queryable repository of all metadata** that supports the SPARQL query language and protocol.
- **Provide mechanisms to update metadata** that require minimum manual intervention.
- **Maintain a central repository of data** in order to deliver the features recommended above. For the SWC conferences starting with ESWC2007, this is <http://data.semanticweb.org>.
- **Move towards a layered architecture** that takes a service-oriented approach, adding a Services layer on top of Data and Query layers.

### 5.4 Functionality Recommendations

- **Define and enable, at the Services layer, a range of common functions** associated with a conference, or shown to be beneficial to delegates.
- **Support conference organization workflows** by exploiting Semantic Web technologies.

Common functions or services might include a session attendance suggestion service, such as that proposed in the ESWC2006 Design Challenge<sup>25</sup>. Prior to the conference a travel arrangement service may also be beneficial. Different services are likely to be of value before, during, and after the conference. Understanding which provides maximum value at each point in time requires further investigation.

There are also many areas in which Semantic Web technologies may assist with organization of a conference itself, and these deserve further consideration. For example, Semantic Web technologies may be able to assist with Programme Committee selection, ensuring that a suitable range of location, affiliation, and expertise is achieved. Research into scheduling on the Semantic Web may be able to inform the development of systems able to plan transfers between airports and the conference venue. Using Semantic Web technologies in planning a conference adds further motivation for using structured data at all stages of the planning process.

<sup>25</sup> <http://www.eswc2006.org/technologies/designchallenge/semantic-conference-program.eps>.

## 6 Conclusions

We argue that the metadata efforts at ESWC2006 and ISWC2006, and the ongoing efforts of 2007, have brought numerous benefits to the Semantic Web community, such as the creation and deployment of significant datasets which can be used in future research, opportunities for the deployment of applications, and demonstrations of the community's commitment to eating its own dog food. That these efforts are seen as important is supported by the ongoing efforts to generate and deploy metadata at ESWC2007 and ISWC2007. The ontology used for the metadata efforts which was started with ESWC2006, was refined during the following conferences and has now reached a relatively stable version<sup>26</sup>.

Furthermore, and perhaps most significantly, the projects represent a form of action research by which the community can identify issues for ongoing research, and barriers to wider adoption of the Semantic Web. We believe that the value of this paper lies in the fact that we report on real world use cases of Semantic Web technology, albeit in the context of research-focussed Semantic Web conferences, identify issues encountered, and make recommendations about how they may be overcome. These are important lessons to be learned for the application of Semantic Web technology in general; in broader settings and by other groups outside the core Semantic Web community. Despite being Semantic Web enthusiasts, in coordinating the metadata projects at both conferences we found the experience of eating our own dog food just as challenging as the idiom implies. We hope that by sharing our recipes and ideas for improvements, eating one's own dog food can become significantly more appetizing.

## Acknowledgements

The work presented in this paper was supported (in part) by the Lión project supported by Science Foundation Ireland under Grant No. SFI/02/CE1/I131 and (in part) by the European project NEPOMUK No FP6-027705. Furthermore, it was supported (in part) by the Advanced Knowledge Technologies (AKT) project, an Interdisciplinary Research Collaboration (IRC), which is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01. The AKT IRC comprises the Universities of Aberdeen, Edinburgh, Sheffield, Southampton and the Open University.

## References

1. Bizer, C., Cyganiak, R., Heath, T.: How to publish linked data on the Web (2007), <http://sites.wiwi.fu-berlin.de/suhl/bizer/pub/LinkedDataTutorial/>
2. Dawson, F., Stenerson, D.: Internet calendaring and scheduling core object specification (iCalendar), RFC 2445 (1998), <http://www.ietf.org/rfc/rfc2445.txt>

<sup>26</sup> <http://data.semanticweb.org/ns/swc/ontology>

3. Elin, G.: Is a picture worth a thousand clicks? Challenges of adding semantic data to images. In: IDSW 2004. WWW2004 Workshop on Interaction Design and the Semantic Web, New York, NY, USA (May 2004)
4. Harisson, W.: Eating your own dog food. *IEEE Software* 23(3), 5–7 (2006)
5. Heath, T., Domingue, J., Shabajee, P.: User interaction and uptake challenges to successfully deploying Semantic Web technologies. In: The 3rd International Semantic Web User Interaction Workshop SWUI2006 at ISWC2006, Athens, GA, USA (November 2006)
6. Huynh, D., Mazzocchi, S., Karger, D.: Piggy Bank: Experience the Semantic Web inside your Web browser. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
7. Klein, M., Mika, P., Serban, R.: Semantics-based publication management using RSS and FOAF. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
8. McCarthy, J.F., Boyd, D.M.: Digital backchannels in shared physical spaces: Experiences at an academic conference. In: CHI 2005 extended abstracts on Human factors in computing systems, Portland, OR, USA (April 2005)
9. Mika, P.: Flink: Semantic Web technology for the extraction and analysis of social networks. *Journal of Web Semantics* 3 (2005)
10. Miles, A., Baker, T., Swick, R.: Best practice recipes for publishing RDF vocabularies (March 14, 2006), <http://www.w3.org/TR/swbp-vocab-pub/>
11. Patashnik, O.: BibTeXing, February 8, BibTeX Documentation (1988)
12. Sure, Y., Bloehdorn, S., Haase, P., Hartmann, J., Oberle, D.: The SWRC ontology - Semantic Web for research communities. In: Bento, C., Cardoso, A., Dias, G. (eds.) EPIA 2005. LNCS (LNAI), vol. 3808, Springer, Heidelberg (2005)

# Matching Patient Records to Clinical Trials Using Ontologies

Chintan Patel<sup>3</sup>, James Cimino<sup>3</sup>, Julian Dolby<sup>1</sup>, Achille Fokoue<sup>1</sup>,  
Aditya Kalyanpur<sup>1</sup>, Aaron Kershenbaum<sup>1</sup>, Li Ma<sup>2</sup>, Edith Schonberg<sup>1</sup>,  
and Kavitha Srinivas<sup>1</sup>

<sup>1</sup> IBM Watson Research Center, P.O. Box 704, Yorktown Heights, NY 10598, USA  
{dolby, achille, adityakal, aaronk, ediths, ksrinivs}@us.ibm.com

<sup>2</sup> IBM China Research Lab, Beijing 100094, China  
malli@cn.ibm.com

<sup>3</sup> Columbia University Medical Center  
{chintan.patel, ciminoj}@dbmi.columbia.edu

**Abstract.** This paper describes a large case study that explores the applicability of ontology reasoning to problems in the medical domain. We investigate whether it is possible to use such reasoning to automate common clinical tasks that are currently labor intensive and error prone, and focus our case study on improving cohort selection for clinical trials. An obstacle to automating such clinical tasks is the need to bridge the *semantic gulf* between raw patient data, such as laboratory tests or specific medications, and the way a clinician interprets this data. Our key insight is that matching patients to clinical trials can be formulated as a problem of semantic retrieval. We describe the technical challenges to building a realistic case study, which include problems related to scalability, the integration of large ontologies, and dealing with noisy, inconsistent data. Our solution is based on the SNOMED CT<sup>®</sup> ontology, and scales to one year of patient records (approx. 240,000 patients).

## 1 Introduction

This paper describes a large case study that explores the applicability of ontology reasoning to problems in the medical domain. Currently, medical ontologies are primarily used for terminology services. We explore whether it is possible to use ontology reasoning to automate common clinical tasks, such as cohort selection of patients for clinical trials, infectious disease monitoring, and clinical decision support. An obstacle to automating these tasks is the need to bridge the *semantic gulf* between raw patient data, such as laboratory tests or specific medications, and the way a clinician interprets this data. For example, a laboratory report which indicates the presence of a class of organisms implies the presence of an infectious disorder; similarly, certain types of chemotherapy drugs imply the presence of certain cancers. Using ontologies, it should be possible to automate this interpretation process and build a reusable solution. Toward this goal, we focus our case study on the problem of cohort selection for clinical trials.



Low participation in clinical trials is a significant problem in clinical and translational research, where participation rates range between 5%-10% for most trials [1]. A key deterrent to participation is that matching patients to clinical trials is currently a manual, physician-driven process. Automating this process has shown some promising results in terms of increased patient referrals from physicians [2]. However, current efforts at automation require the development of custom applications.

The SNOMED CT® ontology [3], which formally defines classes of disorders, drugs, and organisms, is well suited for our case study to see whether ontologies can help automate the problem of cohort selection. Our primary insight is that matching patients to clinical trials can be formulated as a problem of semantic retrieval, i.e., a clinical trial criterion can be expressed as a semantic query, which a reasoner can then use together with SNOMED CT to infer implicit information that results in retrieving eligible patients.

Our goal in this study is to assess the feasibility of this approach in a realistic scenario. The technical challenges fall primarily into three categories: knowledge engineering, scalability, and noisy data, each of which is described below.

*Knowledge Engineering.* A key challenge is to combine the legacy patient data with existing ontologies such as SNOMED CT to demonstrate the value of ontology matching for cohort selection. The following examples illustrate this problem:

- There are currently 39 clinical trials [4] that specify *warfarin medication* as an inclusion criterion. SNOMED CT has the names of generic drug concepts, which are in turn described in terms of their active ingredients, such as warfarin. However, the patient record contains only the names of vendor-specific drugs. What is needed here is a mapping from vendor-specific drug names to generic drug concepts, to allow an inference about active ingredients of drugs.
- There are 26 clinical trials that specify *Methicillin-resistant Staphylococcus aureus (MRSA) disorder* as an inclusion criterion for the trial. SNOMED CT defines *MRSA disorder* as a disorder that indicates the class of MRSA organisms as a causative agent. However, the patient record contains institution-specific laboratory tests that indicate only the presence or absence of a particular organism (e.g., MRSA organism) in institution-specific terminologies. What is needed here is a mapping of the presence or absence of the organism to whether its corresponding SNOMED CT equivalent term is a causative agent or not.
- There are currently 6240 trials that refer to disorders that involve different types of *neoplasms*. SNOMED CT classifies 1522 different types of morphologies as neoplasms. However, the patient record contains information about a specific radiology test that indicates the presence of a certain morphology in a certain body part, all coded in local terminologies. Once again local terms for body parts and morphologies need to be mapped to their SNOMED CT counterparts.

It is clear from these examples that a key knowledge engineering task is to map patient record terms to concepts in the SNOMED CT model. This mapping

process is not simply a matter of establishing equivalences, which is itself non-trivial for large terminologies. The local terminology is often coded as a taxonomy, so there is the additional difficult problem of ontology integration [5]. Because each health care institution codes patient data using an idiosyncratic local terminology, mapping to the SNOMED CT model requires customization per health care institution. Fortunately, while this task is a significant effort, it only has to be performed once per institution, and is reusable for solving different clinical problems.

*Scalability.* Another key challenge is the need for reasoning over ontologies that are very large and expressive. The size of the knowledge base for the clinical trials case study far exceeds the capabilities of most reasoners. There are several reasoners that are designed to handle large Tboxes (e.g., Fact++ [6], Pellet [7], Racer [8]). Other reasoners scale to large Aboxes in secondary storage (e.g. Kaon2 [9], SHER [10]). The combination of a large Abox and a large Tbox required for this case study, however, far exceeds the size of the knowledge bases that have been tested so far with these reasoners.

Another factor is the expressivity needed for solving the clinical problem. While SNOMED CT is modeled within the EL++ [11] formalism (intersections, existential restrictions, role hierarchies), negation and universal restrictions are inherent in the patient data and in the queries. As an example, negation of complex concepts is an important aspect of the patient record, e.g., when pneumonia has been ruled out on the basis of a radiology report. Similarly, clinical trials exclusion criteria are negations of complex concepts, which means that the solution requires the expressivity of OWL-DL.

*Noisy, incomplete data.* The third challenge is that clinical data tends to be incomplete and noisy. SNOMED-CT contains complete definitions for disorders including both information needed to infer the presence of the disease, and also information to relate the disorder to other disorders. However, patient data contains only information needed to infer the disease.

Clinical data is also inconsistent from a logical perspective. It is not uncommon for a laboratory test to contain both positive and negative findings. To perform semantic retrieval, current reasoners assume that the data is consistent. Therefore, cleansing the data efficiently is another open issue.

In the rest of this paper, we present our solutions to these technical challenges, and summarize the results for matching 9 clinical trial criteria against a knowledge base with 59 million Abox assertions and 22,561 Tbox assertions. The clinical trials case study is described in more detail in Section 2. Sections 3-5 present the technical challenges and issues that we faced, and how they were resolved. Section 6 gives results and validation, and Section 7 draws conclusions.

## 2 Case Study Description

The architecture for retrieving patients eligible for clinical trials is shown in Figure 1. Clinical trial criteria are formulated as queries, and a reasoner matches

the queries against a knowledge base to retrieve eligible patients. We use the SHER reasoner, which implements the techniques in ([12], [10]) for scalable Abox reasoning. The first steps in creating this solution are constructing a knowledge base Tbox, based on SNOMED CT, and an Abox from structured patient records. For our case study, we use one year of anonymized patient records from Columbia University Medical Center.

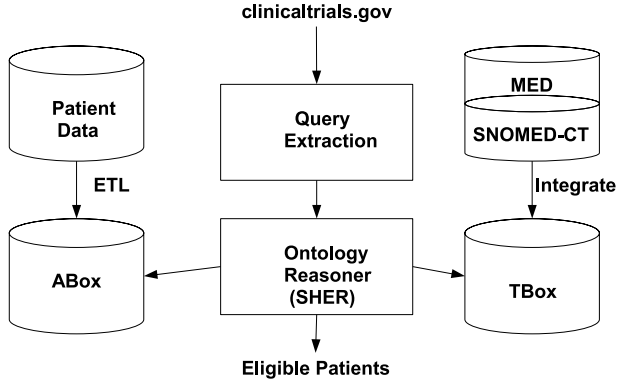


Fig. 1. Case Study Solution Architecture

Constructing the Tbox requires integrating the terminology used in the patient data and SNOMED CT terminology. The Columbia patient data are encoded in a frame-based semantic network called MED [13]. We considered only the MED taxonomy which consists of 100,212 concepts that capture the organism, disease, and medical test hierarchies. SNOMED CT has 379,630 concepts which include organism, pharmaceutical product, specimen, body structure, clinical findings, and procedures. SNOMED CT is not just a taxonomy; 217,619 of SNOMED CT concepts are defined in terms of existential restrictions. Such definitions allow the inferencing of disorders from relationships in the Abox such as associated morphology, finding site, and causative agents.

Constructing the Abox requires translating records encoded in the MED taxonomy into a set of assertions encoded in SNOMED CT in SHER's relational store. The patient database [14], which stores the raw data for the Abox, includes a single table of clinical events, where each event consists of one or two records. The events used in the case study correspond to laboratory test results, radiology findings, and drug treatment. We use an Extract-Transform-Load (ETL) process to transform the patient events into assertions compatible with SNOMED CT.

The queries themselves are extracted from clinical trial criteria found on [4], where the criteria are expressed as text. We convert the text-based queries into logical DL queries, which use SNOMED CT concepts. Sections 3-5 describe the major technical challenges encountered in implementing the case study.

### 3 Knowledge Engineering

#### 3.1 Mapping MED to SNOMED CT

To create the Tbox, the first step is to map concepts in MED to the concepts in SNOMED CT. Our goal is to achieve a high degree of accuracy and coverage through a semi-automated process:

1. **Existing Mappings:** Many of the concepts in both MED and SNOMED CT are mapped to the concepts in the Unified Medical Language System (UMLS®) [15]. Therefore, it is possible to use UMLS as an intermediary target, mapping MED to UMLS to SNOMED CT for a subset of MED concepts.
2. **NLP-based Mapping:** We next use the medical Meta Map tool (MMTx) [16] to map natural language strings associated with MED concepts to UMLS concepts, and then to SNOMED CT concepts when possible. Only mappings with a perfect score on MMTx are retained.
3. **Prefix Removal:** Some MED concept strings contain institution-specific prefixes, such as NYPH. We remove these prefixes to increase the number of perfect matches on MMTx.
4. **Manual mapping:** Vendor-specific drugs in MED do not have a mapping to a generic mapped drug concept in SNOMED CT; these 1000 concepts are manually mapped by domain experts (co-authors CP and JC).

This procedure maps 17,446 out of 100,212 MED concepts to SNOMED CT. The next step, described in Section 3.3, dramatically increases the coverage by including assertions corresponding to the MED taxonomy.

#### 3.2 Validation of Mappings

To determine the accuracy of mapping MED to SNOMED CT, domain experts (co-authors CP and JC) analyzed the mapping results. Since both MED and SNOMED CT share a common upper level ontology (Semantic Network in UMLS), it is possible to determine whether each of the mapped concepts belong to the same conceptual category. These categories were further combined into semantic groups. An example of a valid mapping is the MED concept *fibromyalgia*, which has an upper level concept in UMLS of *Finding*, mapped to the SNOMED CT concept *Primary fibromyalgia syndrome*, which has an upper level concept of *Disease or Disorder*. Both *Finding* and *Disorder* belong to the same semantic group. By this approach, 2,534 invalid mappings were found with mismatching source and target upper level concepts and semantic groups. Manual inspection of invalid mappings revealed that the majority (all but 11) are caused by errors in categorizing MED concepts in terms of the upper UMLS concepts, e.g. 768 are caused by a single missing parent type of *DRUG* in MED. The faulty 11 are true false positives, and are eliminated from our mapping. False negatives from unmapped concepts are discussed in the next section.

For each mapped MED concept, a subclass assertion is added to the Tbox to relate it to its mapped SNOMED concept. We use subclass rather than equivalence assertions because the current mapping between MED and SNOMED is not guaranteed to be sufficiently precise to warrant equivalence. However, without concept equivalence, negated queries fail, since we cannot infer that the negation of a MED concept is a subclass of the negation of its corresponding SNOMED CT concept. A more precise MED to SNOMED mapping will eliminate this issue, and this is an issue for future work.

### 3.3 Integrating the MED Taxonomy with SNOMED CT

Although we succeeded in mapping 17,446 MED concepts, this constitutes only 17% of the MED Tbox. In terms of the 13,313 MED concepts referred to in the Abox of one year patient data, only 9% had a direct mapping to a SNOMED CT concept. This reflects the fact that the patient data is coded in institution-specific MED concepts that do not have direct mappings to SNOMED CT concepts. However, since many of their super-concepts map to SNOMED CT concepts by our mapping process, we can significantly increase coverage by adding subclass assertions corresponding to the MED taxonomy. If we include the subclasses of the mapped MED concepts, we increase coverage of the MED Tbox to 75,514 concepts. For the Abox, including subclasses of mapped MED concepts increases coverage from 9% to 88% (11,732 concepts).

### 3.4 Abox Construction

To construct a SNOMED CT Abox from the one year patient data, we transform the existing relational patient database with implicit relationships into membership and role assertions corresponding to SNOMED CT. As an example of such a mapping, if a patient record states that the patient is on drug *Cerner Drug: Lactulose Syrp 20G/30ml*, it needs to be transformed into the appropriate SNOMED CT role assertion between the patient and the drug. We use the attribute *administeredSubstance* as the relationship and the drug itself is mapped to *Lactulose* in SNOMED CT in the Tbox.

Patient data transformation performs several critical functions:

- In the clinical domain, negative findings for medical tests and procedures are crucial in selection for clinical trials and clinical decision support. Therefore, negative results in the patient data should be modeled using logical negation. The transformation process extracts positive and negative results from the patient record and makes them explicit.
- In the clinical domain, results of laboratory tests and findings form logical groupings (e.g., a specific laboratory test indicates an organism as well as the source specimen for the test). Disorders in SNOMED CT capture such groupings by nesting existential restrictions as illustrated by the SNOMED CT definition of *Breast Neoplasm* below:

$$\exists \text{roleGroup}.(\exists \text{hasMorphology}. \text{Neoplasm} \sqcap \exists \text{hasFindingSite}. \text{Breast}).$$

We therefore model groups of events using the SNOMED CT *roleGroup* attribute, as discussed in the examples below.

The Abox construction process is driven off of set of transformation rules, derived by abstracting implicit information models for both the patient database and SNOMED CT. Fortunately, the structure of the data and these information models are relatively simple, so that the number of rules is small. Table 1 illustrates two radiology rules.

**Table 1.** Transformation Rules for Radiology Events

Radiology Event Template	Abox Assertion Templates
?PatientID, ?TimeStamp, ?Morphology, ?BodyPart, ?HighCertainty	individuals: $p, e, r$ $assocObservation(p, e)$ $roleGroup(e, r)$ $hasTimeStamp(e, ?TimeStamp)$ $r : \exists hasMorphology. ?Morphology$ $r : \exists findingSite. ?BodyPart$
?PatientID, ?TimeStamp, ?Morphology, ?LowCertainty	individuals: $p, e, r$ $assocObservation(p, e)$ $roleGroup(e, r)$ $hasTimeStamp(e, ?TimeStamp)$ $r : \forall hasMorphology. \neg ?Morphology$

These rules generate unique individuals  $p$  and  $e$  in the Abox, representing each unique patient and event. A unique individual  $r$  is generated to represent the grouping of the associated Morphology and BodyPart of an event. New relationship assertions are generated to associate  $p$  with  $e$ , and  $e$  with  $r$ .

The first rule transforms a positive morphology finding and associated body site, and the second rule transforms a negative morphology finding, in which case there is no associated body site. In the positive case, the first rule adds membership assertions with existential restriction concepts to the Abox, associating  $r$  with the morphology and the body site. In the negative case, the second rule adds a membership assertion with a universal restriction concept that includes negation to the Abox.

Table 2 shows examples of rule instantiation. The transformation rules are engineered to match SNOMED CT definitions. For example, the SNOMED CT definition of *Breast Neoplasm* above typifies SNOMED CT rules for radiology findings. Note that a query for patients testing positive for breast neoplasm will match the first patient in Table 2, and a query for patients testing negative will match the second patient.

## 4 Scalability

### 4.1 Dealing with Large Aboxes

SHER embodies techniques [10, 12] which use summarization and refinement to achieve scalable Abox reasoning. Specifically, a summary Abox is constructed

**Table 2.** Sample Radiology Event Transformations

Radiology Event	Abox Assertions
Patient43, 3.15.2006, Malignant Neoplasm, Breast, High Certainty	individuals: $p43, e1, r1$ $assocObservation(p43, e1)$ $roleGroup(e1, r1)$ $hasTimeStamp(e1, 3.15.2006)$ $r1 : \exists hasMorphology.Malignant\ Neoplasm$ $r1 : \exists findingSite.Breast$
Patient32, 12.01.2005, Malignant Neoplasm, Low Certainty	individuals: $p32, e2, r2$ $assocObservation(p32, e2)$ $roleGroup(e2, r2)$ $hasTimeStamp(e2, 12.01.2005)$ $r2 : \forall hasMorphology.\neg Malignant\ Neoplasm$

from the original Abox. The initial summary Abox is built by mapping all instances of the same type in the original Abox to a single instance in the summary Abox. For example, all instances of *Malignant Neoplasm* in the original Abox are represented by a single instance of *Malignant Neoplasm* in the summary Abox. SHER first checks the summary Abox for any inconsistencies in the knowledge base, using Pellet tableau-based reasoner [7] for consistency checking. If the summary is consistent, then the original Abox must be consistent (for technical detail, see [12]). However, the converse is not true. If any inconsistencies exist, then the reasoner finds their justifications (i.e., the minimal set of assertions responsible for the inconsistency), and tries to selectively refine summary instances in these justifications. Refinement is the process of splitting the summary instance by the sets of role assertions that are present in the original Abox for the individuals mapped to the given summary instance. This iterative process of refinement ends when the summary is consistent, or the justifications cannot be refined any more. If the knowledge base is inconsistent, SHER provides a set of justifications that can be used to cleanse the knowledge base of inconsistencies.

To answer a query, the negation of the query is added to the concept set of each instance in the summary Abox, and the same iterative refinement process is followed. During this process, a map from refined individuals in the summary Abox to individuals in the original Abox is maintained. When the process converges, query results are obtained from this mapping. Initially, SHER could not scale to the case study with such a large Abox. The problem was in the refinement step: the map from refined individuals to real individuals was kept in memory. To achieve scalability, the refinement mapping is now maintained in the database. In fact, the refinement process is performed entirely by database operations.

## 4.2 Dealing with Large Tboxes

Even though the MED-SNOMED CT integrated Tbox has a total of 523,368 subclass or equivalence assertions, we do not need all of these for reasoning. As

described in [12], the techniques used in SHER are based on taking the closure of the Abox, which informally is the set of concepts that are present in the Abox, either directly or indirectly through assertions in the Tbox. For query answering, the closure of the query concept must also be included.

More specifically, we compute a subset of the MED-SNOMED CT integrated Tbox using the following procedure: (a) We use the FACT++ [6] tableau reasoner to absorb the Tbox to produce a new set of Tbox assertions  $\mathcal{T}$  that eliminates any general concept inclusions from the original Tbox. In the case of the MED-SNOMED CT integrated Tbox, no general concept inclusions are left after absorption, and no domain or range constraints are added to the Rbox due to role absorption. (b) We then compute the closure of the Abox  $clos(\mathcal{A}, \mathcal{T}, \mathcal{R})$  and queries as defined in [12]. (c) For each concept  $C$  in the  $clos(\mathcal{A}, \mathcal{T}, \mathcal{R})$ , we add the assertions in  $\mathcal{T}$  where that concept appears on the left hand side of the assertion. The resulting Tbox has 22,561 assertions, of which 17,319 assertions are related to MED concepts.

## 5 Noisy, Incomplete Data

The data in the patient records is incomplete with respect to SNOMED CT definitions. For example, suppose a clinical trial criterion is Methicillin resistant Staphylococcus aureus infection (MRSA). The SNOMED CT definition of MRSA is the intersection of three terms:

*∃hasCausativeAgent.Methicillin resistant Staphylococcus aureus,*  
*Infection due to antimicrobial resistant bacteria,* and  
*Infection due to Staphylococcus aureus.*

If a patient record contains a positive test for an MRSA organism, a clinician would likely say the patient matches the eligibility criteria. However, the patient record matches only the first term of the SNOMED CT definition. Since there is not information in the patient record that matches the second two conjuncts, which provide definition completeness, the patient will not be retrieved as eligible. We therefore support users specifying which terms of the definition are required, allowing them to tailor the query to match the data that they have. We refer to this as *query weakening*. The patients retrieved are then ranked based on the number of matching terms.

As discussed earlier, clinical data is inconsistent. As an example, two different laboratory tests for the same disorder can result in contradictory results. SHER is designed to detect multiple inconsistencies in the data efficiently (for technical details, see [12]). We use these algorithms to eliminate inconsistent data before querying it.

## 6 Evaluation

In this section, give the experimental results of our case study. Our experiments were conducted on a 2-way 2.4GHz AMD Dual Core Opteron system with 16GB



of memory running Linux, and we used IBM DB2 V9.1 as our database. Our Java processes were given a maximum heap size of 8GB.

### 6.1 Validation with a 100 Patient Dataset

To validate the clinical correctness of results we first performed an experiment with a randomly selected dataset of 100 patients from a 20 year clinical dataset from the Columbia Medical Center. The 100 patient dataset has 7,451 Tbox subclass assertions, 98,956 type assertions, and 119,206 role assertions.

We selected 9 clinical trials from [4] that query for different types of clinical information (see Table 3). These queries were chosen to cover the domains of laboratory, drug and radiology data. Table 4 shows the DL version of the queries, along with the concepts that were weakened to find solutions, because of the partial information present in the clinical record. The order of the queries Table 4 reflects the order in Table 3. Table 5 shows the queries, the number of patients matched to the queries, whether matches reflect matches to weakened queries, and time to process the queries in seconds. For query *NCT00001162*, the results shown are for the union of 7 different disorders, only 4 of which are illustrated in Table 4.

The matched patients for the 9 clinical trials were manually evaluated by an analysis of the original Columbia database records by one of the authors (CP). Such an analysis revealed no false positives in the reported matches. In terms of recall, we missed 8 patients on the steroid/corticosteroid queries because the manual mapping of drugs to SNOMED CT missed these mappings. We missed 1 patient for the Metronidazole case. Here, the miss occurred because there were duplicate MED concepts Metronidazole and Metronidazole Preparations, with only the former concept being mapped to SNOMED CT. The missed patient for Metronidazole was because some drugs such as *Cerner Drug: Metronidazole Tab 500 mg.* were subclasses of the unmapped Metronidazole concept. For the breast neoplasm query, our transformation process did not distinguish between

**Table 3.** Clinical Trial Requirements Evaluated

ClinicalTrials.gov ID	Description
<i>NCT00084266</i>	Patients with MRSA
<i>NCT00288808</i>	Patients on warfarin
<i>NCT00393341</i>	Patients with breast neoplasm
<i>NCT00419978</i>	Patients with colon neoplasm
<i>NCT00304382</i>	Patients with pneumococcal pneumonia where source specimen is blood or sputum
<i>NCT00304889</i>	Patients on metronidazole
<i>NCT00001162</i>	Patients with acute amebiasis, giardiasis, cyclosporiasis or strongloides...
<i>NCT00298870</i>	Patients on steroids or cyclosporine
<i>NCT00419068</i>	Patients on corticosteroid or cytotoxic agent

**Table 4.** DL Queries for Evaluated Clinical Trials

DL Query	Weakened Concept
$\exists associatedObservation.MRSA$	MRSA
$\exists associatedObservation.$ $\exists roleGroup.$ $\exists administeredSubstance.$ $\exists roleGroup.\exists hasActiveIngredient.Warfarin$	None
$\exists associatedObservation.BreastNeoplasm$	Breast Neoplasm
$\exists associatedObservation.ColonNeoplasm$	Colon Neoplasm
$\exists associatedObservation.$ $\left( \begin{array}{l} PneumococcalPneumonia \\ \sqcap \\ \exists hasSpecimenSource.Blood \sqcup Sputum \end{array} \right)$	Pneumococcal Pneumonia
$\exists associatedObservation.$ $\exists roleGroup.$ $\exists administeredSubstance.$ $\exists roleGroup.\exists hasActiveIngredient.Metronidazole$	None
$\exists associatedObservation.$ $\left( \begin{array}{l} acuteamebiasis \sqcup \\ giardiasis \sqcup \\ cyclosporiasis \sqcup \\ strongloides \sqcup \\ \dots \end{array} \right)$	acute amebi- asis giardiasis cyclosporiasis strongloides ...
$\exists associatedObservation.$ $\exists roleGroup.$ $\exists administeredSubstance.$ $\exists roleGroup.\exists hasActiveIngredient.cyclosporine \sqcup steroids$	None
$\exists associatedObservation.$ $\exists roleGroup.$ $\exists administeredSubstance.$ $\exists roleGroup.\exists hasActiveIngredient.corticosteroid \sqcup cytotoxicAgent$	None

disorders and imaging findings in the radiology data, hence the relevant MED concept *Malignant Neoplasm of Breast (Female) Unspecified* was asserted as a finding. We rectified the problem by including a query extension that also looks at the associated findings:

$\exists associatedObservation.\exists associatedFinding.BreastNeoplasm.$

## 6.2 Results with the 1 Year Dataset

The 1 year patient dataset had records for 240,269 patients with 22,561 Tbox subclass assertions, 26 million type assertions, and 33 million role assertions. In the 1 year patient dataset, we had 15 instances of inconsistencies in the data. These inconsistencies were due to (a) laboratory tests that produced contradictory information, for example, positive and negative assertions about organism respiratory syncytial virus by laboratory tests of direct immunofluorescence assay (DFA) and enzyme immunoassay (EIA), (b) modeling errors in MED that resulted in certain MED concepts that were classified as both negative and positive information, for example, the MED concept *Rule Out Specific Organism* was an indirect subclass of both *Positive Organism Comment Result* and *Negative*

**Table 5.** Patient Matches for Trial DL Queries for 100 Patients

Query	Matched Patients	Time (s)	Weakened Query
<i>NCT00084266</i>	1	54	yes
<i>NCT00288808</i>	4	78	no
<i>NCT00393341</i>	0	29	yes
<i>NCT00419978</i>	1	51	yes
<i>NCT00304382</i>	0	39	yes
<i>NCT00304889</i>	0	29	no
<i>NCT00001162</i>	4	225	no
<i>NCT00298870</i>	6	117	no
<i>NCT00419068</i>	6	118	no

*Culture Result.* Since MED is a taxonomy, and does not contain an assertion that *Positive Organism Comment Result* is disjoint with *Negative Culture Result*, this inconsistency was only found when we transformed the Abox to contain assertions about the presence or absence of an organism based on these concepts. The inconsistent data were detected by SHER, and we manually deleted records that resulted in the inconsistencies.

Table 6 shows the queries, the number of patients matched to the queries, the time to process the queries in minutes, and whether the query needed to be weakened to find solutions. Table 6 demonstrates the scalability of reasoning in the SHER engine for a combination of a large Tbox and a large Abox. We do not present any comparison results because no other reasoner we know of can query this dataset. For the use of clinical trial matching, which is currently a manual process, our results show that using ontology matching to automate this task is practical.

## 7 Discussion

We have presented a feasibility study for an ontology-based approach to match patient records to clinical trials. Using a real world patient dataset, we described various modeling and engineering challenges that we solved, including:

- Mapping the MED terminology to SNOMED CT terminology.
- Integrating the MED taxonomy with SNOMED CT.
- Transforming the 1-year patient database into a SNOMED CT Abox.
- Reasoning over a realistic dataset.
- Identifying and eliminating noise in the patient data.
- Dealing with incomplete patient information.

We are continuing to work on making it easier to integrate large ontologies, improving the integration of MED and SNOMED CT, and tuning SHER for scalability.

An interesting problem with respect to clinical data and clinical trials queries is that of open versus closed world reasoning. Description logics and OWL use

**Table 6.** Patient Matches for Trial DL Queries for 240,269 Patients

Query	Matched Patients	Time (m)	Weakened Query
<i>NCT00084266</i>	1018	68.9	yes
<i>NCT00288808</i>	3127	63.8	no
<i>NCT00393341</i>	74	26.4	yes
<i>NCT00419978</i>	164	31.8	yes
<i>NCT00304382</i>	107	56.4	yes
<i>NCT00304889</i>	2	61.4	no
<i>NCT00001162</i>	1357	370.8	no
<i>NCT00298870</i>	5555	145.5	no
<i>NCT00419068</i>	4794	78.8	no

an open world assumption i.e. if a fact is not explicitly asserted, no assumption is made about the fact, as opposed to a closed world assumption which assumes a fact is negative if not explicitly asserted. In the clinical domain, we need open world reasoning in radiology and laboratory data, because, for example, unless a lab test asserts a negative finding we cannot make arbitrary assumptions about the results. However, in pharmacy data, we can use the closed world assumption to infer that a patient is not on a medication if it is not asserted. Integrating open world with closed world reasoning is a key issue for future consideration [17].

SNOMED CT plays a critical role in the clinical domain; it has been adopted as a national health care standard in the United States and was recently acquired by International Health Terminology Standards Development Organization thereby making it a truly global clinical standard in healthcare. Representing patient data using SNOMED CT has benefits that go beyond the clinical trials matching application. Currently, several decision support systems, infection control systems, public health organizations and regional healthcare information organizations use SNOMED CT merely for terminology services. Our approach provides a means to reuse the knowledge already represented in SNOMED CT to perform semantic retrieval for different biomedical applications.

## Acknowledgements

We gratefully acknowledge the help that Bishwaranjan Bhattacharjee provided in tuning our database queries.

## References

1. Simes, R.: Clinical trials and real-world medicine. Trial evidence best informs real-world medicine when it is relevant to the clinical problem. *Med. J.Aust.* 177(8), 410–411 (2002)
2. Embi, P., Jain, A., Clark, J., Bizjack, S., Hornung, R., Harris, C.: Effect of a clinical trial alert system on physician participation in trial recruitment. *Arch.Intern.Med.* 165(19), 2272–2277 (2005)

3. SNOMED-CT, <http://www.snomed.org>
4. Clinical Trials, <http://clinicaltrials.gov/>
5. Noy, N.F.: Semantic integration: a survey of ontology-based approaches
6. Horrocks, I.: Using an expressive descriptive logic: fact or fiction? In: KR 1998. Proceedings of the 6th Int. Conf. on principles of Knowledge Representation and Reasoning, pp. 636–647 (1998)
7. Sirin, E., Parsia, B.: Pellet: An owl dl reasoner. In: Description Logics (2004)
8. Haarslev, V., Moller, R.: Racer system description. In: Goré, R.P., Leitsch, A., Nipkow, T. (eds.) IJCAR 2001. LNCS (LNAI), vol. 2083, pp. 701–705. Springer, Heidelberg (2001)
9. Hustadt, U., Motik, B., Sattler, U.: Reducing shiq description logic to disjunctive datalog programs. In: KR 2004. Proc. of 9th Intl. Conf. on Knowledge Representation and Reasoning, pp. 152–162 (2004)
10. Dolby, J., Fokoue, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K.: Scalable semantic retrieval through summarization and refinement. In: Proc. of the AAAI Conf. (2007)
11. Baader, F.: Brandt, S., Lutz, C.: Pushing the el envelope. Technical report, Chair of Automata Theory, Institute for Theoretical Computer Science, Dresden University of Technology (2005)
12. Fokoue, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K.: The summary abox: cutting ontologies down to size. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 136–145. Springer, Heidelberg (2006)
13. Cimino, J., Clayton, P., Hripcsak, G., Johnson, S.: Knowledge-based approaches to the maintenance of a large controlled medical terminology. *J. Am Med. Inform. Assoc.* 1(1), 35–50 (1994)
14. Johnson, S.: Generic data modeling for clinical repositories. *J. Am Med. Inform. Assoc.* 3(5), 328–339 (1996)
15. DA, D.L., Humphreys, B., McCray, A.: The unified medical language system. *Methods Inf. Med.* 32(4), 281–291 (1993)
16. MMTx, <http://mmtx.nlm.nih.gov/>
17. Grimm, S., Motik, B.: Closed world reasoning in the semantic web through episodic operators. In: OWLED 2005. Proc. of the Workshop on OWL: Experiences and Directions (2005)

# Application of Ontology Translation

James Ressler<sup>1</sup>, Mike Dean<sup>2</sup>, Edward Benson<sup>2</sup>, Eric Dorner<sup>1</sup>, and Chuck Morris<sup>1</sup>

<sup>1</sup> Northrop Grumman Corporation

1010 Market Street, Suite 1740, St. Louis, Missouri, 63101

james.ressler@ngc.com, eric.dorner@ngc.com, chuck.morris@ngc.com

<sup>2</sup> BBN Technologies

1300 N. 17<sup>th</sup> Street, Suite 400, Arlington, Virginia, 22209

mdean@bbn.com, ebenson@bbn.com

**Abstract.** An ontology provides a precise specification of the vocabulary used by a community of interest (COI). Multiple communities of interest may describe the same concept using the same or different terms. When such communities interact, ontology alignment and translation is required. This is typically a time consuming process. This paper describes Snoggle, an open source tool designed to ease development of ontology translation rules, and discusses its application to geospatial ontologies.

**Keywords:** ontology, alignment, translation, SWRL, geospatial, Snoggle.

## 1 Introduction

An ontology is a valuable resource for making sense of data content because it places information in context by using a formal vocabulary to describe the information. Applied to semantic reasoning, a query against instances of an ontology provides the ability to arrive at relevant responses to otherwise complex questions. However, when data from multiple ontologies are queried together, the context can be lost when the ontologies lack uniformity, i.e. sources written with different vocabularies do not lend themselves to machine-intelligible reasoning. The goal of data interoperability can be both advanced and hindered by the use of ontologies. By providing context to data, interoperability is achieved through the relations expressed between data elements. However, if the relations do not rely upon the same vocabulary, connections between data cannot be inferred. Similar to human reasoning, when a person provides information using a different vocabulary, humans mentally build a translation from terms that are known to the new terms that are provided. A mental translation between similar terms allows experts in related fields to communicate. The same type of translation is needed to make ontologies useful across multiple domains in ontologies.

The mapping of two knowledge representations requires precision to clearly define synonymous concepts and the conditions in which those concept equivalencies and other relationships are valid. A capable mapping tool must also be able to express logical, string manipulation and mathematical functions as part of its rules. The tool requires the ability to filter the rules and quantify the extent of mapping in order to determine coverage. Once a mapping is defined, the rules used to implement the

mapping between the two ontologies become part of the knowledge base and can be used to process queries.

Multiple ontologies arise naturally as different communities of interest address overlapping problem domains. Alignment of these ontologies is required to integrate data from their corresponding data sources. Mapping rules are the product of such alignment.

This paper presents research into a variety of geospatial ontologies, the alignment process and presents a graphical mapping tool based on that research that seeks to simplify the alignment process. We address the topics in the following organization:

- Section 2 summarizes a case study and overview performed among multiple geospatial semantic vocabularies. This study highlights the need to better understand and support the ontology alignment process.
- Section 3 discusses the problem of mapping and alignment as applied to domain ontologies and presents prior work in this area.
- Section 4 presents a graphical mapping language and supporting tool called Snoggle<sup>1</sup> that addresses the ontology mapping problem in an intuitive, visual way.
- Section 5 demonstrates an application of this tool to alignment of geospatial vocabularies from the case study.

## 2 Geospatial Semantic Vocabularies

The W3C Geospatial Incubator Group<sup>2</sup> has defined seven initial categories of basic geospatial ontologies: features, feature types, spatial relationships, coordinate reference systems, toponyms (place names), geospatial metadata, and web services. The authors have recommended four additional categories: geometric, coverage, geopolitical, and temporal. A wide range of existing semantic vocabularies cover most of these categories of geospatial ontologies. Selecting an ontology for a geospatial dataset is based upon many factors, including authorship of the data, required expressivity, user preferences, other data sources to be integrated, and the tools to be used. A study of existing geospatial ontologies [14] categorized the ontologies as shown in Figure 1, where “F” indicates the ontology fully applies to the category and “P” indicates the ontology partially applies.

As this figure shows, many ontologies cover the majority of the core spatial domains. The Cyc, ISO and NGA based ontologies are most prevalent in the geospatial domain, while other niche models possess unique and useful roles in middle and domain specific ontologies. This diversity demonstrates the need for ontology alignment.

### 2.1 Maritime Case Study

The rest of this paper considers two specific domain ontologies targeted at different communities of interest. The domain ontologies draw classes from several of the

<sup>1</sup> <http://snoggle.projects.semwebcentral.org/>

<sup>2</sup> <http://www.w3.org/2005/Incubator/geo/Wiki/FrontPage>

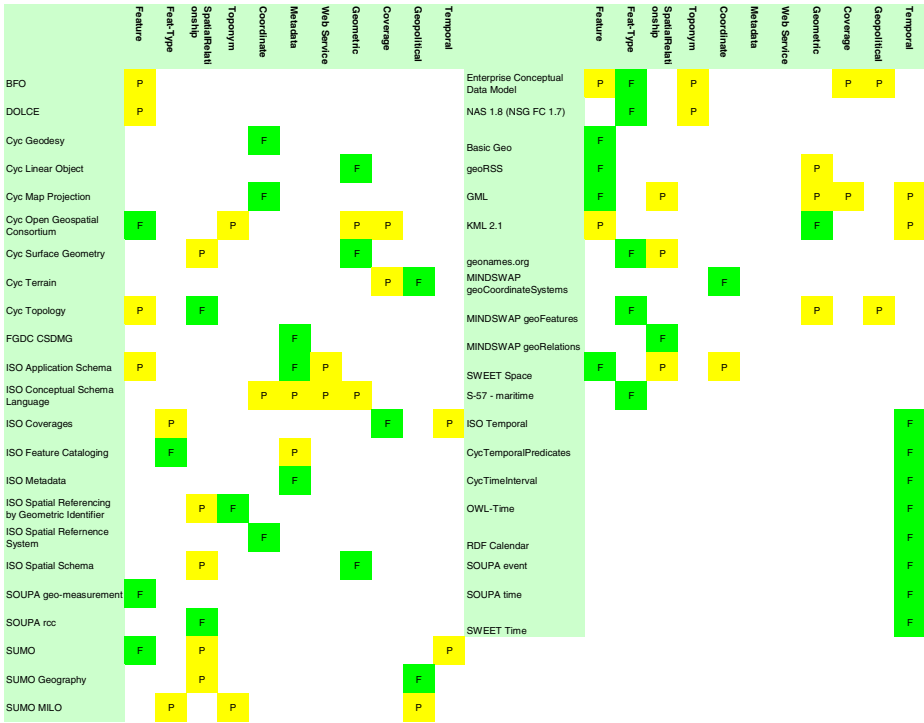


Fig. 1. Ontology Coverage of Core Geospatial Categories

ontologies referred to earlier, including those in the Features, Feature Types, Coordinates, and Geometry categories. The Digital Nautical Chart® (DNC®) is produced by the US National Geospatial-Intelligence Agency (NGA) and is a vector-based, digital database containing significant nautical features essential for safe maritime navigation. DNC® is based on the MIL-PRF-89023 specification [11]. The Electronic Navigational Chart (ENC) is produced by the US National Oceanic and Atmospheric Administration (NOAA) and supports the marine transportation infrastructure and coastal management. NOAA ENCs use the International Hydrographic Office (IHO) S-57 exchange format [3]. To illustrate the application of multiple data sets using similar ontologies, these two maritime product ontologies were mapped. The S-57 ontologies were created by Malyankar during his work on maritime text markup [8] and the DNC® ontology was created for this paper from MIL-PRF-89023. For example, classes exist for bridges, berths, mooring facilities, and pontoons in ENC which are equivalent to bridges, berths, and piers (equivalent to pontoons) in the DNC®. Their associated properties are also structurally equivalent but named differently. By mapping these two ontologies, queries can be made using either with equivalent results. The results of a mapping from ENC to DNC® are shown in Figure 4 on page 9.



### 3 Ontology Mapping and Alignment

Kolas [6] describes a frequently used Semantic Web data integration design pattern. A user domain ontology is defined by subject matter experts and used to express queries within a community of interest. This ontology is mapped to one or more data source ontologies. When it's sufficiently complete, a data source ontology may also be used as the domain ontology. Queries expressed in a domain ontology are translated into queries on the individual sources and the results are translated back into the domain ontology. We applied this design to two maritime data sources as described in Section 5.

#### 3.1 SWRL Rules

Ontology mapping can be viewed at a practical level as an applied case of the larger field of rule creation. Mapping between OWL ontologies is typically done using SWRL [13], the Semantic Web Rule Language, which combines OWL and RuleML<sup>3</sup> concepts.

SWRL rules are an implication between an antecedent and a consequent, both comprised of multiple atoms. When the atoms in the antecedent hold, then the conditions in the consequent also hold. Atoms may be of various types that encompass the possible entities and relationships expressible in OWL. Specifically, atoms can take the form  $C(x)$ ,  $P(x,z)$ ,  $\text{sameAs}(x,y)$ , or  $\text{differentFrom}(x,y)$ , where  $x$  and  $y$  are either variables or OWL individuals,  $z$  is a variable, OWL individual, or an OWL data value,  $C$  is an OWL class, and  $P$  is an OWL property.

Mapping from a source ontology to a target ontology using SWRL is thus accomplished by partitioning the source ontology into atomic components, using these components as the antecedents in a set of rules, and placing the equivalent structures in the target ontology as the consequents of such rules. If a source: `SalariedEmployee` in the source ontology maps to an target: `Employee` in the target ontology, the following rule would accomplish this mapping:

$$\text{source:SalariedEmployee}(\text{var}) \Rightarrow \text{target:Employee}(\text{var})$$

Additional atoms or rules would be used to map the properties associated with each class.

### 4 A Graphical Mapping Tool

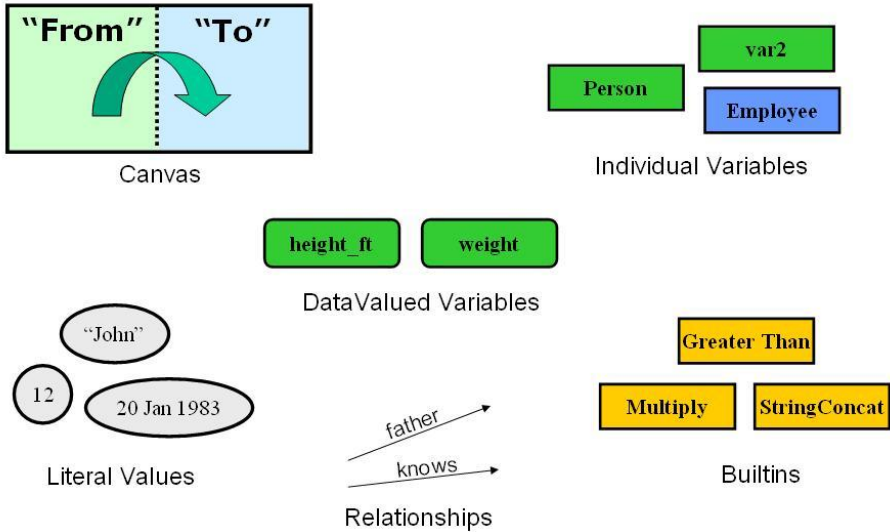
Previous literature on the subject of alignment and the maritime case study presented in Section 5 make it clear that mapping rules are critical to the ontology alignment process, but few tools exist to assist in the creation of such rules. Users are faced with the choice of general-purpose rule editors or of lengthy hand-coded XML files. To ease the translation process, we designed and developed Snoggle, a graphical ontology mapping environment.

---

<sup>3</sup> <http://www.ruleml.org/>

*Visual Vocabulary*

Central to Snoggle is the idea of a visual language created specifically for mapping. This language is designed for the general problem of mapping OWL-based data and not toward rule creation or any one particular rule language. Snoggle currently exports its results as SWRL, but can support exporters for other rule languages. Snoggle’s visual language is depicted in Figure 2.



**Fig. 2.** Visual Mapping Language Elements

Snoggle defines a mapping rule as consisting of two graphs: the source graph and the target graph. Using the visual language depicted in Figure 2, users draw structures using the source ontology in the source graph and a corresponding structure using the target ontology in the target graph. The user then “maps” the source graph to the target graph by drawing arrows between corresponding nodes. To accommodate unit conversions and other calculations, the source graph can also include nodes representing standard or user-defined SWRL builtin functions.

*Representing Variables, Literals, and Comments*

Variables in Snoggle are represented by rectangles (referred to as nodes) on the canvas. When creating a variable, the user must explicitly choose either a SWRL data variable or a SWRL individual variable. Data variables are displayed with rounded corners and individual variables are shown with hard corners, and the background shading on a variable node is dependent on the ontology (source or target) from which it came. Literal values are displayed within ovals and comments are displayed in pastel-colored rectangles.

*Mapping Rules*

Within the canvas, every node in the source (antecedent) region that corresponds to a node in the target (consequent) region must be connected via a map. A map is simply

a boldface arrow which is drawn between a node in the source region and a corresponding node in the target region.

As a user builds rules, graphical depictions of the source and target ontologies incrementally update themselves to show which aspects of the two ontologies remain unmapped. This provides a quick means of assessing coverage and tracking workflow. At any time, Snoggle allows the user to export the current mapping as a set of SWRL rules, or rules of some other language supported by an exporter.

## 4.1 Snoggle Design

Snoggle is conceptually divided into four principal user-facing components: the workspace, the canvas, the ontology browser, and the editors. Each of these presents a particular view of the rule development process and together they coordinate all of Snoggle's user-directed functionality. In addition to these components, a SWRL Model and Canvas Model play important roles in transforming graphs on the screen into mapping rules.

### Workspace

The Snoggle workspace is the container for all other Snoggle operations. It handles the loading and unloading of Snoggle project files, which contain all the information about a particular mapping task. Mapping from one ontology to another generally requires a large set of rules, depending on the size of the concerned ontologies, so projects are intended to contain multiple rules at the same time.

Each project file contains the following information:

- References to the source and target ontologies
- A set of prefixes for defined namespaces
- Any custom builtins loaded into the project
- The mapping rules created for the project

Each component of the Snoggle workspace shown in Figure 3 is described separately.

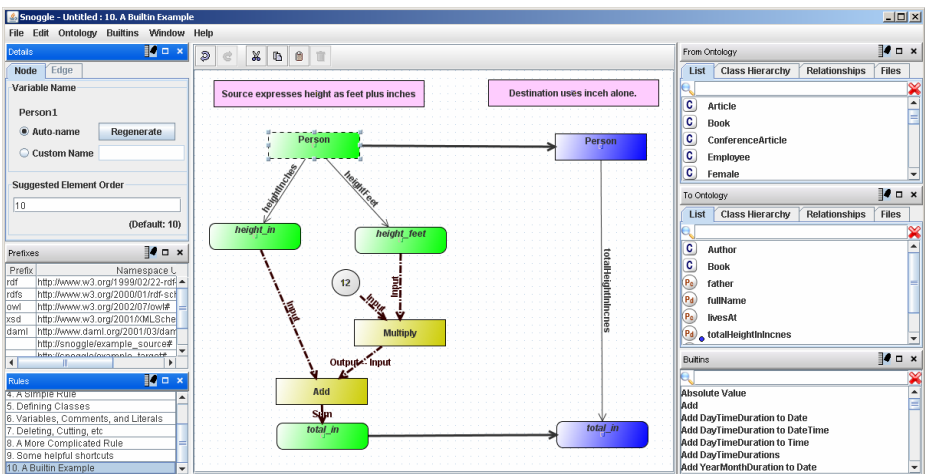


Fig. 3. Snoggle Workspace

### *Ontology Browser*

The Ontology Browser is a container for viewing any ontology loaded into a project. It provides three ways to visualize these ontologies: a flat list of classes and properties, a tree hierarchy of classes, and a bi-directional tree depicting the relationships between properties and classes. The most common mode of ontology mapping uses two separate ontology browsers: a “From” browser and a “To” browser. As objects in the “From” and “To” ontology are used to create rules for a project, users have the option of having them automatically hidden from the Ontology Browser’s view, thus providing a running list of everything that remains to be mapped in a particular Snoggle project.

### *Canvas*

Snoggle’s Canvas provides a graphical space on which to draw mapping rules using the objects loaded into the Ontology Browsers. The Canvas is divided into two special areas – the left side contains all entities and relationships that form the antecedent of the mapping rule, and the right side contains those that appear in the consequent. The left side represents the source ontology’s structure and the right side contains the equivalent structure in the target ontology. Nodes in the antecedent are displayed in green, and nodes in the consequent are displayed in blue.

Using the visual language, users drag objects from the Ontology Browser onto the Canvas and connect them with various types of arcs. Class objects dragged from the Ontology Browser become typed variable nodes; property objects can be dragged onto edges between nodes. Builtin functions are represented with a special color node, with incoming and outgoing edges mapped to the arguments of the builtin.

### *Builtins Viewer*

The Builtins Viewer contains a list of builtin functions that can be used in the project. All of the builtin functions defined within the SWRL specification are included. Custom Builtins can also be added by loading a file containing function definitions. Builtins are represented by red rectangles with hard corners.

### *Rule Editor*

The Rule Editor keeps track of the various rules within a project, and manages the loading and unloading of those rules onto the Canvas.

### *Namespace Editor*

The Namespace Editor manages all namespaces loaded into the project and handles the automatic prefixing of displayed information.

### *Object Editor*

The Object Editor provides information and settings for the object on the Canvas that currently has focus. The available settings depend on the particular type of object selected, such as variable nodes to contain class and variable information. Arcs denote input to a builtin that allows the user to specify which of the builtin’s inputs are represented by the source of the arc.

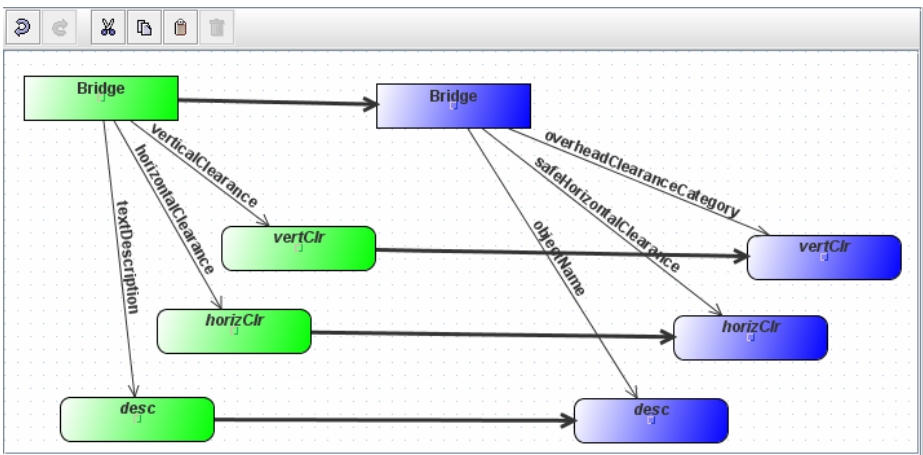
An illustration of the Canvas defining a rule between two definitions of the class Person is shown in Figure 3. Note the use of the Add and Multiply builtin functions to convert different representations of height.

## 5 Application to a Maritime Query

We used Snoggle to integrate the heterogeneous maritime ontologies introduced in Section 2 to answer a maritime query that neither data source could answer alone: “Which bridges in Boston Harbor can ship  $x$  navigate through?” In other words, given ship  $x$  with an aircraft of  $y$ , find all bridges where the height of the bridge  $h > y$ . Digital Nautical Chart® (DNC®) and Electronic Nautical Chart (ENC) are the maritime infrastructure data sources used. A data source containing information associated with ship  $x$  was also used.

DNC® and ENC contain similar maritime infrastructure objects such as bridges, piers, and buoys, and an ontology was generated for each data source based on their respective specifications. However, since both data sources are based on different specifications, their associated properties are also equivalent but named differently. For instance, both ontologies have a Bridge class and corresponding properties associated with the class. However, the property corresponding to the distance from the bridge span to the surface of the water is different in the two ontologies. The DNC® property is named `overheadClearanceCategory` while the ENC property is named `verticalClearance`. In addition to semantic differences, the instance data itself varies between the two data sources. For example, the DNC® data source contains bridges that are not present in the ENC data source and vice versa.

The mapping between corresponding Bridge classes was constructed using Snoggle as illustrated in Figure 4.



**Fig. 4.** Mapping of Bridge classes from ENC to DNC®

The resulting SWRL rule created by Snoggle for the bridge mapping is as follows (using a more accessible presentation syntax – the actual SWRL/XML syntax appears in the Appendix) :

```

enc:Bridge(?Bridge1) &
enc:textDescription(?Bridge1, ?desc) &
enc:horizontalClearance(?Bridge1, ?horizClr) &
enc:verticalClearance(?Bridge1, ?vertClr)
=>
dnc:Bridge(?Bridge1) &
dnc:objectName(?Bridge1, ?desc) &
dnc:safeHorizontalClearance(?Bridge1, ?horizClr) &
dnc:overheadClearanceCategory(?Bridge1, ?vertClr)

```

This SWRL rule maps the verticalClearance attribute of the ENC ontology to the overheadClearanceCategory property of the DNC® ontology. The mapping will integrate the two datasets and allow the merged result to be queried using the overheadClearanceCategory property.

The SWRL mapping rules were translated into Jena rules using the SweetRules<sup>4</sup> package within a maritime application interface. The application then executed the Jena rules using the Jena rule engine. This resulted in the addition of overheadClearanceCategory statements to the ENC RDF instance data. The SPARQL query shown below was executed and returned all DNC® and ENC bridge instances that ship x (in this case, a Tennessee class battleship) could navigate under. The returned geometries were then converted to KML and the results were displayed in Google Earth.

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-
ns#>
PREFIX dnc:
<http://ww4.geoenterpriselab.com/ont/dnc/dncsubset.owl#
>
PREFIX class:
<http://ww4.geoenterpriselab.com/ont/watercraft/watercr
aft_1.0.owl?object=Class#>

SELECT ?bridge ?ohc ?geom
WHERE {
    ?bridge rdf:type dnc:Bridge .
    ?bridge dnc:overheadClearanceCategory ?ohc .
    ?bridge dnc:coordinates ?geom .
    ?bridge dnc:objectName ?nam .
    <http://ww4.geoenterpriselab.com/rdf/watercraft.rdf?obj
ect=Class/Military/Battleship/Tennessee#Object>
    class:airDraft ?airdraft .
    FILTER (?airdraft < ?ohc)
}

```

In conclusion, the mapping between the DNC® and ENC ontologies integrates both a military and a commercial maritime data source. Since the data contained within

<sup>4</sup> <http://sweetrules.projects.semwebcentral.org/>

these sources is not identical, the mapping allows both of the sources to be used to paint a more complete picture of the maritime infrastructure.

## 6 Related Work

We use the term ontology translation to refer to a three phase process: 1) identifying related ontologies, 2) aligning related classes within them, and 3) creating executable mappings between those classes. Various subsets of this process have been addressed by other researchers in the ontology and database communities under a variety of names including ontology/schema alignment, matching, merging, mapping, and articulation. Kalfoglou [5] and Noy [12] surveyed such work; links to more recent work are available at [ontologymatching.org](http://ontologymatching.org). Kotis [7] defines the ontology merging problem to be:

Given two ontologies find an alignment between these two ontologies, and then, get the minimal union of their translated vocabularies and axioms with respect to their alignment.

The alignment and merging of ontologies has been addressed by tools such as the INRIA alignment tool [1], the HCONE approach to ontology merging [7] and PROMPT [12], a merging plug-in to Protégé. Many alignment tools place some measure of confidence or rank order on the suggested mappings between concepts. All of these tools can align with some degree of accuracy, and are usually between 30% and 40% effective at precise matching [2]. It can be assumed that human reasoning will be required to clarify the relations between concepts. Snoggle is intended to capture human-defined mappings in a graphical representation.

While previous work has focused on the first two phases, Snoggle currently covers the third phase of the ontology translation process to create executable mappings. We would like to extend it to address earlier phases, through the use of INRIA's Alignment API<sup>5</sup> [1] which is already supported by many matching tools.

Snoggle differs from other SWRL rule editors such as Protégé [13], the Object Rule Editor (ORE) [10], and RuleVISor [9] in supporting a graph-based visual paradigm and in focusing on ontology translation through the use of multiple ontology browsers, a divided Canvas, and coverage indicators.

## 7 Conclusions

This paper has motivated the need for ontology alignment and shown the utility of creating mapping rules between data source and domain ontologies. We highlighted this need through a case study involving multiple overlapping but incomplete maritime ontologies and data sets.

Snoggle is a new tool that provides an easy method to visually define SWRL mappings between OWL ontologies using a simple workflow. It defines an intuitive visual language to describe the mapping metaphor and permits use of that language

---

<sup>5</sup> INRIA's alignment tool is also capable of exporting OWL and SWRL.

through a drag-and-drop interface using visualizations of source and target ontology and the mappings between them.

We demonstrated Snoggle in practical use by aligning elements of the maritime ontologies from the case study so that a query could be performed across them that would not be possible with either of the ontologies alone. This straightforward mapping could easily be expanded to more complicated operations using Snoggle's builtin support, encompassing operations ranging from unit conversions to spatiotemporal processing.

**Acknowledgments.** The authors express their appreciation to the National Geospatial-Intelligence Agency (NGA) for support in developing Snoggle and preparing this paper.

## References

1. Euzenat, J.: An API for ontology alignment (version 2.1), INRIA Rhone-Alpes (2006)
2. Euzenat, J., et al.: Results of the Ontology Alignment Evaluation Initiative 2006. In: Ontology Alignment Evaluation Initiative (OAEI). Proc. 1st International Workshop on Ontology Matching at the 5th International Semantic Web Conference, Athens, GA (November 2006)
3. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosz, B., Dean, M.: SWRL: A Semantic Web Rule Language Combining OWL and RuleML, W3C Member (submission, May 21, 2004)
4. International Hydrographic Organization, IHO Transfer Standard for Digital Hydrographic Data, S-57 Edition 3.1, Nov (2000)
5. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. *The Knowledge Engineering Review* 18(1) (2003)
6. Kolas, D., Dean, M., Hebel, J.: Geospatial Semantic Web: Architecture of Ontologie. In: Proc. IEEE Aerospace Conference, Big Sky, MT (March 2006)
7. Kotis, K., Vouros, G.: The HCONE Approach to Ontology Merging. In: Proc. First European Semantic Web Symposium, Heraklion, Greece (May 2004)
8. Malyankar, R.: Elements of Semantic Web Infrastructure for Maritime Information. In: National Technical Meeting for the Institute of Navigation, San Diego, CA (2002)
9. Matheus, C.: SAWA: An Assistant for Higher-Level Fusion and Situational Awareness. In: Proc. SPIE Conference on Multisensor, Multisource Information Fusion., Orlando, FL (March 2005)
10. Munoz, A., Vera, A., Botia, J., Gomez Skarmeta, A.: Defining Basic Behaviors in Ambient Intelligence Environments by Means of Rule-based Programming with Visual Tools. In: ECAI 2006. Proc. 1st Workshop on Artificial Intelligence Techniques for Ambient Intelligence, Trento, Italy (2006)
11. National Imagery and Mapping Agency (NIMA), Performance Specification Digital Nautical Chart, MIL-PRF-89023 (December 19, 1997)
12. Noy, N., Musen, M.: Evaluating Ontology-Mapping Tools: Requirements and Experience. In: Proc. OntoWeb-SIG3 Workshop at the 13th International Conference on Knowledge Engineering and Knowledge Management, Siguenza, Spain (2002)
13. O'Connor, M., Knublauch, H., Tu, S., Grosz, B., Dean, M., Grosso, W., Musen, M.: Supporting Rule Systems Interoperability on the Semantic Web with SWRL. In: Proc. 4th International Semantic Web Conference, Galway, Ireland (November 2005)
14. Ressler, J., Dean, M., Kolas, D.: Geospatial Ontology Trade Study (June 2007), [http://projects.semwebcentral.org/docman/?group\\_id=84](http://projects.semwebcentral.org/docman/?group_id=84)



**Appendix: ENC to DNC Mapping Rule in SWRL XML Format**

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<swrlx:Ontology
xmlns:swrlx="http://www.w3.org/2003/11/swrlx"
xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
xmlns:j.0="http://protege.stanford.edu/plugins/owl/protege#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:owlx="http://www.w3.org/2003/05/owl-xml"
xmlns:p1="http://www.owl-ontologies.com/assert.owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:ruleml="http://www.w3.org/2003/11/ruleml"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ruleml:var>vertClr</ruleml:var>
  <ruleml:var>Bridge1</ruleml:var>
  <ruleml:var>Bridge1</ruleml:var>
  <ruleml:var>horizClr</ruleml:var>
  <ruleml:var>desc</ruleml:var>
  <ruleml:var>Bridge1</ruleml:var>
  <ruleml:var>Bridge1</ruleml:var>
  <ruleml:imp>
    <ruleml:_rlob ruleml:href="#BridgeMap"/>
    <ruleml:_body>
      <swrlx:classAtom>
        <owlx:Class
owlx:name="http://ww4.geoenterpriselab.com/ont/s57/S57V31C016.owl#Bridge"/>
          <ruleml:var>Bridge1</ruleml:var>
        </swrlx:classAtom>
        <swrlx:datavaluedPropertyAtom
swrlx:property="http://ww4.geoenterpriselab.com/ont/s57/S57V31C016.owl#horizontalClearance">
          <ruleml:var>Bridge1</ruleml:var>
          <ruleml:var>horizClr</ruleml:var>
        </swrlx:datavaluedPropertyAtom>
        <swrlx:datavaluedPropertyAtom
swrlx:property="http://ww4.geoenterpriselab.com/ont/s57/S57V31C016.owl#verticalClearance">
          <ruleml:var>Bridge1</ruleml:var>
          <ruleml:var>vertClr</ruleml:var>
        </swrlx:datavaluedPropertyAtom>
        <swrlx:datavaluedPropertyAtom
swrlx:property="http://ww4.geoenterpriselab.com/ont/s57/S57V31C016.owl#textDescription">
          <ruleml:var>Bridge1</ruleml:var>
          <ruleml:var>desc</ruleml:var>

```

```

        </swrlx:datavaluedPropertyAtom>
    </ruleml:_body>
    <ruleml:_head>
        <swrlx:classAtom>
            <owlx:Class
owlx:name="http://ww4.geoenterpriselab.com/ont/dnc/dncsubset.owl#Bridge"/>
                <ruleml:var>Bridge1</ruleml:var>
            </swrlx:classAtom>
            <swrlx:datavaluedPropertyAtom
swrlx:property="http://ww4.geoenterpriselab.com/ont/dnc/dncsubset.owl#safeHorizontalClearance">
                <ruleml:var>Bridge1</ruleml:var>
                <ruleml:var>horizClr</ruleml:var>
            </swrlx:datavaluedPropertyAtom>
            <swrlx:datavaluedPropertyAtom
swrlx:property="http://ww4.geoenterpriselab.com/ont/dnc/dncsubset.owl#overheadClearanceCategory">
                <ruleml:var>Bridge1</ruleml:var>
                <ruleml:var>vertClr</ruleml:var>
            </swrlx:datavaluedPropertyAtom>
            <swrlx:datavaluedPropertyAtom
swrlx:property="http://ww4.geoenterpriselab.com/ont/dnc/dncsubset.owl#objectName">
                <ruleml:var>Bridge1</ruleml:var>
                <ruleml:var>desc</ruleml:var>
            </swrlx:datavaluedPropertyAtom>
        </ruleml:_head>
    </ruleml:imp>
</swrlx:Ontology>

```

# Ontology-Based Information Extraction for Business Intelligence

Horacio Saggion, Adam Funk, Diana Maynard, and Kalina Bontcheva

Department of Computer Science, University of Sheffield  
Regent Court, 211 Portobello Street,  
Sheffield, S1 4DP  
United Kingdom  
{saggion,adam,diana,kalina}@dcs.shef.ac.uk

**Abstract.** Business Intelligence (BI) requires the acquisition and aggregation of key pieces of knowledge from multiple sources in order to provide valuable information to customers or feed statistical BI models and tools. The massive amount of information available to business analysts makes information extraction and other natural language processing tools key enablers for the acquisition and use of that semantic information. We describe the application of ontology-based extraction and merging in the context of a practical e-business application for the EU MUSING Project where the goal is to gather international company intelligence and country/region information. The results of our experiments so far are very promising and we are now in the process of building a complete end-to-end solution.

**Keywords:** Ontology-based Information Extraction, Business Intelligence, Cross-source Entity Coreference.

## 1 Introduction

Business intelligence (BI) can be defined as the process of finding, gathering, aggregating, and analysing information for decision making (See [9] for example). Semantic technologies of the type advocated by Semantic Web [6] are being applied for BI in the context of the EU MUSING<sup>1</sup> Project. MUSING is developing a new generation of BI tools and modules based on semantic-based knowledge and natural language processing (NLP) technology to mitigate the efforts involved in gathering, merging, and analysing information.

Information Extraction (IE) is a key NLP technology for automatically extracting specific types of information from text or other sources to create records in a database or populate knowledge bases, for example. Without an IE system, business analysts carrying out BI activities would have to read hundreds of textual reports, web sites, and tabular data to manually dig out the necessary information to feed BI models and tools.

---

<sup>1</sup> Multi-industry, Semantic-based next generation business INtelliGence.

<i>The road infrastructure in Argentina is <b>excellent</b>, even in remote areas. This is in sharp contrast to Brazil and, to a lesser extent, Chile. The transportation and communication infrastructures seem more than adequate to allow quick price discovery and easy communication between the processors and farmers for sample test results and other marketing matters.</i>
<i>The forest area in India extended to about <b>75 million hectares</b>, which in terms of geographical area is approximately 22 per cent of the total land. Out of this, 9.5 million hectares is fallow and 7 million hectares is under shrub formation. Thus, an actual forest area is less than 17 per cent. <i>The total area under forest in Tamil Nadu is <b>21,072 sq.km.</b> of which 17,264 sq.km. is reserved forest and 3,808 sq.kms is reserved land. This constitutes 16 per cent of the total geographical area of the State.</i></i>
<i>Political stability in India is <b>threatened</b> by the Kashmir dispute and other internal issues.</i>
<i>The population in India as of March 2001 stood at <b>1,027,015,247 persons</b>. With this, India became only the second country in the world after China to cross the one billionmark. India's population rose by 21.34% between 1991 - 2001. The sex ratio (i.e., number of females per thousand males) of population was 933, rising from 927 as at the 1991 Census. Total literacy rate in India was returned as 65.38%.</i>

**Fig. 1.** Multiple Textual Sources of Information for Internationalisation Applications

Here, we concentrate on the application of *Ontology-based Information Extraction* (OBIE) in the context of Business Intelligence. OBIE is the process of identifying in text or other sources relevant concepts, properties, and relations expressed in an ontology. We are working with *domain ontologies* which represent the domain of application and which capture the experts' knowledge. Ontologies contain concepts arranged in class/sub-class hierarchies (e.g. a joint venture is a type of company agreement), relations between concepts (e.g., a joint venture has a starting date), and properties (e.g., a joint venture has only one starting date). An ontology we are working with is being developed for a e-business application in the internationalisation domain<sup>2</sup> where the objective is to model information about companies, countries, and regions. The ontologies are developed with the help of domain experts. These experts have identified that in a domain such as that of joint ventures, relevant concepts are: companies, nationalities, type of contractual form, date of constitution of the alliance, etc.

We have developed robust and adaptable technology for the extraction of relevant semantic information (expressed in the ontology) to be used in business intelligence processes in the following areas: financial risk management, internationalisation, and IT operational risk management. Specific applications in these areas are: credit risk assessment, international company intelligence, country or region selection, risk identification and mapping.

All these applications require the *extraction and merging* of information from a number of trusted but diverse data sources (e.g. database, financial reports, news reports, company web sites) which represents a challenge for any information extraction system.

We focus here on IE for the development of internationalisation applications. We rely on robust and adaptable tools from the GATE architecture [10]. Extraction of information for internationalisation applications consists on the identification of all mentions of concepts, instances, and properties in text or other sources (e.g. multimedia material such as tables and images). Some examples

<sup>2</sup> Internationalisation is the process that allows an enterprise to evolve its business from a local to an international dimension. In the MUSING context this involves for example the acquisition of information about international partnerships, contracts, investments, etc.

are presented in Figures 11. Information in these sources is required for internationalisation applications dealing with, for example, companies desiring to take their business abroad and interested in knowing the best places to invest. These applications usually require the gathering of information on *economic indicators* such as the population of a particular country or region (e.g. the market potential), the status of the transport infrastructure, the literacy rate, the political situation, criminality indices or whether a region is prone to particular natural disasters. Both quantitative (e.g. numeric) and qualitative (e.g. categorical) information is necessary. Our IE components also extract information from images which is carried out by components which operate on the output of an OCR system.

Once the information has been gathered from different sources, the ontology has to be populated with all mentions found in text, in order to do so, the system has to decide if two mentions in different sources refer to the same entity in the real world. The populated ontology (or knowledge base(KB)) is queried by the different applications in the MUSING Project, the semantic information from the KB is used as valuable information for customers or applied to statistical models of decision making. Because some applications require perfect output from the IE system, a user verifies the extraction results.

In this paper we describe the development of an Ontology-based information extraction for business intelligence in the context of internationalisation applications. The paper is structured as follows: In the next Section, we describe the MUSING project with respect to the information extraction task. Section 3 describes our approach to cross-source entity identification for Ontology population. Adaptation of our NLP technology for internationalisation applications is described in Section 4. Section 5 reports on related work on business intelligence and ontology-based extraction. Finally, in Section 6, we present our conclusions.

## 2 MUSING Information Extraction Technology

In Figure 2 we present the MUSING IE architecture. A number of data sources for information extraction have been identified and documents and multimedia material collected and stored in the MUSING document repository. In addition to data provided by different partners in the project, a number of on-line data sources for business intelligence (e.g., Yahoo! Finance, World Bank, CIA Fact Book) are being targeted.

Documents are then processed by an Ontology-based annotation tool which automatically detects information specified in a domain ontology. The ontology has been developed through interaction with MUSING domain experts and implemented in OWL [11] extending the PROTON Upper Ontology<sup>3</sup>.

A collaborative annotation tool developed by the University of Sheffield as part of the EU Neon Project<sup>4</sup> has been adapted to the MUSING ontologies in

<sup>3</sup> <http://proton.semanticweb.org/>

<sup>4</sup> <http://www.neon-project.org>

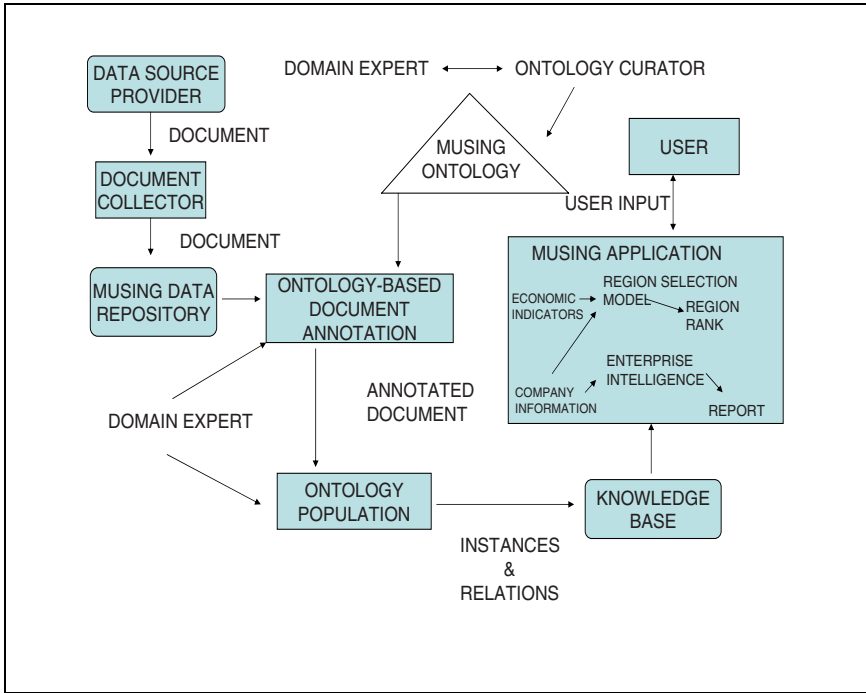


Fig. 2. Ontology-based Extraction Architecture in MUSING

order to allow users not only to annotate documents from scratch but also to correct the results of the automatic semantic annotation process.

The current version of the tool supports both annotation of ontology class as well as relations. The tool is being used by experts to identify qualitative information in text which may include complicated statements about inflation, growth, reliability, etc. expressed in phrases, sentences or even full paragraphs.

Once documents have been automatically annotated, their annotations are analysed by an ontology population mechanism in charge of creating instances and relations for knowledge base population. Tuples in the knowledge base are used in different Musing applications. One such application is providing up-to-date information about companies (e.g. for identifying possible business partners) and another application is providing ranked lists of countries/regions for companies interested in investing into new country/regions.

### 2.1 GATE Processing Tools

We have developed our information extraction system using the General Architecture for Text Engineering (GATE). GATE is a framework for the development and deployment of language processing technology in large scale [10]. It provides three types of resources: Language Resources (LRs) which collectively refer to data; Processing Resources (PRs) which are used to refer to algorithms; and

Visualisation Resources (VRs) which represent visualisation and editing components. GATE can be used to process documents in different formats including plain text, HTML, XML, RTF, and SGML. When a document is loaded or opened in GATE, a document structure analyser is called upon which is in charge of creating a GATE document, a LR which will contain the text of the original document and one or more sets of annotations, one of which will contain the document mark-ups (for example HTML). Annotations are generally updated by PRs during text analysis - but they can also be created during annotation editing in the GATE GUI. Each annotation belongs to an annotation set and has a type, a pair of offsets (the span of text one wants to annotate), and a set of features and values that are used to encode the information.

A key element in the annotations is the encoding of ontological information - our applications create *Mention* annotations which make reference to the target ontology as well as the ontological concept a string of text refers to.

While GATE comes with a default information extraction system called ANNIE [18], it is only partially relevant to the business domain. The ANNIE system identifies generic concepts such as person names, locations, organisation, dates, etc. Therefore we had to develop new rules or adapt rules for our applications.

For the work reported here, we have carried out adaptation of the GATE named entity recognition components because most target entities are not covered by ANNIE. We have also developed a conceptual mapping module to map concepts identified by our system into the ontologies of the application domains. Future versions of our system will apply machine learning techniques incorporated into the GATE framework.

### 3 Cluster-Based Cross-Document Entity Coreference

In a scenario such as the MUSING one where information is extracted from many sources, one has to deal with the problem of identifying whether two business entities in two different sources refer to the same individual in the real world. A problem known as ontology population [1] in the Semantic Web community. Solving this problem is extremely important in order to create an accurate picture of individuals as well as organisations. In fact, in business, the reputation of a particular company may depend on the reputation of its board of directors, and therefore bad news about a company director may influence a company's performance.

An example of this is presented in Figure 3, where a number of sources contain references to the same person name "Dale Merritt". Knowing if this particular person has criminal charges is important and may well influence a decision such as participating in a commercial agreement involving such person.

We have applied text mining techniques to the cross-source coreference problem focusing on the problem of person name coreference. The context we are carrying out this piece of research is one where information about a particular entity is required and a set of documents are retrieved from data sources based on the entity description (e.g. person name). The task at hand consist on

<b>Dale Merritt</b> , CEO of DotVentures, LLC has just signed on for a five week pilot season, with guests who specialize in domain investing and world wide web real estate, on the live talk radio show, "Domain Investing" ...
DotVentures, an Internet domain name investing company, today announced a partnership between their company and Internet marketing software company, SearchMarketing. President of DotVentures, <b>Mr. Merritt</b> says, "DotVentures and SearchMarketing have integrated ...
<b>Dale Merritt</b> , who was charged by the Federal Trade Commission along with his firm, Showcase Distributing, Inc., of Phoenix, Arizona, as part of a nationwide crackdown on business opportunity fraud, ..
Haystack Ranch Events... June 30 - July 1, 2007... Registration Contact: E-mail Brenda or <b>Dale Merritt</b> or phone 303.681.2098 (Brenda or <b>Dale Merritt</b> )

**Fig. 3.** Cross-source Coreference Problem Instance

identifying what sets of documents refer to the same entity in the real world. Once this has been done, the entity can be assigned a unique identifier in the knowledge base.

As past and recent research [4,20], we have addressed the problem as a document clustering problem. We have implemented an agglomerative clustering algorithm. The input to the algorithm is a set of vectors representations (e.g. terms and weights) which are extracted from the annotated documents. We have experimented with two types of data representation which are derived from the annotation of documents using our NLP tools. One representation is based on a bag-of-words approach while the other uses specific types of semantic information extracted from the annotated documents [21]. Terms are either words from the documents or named entities in the targeted ontology (PROTON Upper).

When clustering starts, there are as many clusters as input documents; as the algorithm proceeds clusters are merged until a certain termination condition is reached. The algorithm computes the similarity between vector representations in order to decide whether or not to merge two clusters. The similarity metric we use is the cosine of the angle between two vectors. This metric gives value one for identical vectors and zero for vectors which are orthogonal (non related). Various options have been implemented in order to measure how close two clusters are, one metric we have used is the following: the similarity between two clusters ( $\text{sim}_C$ ) is equivalent to the "document" similarity ( $\text{sim}_D$ ) between the two more similar documents in the two clusters; the following formula is used:

$$\text{sim}_C(C_1, C_2) = \max_{d_i \in C_1; d_j \in C_2} \text{sim}_D(d_i, d_j)$$

Where  $C_k$  are clusters,  $d_l$  are document representations (e.g., vectors), and  $\text{sim}_D$  is the cosine metric.

If this similarity is greater than a threshold – experimentally obtained – the two clusters are merged together. At each iteration the most similar pair of clusters is merged. If this similarity is less than a certain threshold the algorithm stops. In order to identify the optimal threshold we have experimented with training data. The threshold was selected in order to obtain optimal performance in the training data.



In order to test the success of the implemented techniques, we have carried out a number of experiments with test data from the SemEval 2007 evaluation on People Web Search task [2]. In this evaluation, systems receive a name and a set of documents containing the name, the system has to decide how many different individuals are there, and what documents correspond to what individuals (e.g. clustering). Evaluation of the task is carried out using standard clustering evaluation measures of “purity” and “inverse purity” [13], and the harmonic mean of purity and inverse purity: F-score. Our algorithm is competitive when compared with the best system in that evaluation. One particular configuration of our system which uses specific types of semantic information obtained a (micro-averaged) F-score of 78% (same performance as the best SemEval 2007 system [7]) and a macro-average F-score one point more than the best system.

## 4 Information Extraction for Internationalisation Applications

In order to support intelligence gathering for BI we use a number of sources of information for developing internationalisation application: one is a set of company profiles that we have mined from Yahoo! Finance, another source is a set of around 100 company web sites, yet another source is company reports and newspaper articles provided by our partners in the project.

In order to collect available information for companies we have automatically gathered the main page of the company web site, and crawled pages containing contact information and company activities. We followed page links which contain certain keywords such as “contact us”, “about us”, etc. in the **href** attribute of an html anchor link or the text surrounding the anchor. For Yahoo! Finance documents we have developed a script which crawls the information for each company based on their company symbol.

### 4.1 Company Intelligence

One prototype we are developing is an International Enterprise Intelligence application the objective of which is to provide customers with up-to-date and correct information about companies. The information is mined from many different sources such as web pages, financial news, and structured data sources which after annotation and merging is stored in knowledge base tuples. Among other concepts to be targeted by the application are the company name, its main activities, its number of employees, its board of directors, etc.

The application consists of standard GATE components together with new linguistic and named entity recognition processors which map concepts to ontological classes. Figure 4 shows the automatic annotation of concepts in text. The OCAT tool is used to display the link between ontology and annotated text. The result of the annotation is further analysed by the ontology population module responsible for knowledge base population. KB tuples returned in answer to a user request are used to display information in a Web-based user interface (e.g.

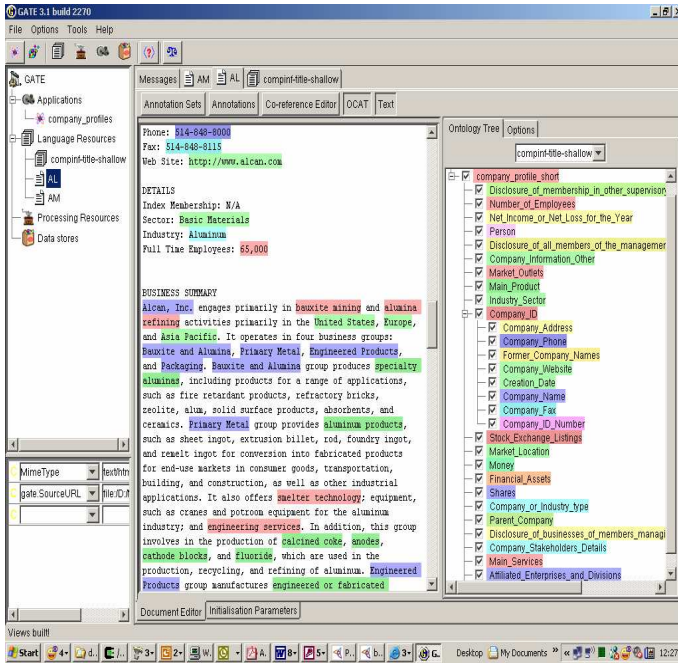


Fig. 4. OBIE for International Company Intelligence

Yellow Page style). We have carried out evaluation of this application using traditional IE metrics [8,22]: precision, recall, and f-score. An expert manually annotated 5 documents and we compared the results of the system annotations against this gold standard set. The overall performance of the system was an f-score of 84% (for details of this evaluation see [17]) which is acceptable for system deployment.

## 4.2 Country/Region Intelligence

The second application we present here aims at extracting relevant information about countries and region in the globe. Sources of information used in this application are country profiles and statistics from various sources (e.g. World Bank, Monetary Fund). For system development and testing we have collected a corpus of documents, we have used crawling scripts that target specific web sites (e.g., BBC, Wikipedia, CIA World Fact Book) which contain the information required by the application. Concepts we target are: country name; official language; currency; exchange rate; foreign debt; unemployment rate; GDP; and foreign investments. In addition to these common types of information, more specific are economic indicators and indices such as the mortality index, region area, population, education, etc. In Figure 5 we show a screen-shot of a document annotated with semantic information for this particular application. The

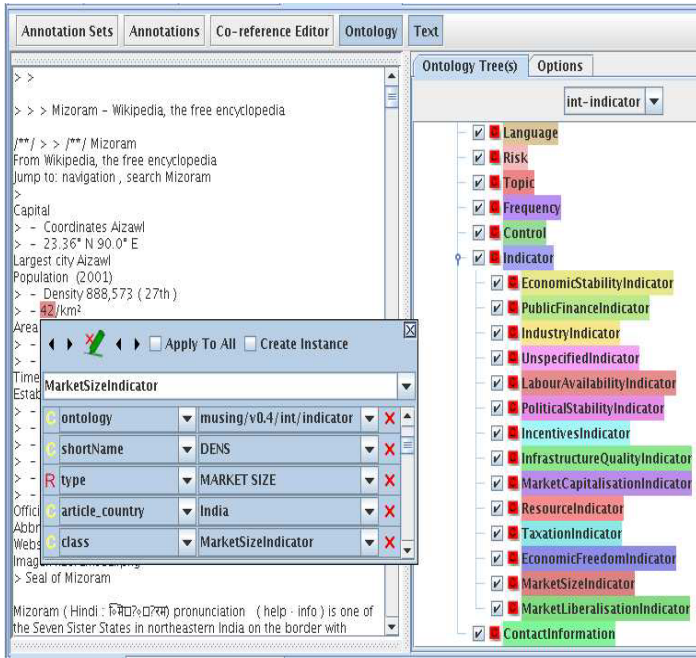


Fig. 5. Output of the Automatic Annotation Process

Figure also shows the ontology being used which contains among others some relevant economic indicators.

For each identified concept, features and values are stored in the annotation. The resulting information is used to feed a statistical model of country/region selection [15] which using both information about a country and a company (Section 4.1) decides which regions in the globe are more suitable to undertake business in.

The application’s output can be seen in Figure 6. It shows a ranking of Indian regions most promising for investment. In addition to the ranking, the application indicates which model variables have mainly contributed to the obtained ranking.

The application was developed using standard as well as adapted GATE processing resources. A number of domain specific gazetteer lists were developed. One set of gazetteer lists is in charge of helping identify text types targeted by the application, another set identifies names of places associated with countries targeted by the application and helps associate capital cities with regions or countries for example. Named entity recognizers target the specific concepts or indicators required by the application and map them to the ontology. The rules for some particular types of text are highly accurate.

In Table 1 we present the performance of the extraction system in terms of precision, recall, and f-score. This is an evaluation of the extraction of 6 key economic indicators from 34 semi-structured web pages about Indian regions found in the Wikipedia web site. The economic indicators targeted by the

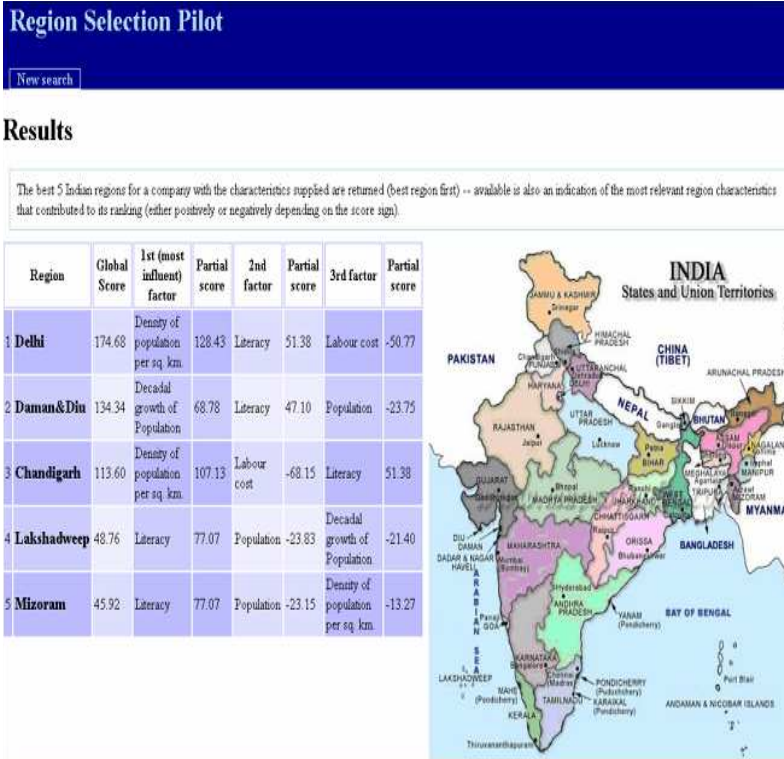


Fig. 6. Output of the Region Selection Application

Table 1. Quantitative Evaluation of Region/Country Intelligence Application

Concept	Precision	Recall	F-score
DENS	92%	68%	80%
SURF	100%	94%	97%
EMP	50%	100%	75%
LRT	88%	41%	64%
LRM	100%	29%	64%
LRF	100%	38%	69%
Total	94%	67%	81%

application are density of population (DENS), region surface (SURF), employment rate (EMP), literacy rate (LRT), literacy rate male (LRM), literacy rate female (LRF). The overall performance of the application is an F-score of 81%. Note that because country/region information changes periodically, this application has to be run whenever new documents are available, thus ensuring that the value of the indicators are up-to-date.

## 5 Related Work

Information extraction in the business domain is not new, [23] developed a machine learning approach to identify patterns for the identification of *corporate management changes* in text, which is relevant in the context of BI. Such system should be able to identify positions in an organisation which are changing hands as well as who are the actors involved in the changes. While succession management is not an specific focus of MUSING we are dealing with a variety of applications in BI. In addition we are dealing with the whole development cycle from the creation of patterns to the extraction and mapping of the information to the ontology.

h-TechSight [19] is a system which also uses GATE to detect changes and trends in business information and to monitor markets. It uses semantically-enhanced information extraction and information retrieval tools to identify important concepts with respect to an ontology, and to track changes over time. This system differs from MUSING in that the information acquired is only related to a quite shallow and simple ontology with a few fairly fixed concepts.

Information extraction is also used in the MBOI tool [12] for discovering business opportunities on the internet. The main aim is to help users to decide about which company tenders require further investigation. This enables the user to perform precise querying over named entities recognised by the system. Similarly the LIXTO tool is used for web data extraction for business intelligence [5], for example to acquire sales price information from online sales sites. However, this requires a semi-structured data source which is not always available or sufficient for the kind of financial information we are concerned with.

EBiZPort [16] is a portal for information gathering in BI. The tool incorporates a meta-search process to leverage different information sources also addressing the merging problem (but not at the entity level as in our case). The tool incorporates summarization, classification, and visualisation techniques. In this approach, it is still up to the user to find the relevant information in the mass of documents returned by the system. We go beyond this by providing extraction of relevant concepts to feed BI models.

Similar to our approach to instance merging is [3] where the problem of instance unification for author names is addresses. They mine information from the Web for authors including full name, personal page, and co-citation information to compute the similarity between two person names. Similarity is based on a formula which combines numeric features with appropriate weights experimentally obtained.

## 6 Conclusions and Further Work

Business Intelligence requires business analysts to gather, merge, and analyse considerable amounts of information in multiple formats and from heterogeneous sources. Information extraction technology is a key enabler to identify in text key pieces of information to be used in BI tools. Clustering techniques are powerful tools to merge information across different sources.

We have described how available and robust information extraction technology is being adapted to create an Ontology-based information extraction system in the context of the MUSING project. The system produces ontological annotations which are transformed into tuples for ontology population. The system already extracts and merges information from various sources and for specific applications in financial risk management and internationalisation. Applications are being created which use the valuable information in the knowledge based to perform reasoning or provide valuable information to customers. Performance measured through quantitative evaluation in both extraction and cross-source coreference look promising.

While MUSING is an ongoing project, we have already developed robust technology for deploying BI applications. The evaluation presented here is mainly quantitative, in the future the applications will be evaluated in terms of usability and user satisfaction.

Our current work is exploring the extraction of information from business graphics and tabular data which is based on the use of flexible gazetteer lookup procedures available in GATE which are being applied to OCR analysis of images. The output of the OCR analysis is being corrected by the exploitation of collateral information found around the graphics. This methodology has already prove useful, with improvements of around 3% over extraction from OCR alone.

Our future work will further improve our extraction tools incorporating machine learning capabilities into the extraction system [14], this will ensure that scalability is properly addressed in the extraction process. Our work on merging or ontology population will be extended to cover other semantic categories including locations, organisations, and specific business events (e.g., joint ventures).

## Acknowledgements

This work is partially supported by the EU-funded MUSING project (IST-2004-027097).

## References

1. Alani, H., Dasmahapatra, S., Gibbins, N., Glaser, H., Harris, S., Kalfoglou, Y., O'Hara, K., Shadbolt, N.: Managing Reference: Ensuring Referential Integrity of Ontologies for the Semantic Web. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS (LNAI), vol. 2473, pp. 317–334. Springer, Heidelberg (2002)
2. Artiles, J., Gonzalo, J., Sekine, S.: The SemEval-2007 WePS Evaluation: Establishing a benchmark for Web People Search Task. In: Proceedings of Semeval 2007, Association for Computational Linguistics (2007)
3. Aswani, N., Bontcheva, K., Cunningham, H.: Mining information for instance unification. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)



4. Bagga, A., Baldwin, B.: Entity-Based Cross-Document Coreferencing Using the Vector Space Model. In: COLING-ACL 1998. Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics, pp. 79–85 (1998)
5. Baumgartner, R., Frlich, O., Gottlob, G., Harz, P., Herzog, M., Lehmann, P.: Web data extraction for business intelligence: the lixt0 approach. In: Proc. of BTW 2005 (2005)
6. Bontcheva, K., Cunningham, H.: The semantic web: A new opportunity and challenge for human language technology. In: Cunningham, H., Ding, Y., Kiryakov, A. (eds.) Proceedings of Workshop on Human Language Technology for the Semantic Web and Web Services, 2<sup>nd</sup> International Semantic Web Conference, Sanibel Island, Florida, October 2003 (2003), <http://www.gate.ac.uk/sale/iswc03/iswc03.pdf>
7. Chen, Y., Martin, J.H.: Cu-comsem: Exploring rich features for unsupervised web personal named disambiguation. In: Proceedings of SemEval 2007, Association for Computational Linguistics, pp. 125–128 (2007)
8. Chinchor, N.: Muc-4 evaluation metrics. In: Proceedings of the Fourth Message Understanding Conference, pp. 22–29 (1992)
9. Chung, W., Chen, H., Nunamaker Jr., J.F.: Business Intelligence Explorer: A Knowledge Map Framework for Discovering Business Intelligence on the Web. In: Hawaii International Conference on System Sciences, IEEE Computer Society Press, Los Alamitos (2003)
10. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: ACL 2002. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (2002)
11. Dean, M., Schreiber, G., Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: OWL web ontology language reference. In: W3C recommendation, W3C (February 2004), <http://www.w3.org/TR/owl-ref/>
12. Nie, J.-Y., Paradis, F., Tajarobi, A.: Discovery of business opportunities on the internet with information extraction. In: IJCAI 2005. Workshop on Multi-Agent Information Retrieval and Recommender Systems, Edinburgh, Scotland, pp. 47–54 (2005)
13. Hotho, A., Staab, S., Stumme, G.: WordNet improves text document clustering. In: Proc. of the SIGIR 2003 Semantic Web Workshop (2003)
14. Li, Y., Bontcheva, K., Cunningham, H.: An SVM Based Learning Algorithm for Information Extraction. Machine Learning Workshop, Sheffield (2004), <http://gate.ac.uk/sale/ml-ws04/mlw2004.pdf>
15. Majocchi, A., Strange, R.: The FDI Location Decision: does Liberalisation Matter? Transactional Corporation Review (to appear, 2007)
16. Marshall, A., McDonald, D., Chen, H., Chung, W.: EBizPort: Collecting and Analysing Business Intelligence Information. Journal of the American Society for Information Science and Technology 55(10), 873–891 (2004)
17. Maynard, D., Saggion, H., Yankova, M., Bontcheva, K., Peters, W.: natural language technology for information integration in business intelligence. In: Abramowicz, W. (ed.) 10th International Conference on Business Information Systems, Poland, pp. 25–27 (April 2007), <http://gate.ac.uk/sale/bis07/musing-bis07-final.pdf>
18. Maynard, D., Tablan, V., Ursu, C., Cunningham, H., Wilks, Y.: Named Entity Recognition from Diverse Text Types. In: Recent Advances in Natural Language Processing 2001 Conference, Tzigrav Chark, Bulgaria, pp. 257–274 (2001)

19. Maynard, D., Yankova, M., Kourakis, A., Kokossis, A.: Ontology-based information extraction for market monitoring and technology watch. In: ESWC Workshop End User Apects of the Semantic Web, Heraklion, Crete (2005)
20. Phan, X.-H., Nguyen, L.-M., Horiguchi, S.: Personal name resolution crossover documents by a semantics-based approach. In: IEICE Trans. Inf. & Syst. (February 2006)
21. Saggion, H.: Shef: Semantic tagging and summarization techniques applied to cross-document coreference. In: Proceedings of SemEval 2007, Association for Computational Linguistics, pp. 292–295 (2007)
22. van Rijsbergen, C.J.: Information Retrieval, Butterworths, London (1979)
23. Yangarber, R., Grishman, R., Tapanainen, P., Huttunen, S.: Unsupervised Discovery of Scenario-level Patterns for Information Extraction. In: Proceedings of ANLP-NAACL 2000, Seattle, WA (2000)



# EIAW: Towards a Business-Friendly Data Warehouse Using Semantic Web Technologies

Guotong Xie<sup>1</sup>, Yang Yang<sup>1</sup>, Shengping Liu<sup>1</sup>, Zhaoming Qiu<sup>1</sup>, Yue Pan<sup>1</sup>,  
and Xiongzhi Zhou<sup>2</sup>

<sup>1</sup> IBM China Research Laboratory  
Zhongguancun Software Park, Beijing, 100094, China  
{xieguot, yangyy, liusp, qiuzhaom, panyue}@cn.ibm.com

<sup>2</sup> Taikang Life Insurance Company  
No.156, FuXingMen Street, Beijing, 100032, China  
zhouxiongzhi@taikanglife.com

**Abstract.** Data warehouse is now widely used in business analysis and decision making processes. To adapt the rapidly changing business environment, we develop a tool to make data warehouses more business-friendly by using Semantic Web technologies. The main idea is to make business semantics explicit by uniformly representing the business metadata (i.e. conceptual enterprise data model and multidimensional model) with an extended OWL language. Then a mapping from the business metadata to the schema of the data warehouse is built. When an analysis request is raised, a customized data mart with data populated from the data warehouse can be automatically generated with the help of this built-in knowledge. This tool, called Enterprise Information Asset Workbench (EIAW), is deployed at the Taikang Life Insurance Company, one of the top five insurance companies of China. User feedback shows that OWL provides an excellent basis for the representation of business semantics in data warehouse, but many necessary extensions are also needed in the real application. The user also deemed this tool very helpful because of its flexibility and speeding up data mart deployment in face of business changes.

## 1 Introduction

Data warehousing and business intelligence (BI) are key technologies for decision making in the industry. A typical BI application deployment process usually requires an existing Enterprise Data Warehouse (EDW), which integrates enterprise-wide data from multiple autonomous heterogeneous data sources and provides a consistent single view of data. A multidimensional model and its corresponding data mart schema can be designed based on business user's analysis requirement. Then, relevant data from EDW are transformed and loaded into the data mart and/or a cube for doing the analysis. The analysis results and reports are finally delivered to the business user.

However, in practice, it proves very difficult to successfully implement the above process [14]. Given the fact that building an enterprise-wide data warehouse is a very time-consuming and expensive activity, to make satisfactory return from this

investment is a key issue for the success of BI+EDW system. The typical BI application deployment process is not flexible enough to deal with a fast changing dynamic business environment. For example, about every 2 working days, a new analysis requirement is submitted to the BI department from business units of Taikang Life Insurance Company. For all these analysis requirements IT workers must communicate with business people, understand the business content and redesign the data mart schema and the ETL processes from the EDW to data mart.

This difficulty is mainly due to the fact that the business semantics [6] is only kept in BI designer's mind, then it is hard-coded for physical implementations. The business semantics can be, at least partially, represented by the business metadata that provides a business-oriented description of the data warehouse content and a formal representation of the analysis requirement [8,9]. The two basic types of business metadata are conceptual enterprise data model and multidimensional model. Conceptual enterprise data model is a model to organize business terminology in a semantic way. It is a view of *how the business works* and consists of business concepts, attributes of concepts and relationships among concepts. Multidimensional model is a model to define the analytic requirements for BI application. It is a view of *how the business is measured* and consists of measures and dimensions.

In the typical deployment process, the business metadata becomes technical artifacts which cannot be understood by business users. The conceptual enterprise data model is hidden behind the schema of EDW. The intended meaning of the measures and dimensions are implemented by the ETL process from EDW to the data mart. There is no distinction between transformations needed for the business semantics and transformations which are mainly due to the technical issues.

Fortunately, with the emergence of Semantic Web [2], the formal ontology representation language OWL (Web Ontology Language) [10] has been standardized by W3C. OWL is appropriate for representing business semantics in a formal way [9]. Therefore, we develop a tool to make data warehouses more business-friendly by adopting the Semantic Web technologies. This tool is called Enterprise Information Asset Workbench (EIAW). The main idea is to make business semantics explicit in the data warehouse system by formally representing the business metadata with an extended OWL language. In our tool, the conceptual enterprise data model is expressed by W3C's Web Ontology Language OWL, in particular, OWL-DL, and the multidimensional model is expressed by OWL-DL extended with concrete domain, predefined functions, property path expression, etc.

Based on the explicit business semantics, EIAW supports the deployment of a data warehouse-based BI application with the following steps, assuming the pre-existence of a conceptual enterprise data model and an enterprise data warehouse:

- 1) Business users build the analysis requirements (multidimensional model) using business terms from the conceptual enterprise data model;
- 2) IT users only need to build mapping from the business terms involved in the multidimensional model to the data warehouse schema;
- 3) The system automatically generates a customized data mart with aggregated data and an OLAP cube metadata supported by industry standard.

The main advantage of the above deployment process is the separation of concerns of business user and IT user. Business users can organize their business knowledge

and express their analysis requirements using business terms familiar to them. IT users focus on resolving the mapping from business terms to EDW schema from the technical view and do not care the business contents of measures and dimensions. In conventional approach, IT users need to fully understand the meanings of measures and dimensions and then design schema and ETL scripts for them. They also need to redo the whole process if the measures and dimensions are changed. In our proposed approach, business users can have a more efficient way to describe what they want to get instead of arranging lots of meetings to communicate with IT people. In addition, in case that analysis requirements are changed, business users can modify the definitions of measures and dimensions, then the data mart can be automatically deployed without IT worker's engagement if the mappings are pre-built.

EIAW is implemented as a plug-in on the Eclipse 3.2 and deployed in Taikang Life Insurance Company as a result of a collaborative project between IBM and Taikang<sup>1</sup>. The positive feedbacks from Taikang are that the explicit business semantics by OWL greatly increases the flexibility, speeds up data mart deployment and improves the quality of multidimensional model. On the other hand the customer also thinks that OWL is not an easy-to-learn language, especially for the property restrictions.

This paper is structured as follows. Section 2 describes the system architecture of EIAW. Section 3 describes the unified representation of business metadata in the EDW system with an extended OWL language. Section 4 shows the mapping from the business terms to the EDW schema. Section 5 presents the approach to automatically deploying the data mart according to business users' requirements. Section 6 discusses the users' evaluation and feedback. Section 7 discusses the related works. Finally, Section 8 draws the conclusions and discusses the future works.

## 2 System Architecture

The simplified view of the system architecture of EIAW tool is depicted at Fig. 1. The system consists of three main modules: business metadata builder, mapping builder and deployment engine.

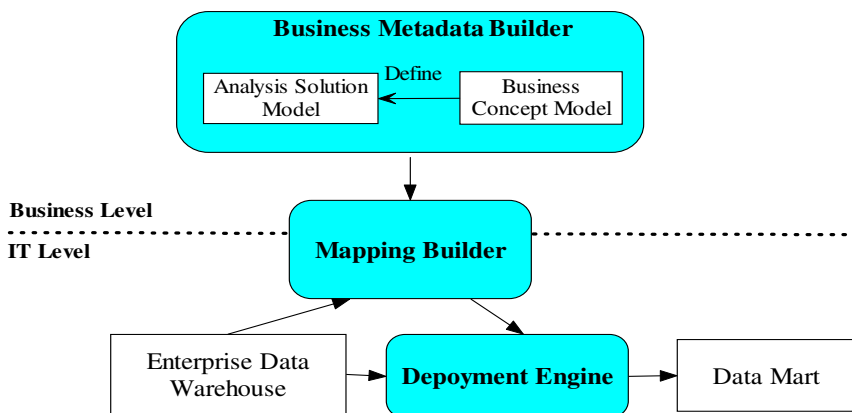


Fig. 1. The System Architecture

<sup>1</sup> The press release: <http://www-03.ibm.com/press/us/en/pressrelease/19434.wss>

**Business Metadata Builder.** The business metadata builder supports business user to create and edit the business metadata in the data warehousing environment. In this tool, the conceptual enterprise data model is called Business Concept Model (BCM) and the conceptual multidimensional model is called Analysis Solution Model (ASM). The ASM is defined using the business terms from the BCM.

**Mapping Builder.** The mapping builder supports IT users to build the mapping from business terms to the EDW schema. IT users only need to build the mapping for the business terms appeared in the definitions of ASMs. The mapping builder also supports the reuse and incremental building of the mappings. That is to say, the mapping for one business term can be shared if it appears in the definitions of other ASMs. With the accumulation of the mappings for business terms, there will be more and more new ASMs for which all the mappings for their business terms have already been built by others. That means these ASMs can be automatically deployed without IT people's involvement.

**Deployment Engine.** The deployment engine automatically generates a data mart with aggregated data populated from the EDW, provided the definition of ASM and mappings are given. The deployment engine can also generate cube metadata for this data mart to enable OLAP analysis.

### 3 Business Metadata

Business metadata plays an important role in the business-friendly data warehouse system. There are two types of business metadata supported in EIAW: BCM and ASM.

#### 3.1 Business Concept Model

Enterprise Data Model plays a critical role in the planning and designing phase and is critical for the future success of the enterprise data warehouse. An Enterprise Data Model is an integrated view of the data produced and consumed across the entire organization. It unifies, formalizes and represents the things important to an organization, as well as the rules governing them. The conceptual enterprise data model is always represented by an Entity-Relationship (ER) model in industry.

Because W3C's Web Ontology Language (OWL), in particular, OWL-DL, is more expressive and has more formal semantics compared to the ER language, EIAW adopts OWL-DL to represent the conceptual enterprise data model (called BCM in EIAW). So, the BCM editor is basically an OWL editor as shown by **Fig. 2**. In the BCM editor, the names used for the terminology are concept (owl:Class), attribute (owl:DatatypeProperty) and relationship (owl:ObjectProperty), which are more familiar to the data warehouse people.

In practices, BCM can be constructed from the scratch or transformed from existing ER models in industry.

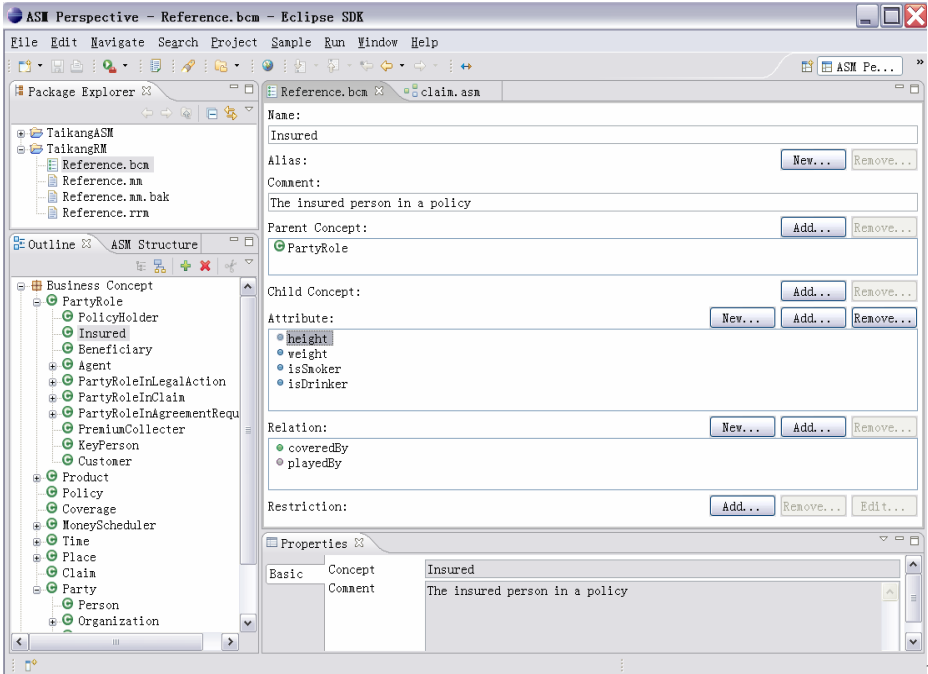


Fig. 2. The BCM Editor

### 3.2 Analysis Solution Model

Multidimensional model is a kind of data model widely used for description of the multidimensional and aggregative nature of OLAP applications [3]. The basic notions are the dimension, measure and cube. A dimension represents a business perspective under which data analysis is to be performed and is organized in a hierarchy of dimension levels, which correspond to granularity of the dimension. A dimension can be organized into different hierarchies that correspond to different views of the dimension. A measure represents factual data to be analyzed. A cube associates a set of measures with some defined dimensions. For example, in a claim analysis for insurance company, some of the measures which are of interests are number of fraudulent claims, total amount paid to the claims. These measures can be viewed along several dimensions: actuarial category of the insurance products, professional risk level of the insured, age of policy holders, etc.

In EIAW, the multidimensional model is called Analysis Solution Model (ASM). The most prominent feature of ASM is that the intended meaning for the dimensions and measures are further represented in a semantic way. Different from traditional multidimensional models, all measures and dimensions in ASM are defined by using business terms in BCM. This approach makes automatic

deployment of data mart possible. Our business terms include the vocabularies (the named classes and properties) from the BCM and the role expressions on the BCM. The role expression as an extension on the OWL-DL, called *property path* in EIAW, is defined as:

$$R, S \rightarrow P \mid P^{-} \mid R.S \mid C.R \mid R_{[D]}$$

where P denotes the atomic role in OWL-DL, C and D denote the named class in OWL-DL, and R and S denote the property path in EIAW. The property path enables to access an indirect property from the starting class. For example, “Agent’s name” is represented in textual form as *Agent.playedBy[Person].name*, which means that: the name of an agent is got by the name of the person that plays the role of *Agent*.

From the semantic point of view, given an interpretation  $\mathcal{I}$ , class is interpreted as a subset of the domain  $\Delta^{\mathcal{I}}$ , and roles as binary relations over  $\Delta^{\mathcal{I}}$ , the semantics for the newly introduced constructs are:  $(C.R)^{\mathcal{I}} \rightarrow R^{\mathcal{I}} \cap (C^{\mathcal{I}} \times \Delta^{\mathcal{I}})$ ,  $(R_{[D]})^{\mathcal{I}} \rightarrow R^{\mathcal{I}} \cap (\Delta^{\mathcal{I}} \times D^{\mathcal{I}})$ ,  $(R.S)^{\mathcal{I}} \rightarrow R^{\mathcal{I}} \circ S^{\mathcal{I}}$ . It should be note that the property path is not allowed to participate in the class definitions of OWL because it is only used for the definition of ASM.

### 3.2.1 Representation of Measures

There are two kinds of measure in ASM. One kind is atomic measure, which defines basic variable to be evaluated in business analysis. E.g. number of claims. Another kind is complex measure, which is defined by a function on other atomic or complex measures.

The atomic measure is represented by an aggregate function on an extended expression defined on the ontology. For example, the atomic measure “Number of fraudulent claims” can be defined as:

```
NumberOfFraudulentClaims=
```

```
if(Claim.settlementStatus="FRAUDULENT") COUNT(Claim)
```

which means that the value of this atomic measure is calculated by counting the instances of the class *Claim* that satisfy the condition that the settlement status is fraudulent. Another atomic measure example is “the overall days of settlement for Individual Insurance”, which can be defined as:

```
TotalDaysOfSettlement=
```

```
if(Claim.ofProduct.category="INDIVIDUAL")
```

```
SUM(Claim.settlementTime.date - Claim.requestTime.date)
```

The complex measure can be defined by a function on other measures: i.e.  $m = f(m_1, \dots, m_n)$ , where  $m_i$  is an atomic measure or complex measure. The function expression supported in EIAW is the four arithmetic operations: addition, subtraction, multiplication and division. For example, the complex measure “Fraudulent Claims Ratio” can be defined as:

```
FraudulentRatio=NumberOfFraudulentClaims/NumberOfClaims
```

The grammar of the script language to define measures is shown in **Table 1**. The formal semantics of the language is out of the scope of this paper.

**Table 1.** Parts of the BNF grammar for measure definition

```

Literal := String | Number
PEExpr := PropertyPathExpr | OWLClassExpr
BOper( Boolean Operator ):= "and" | "or" | "not"
COper( Compare Operator ):= "<" | "<=" | "=" | ">=" | ">"
AOper( Arithmetic Operator ):= "+" | "-" | "*" | "/"
Func (Predefined Functions):= currentYear() etc.
// currently only support functions in SQL
Expr ::= Expr AOper Expr | Func "(" ArgLst ")"
      | PEExpr | Literal
ArgLst ::= Expr ("," Expr)*
CondExpr(Condition Expression)::=
      CondExpr(BOper CondExpr)* | Expr COper Expr
AggFunc(Aggregation Function):= "SUM" | "COUNT" | "AVG"
Statement ::= AggFunc "(" Expr ")"
MsrName ::= String
AmDef(atomic measure) := MsrName "=" "if" CondExpr
Statement ("else if" CondExpr Statement)*["else" Statement]
CmDef(Complex measure) :=
      MsrName "=" MsrName (AOper MsrName)*

```

### 3.2.2 Representation of Dimensions

In conventional approach, dimension in multidimensional model consists of a set of dimension levels with a partial order on the dimension levels to support roll-up and drill-down analysis, called level-based hierarchy. To support the OLAP analysis on the data mart, the multidimensional model is strictly adhered to the schema of the data mart. For example, every dimension level is correspondent to a column in the dimension table of the data mart, and the values of every dimension level are stored as values in the correspondent column. The strong dependency between multidimensional model and the data mart make it difficult to modify the multidimensional model to meet the changed analysis requirements. If business user wants to re-organize the dimension levels, the whole deployment process need to be redone manually, including the re-design the data mart schema and the ETL processes from EDW to data mart.

To make the data warehouse more adaptable to the business change, EIAW also supports the user-definable value-based hierarchy. In EIAW, a dimension is a virtual property, whose range is an enumeration of nodes organized as a tree-like hierarchy (the multiple hierarchy is not supported yet), called value-based hierarchy. For each node in the value-based hierarchy, it is denoted by a string literal and attached with a condition expression that defines the meaning of the node. The condition expression has the same expressivity as in the measure definition.

For example, the dimension “the professional risk level of the insured person” is defined as in the **Table 2** and **Table 3**.

**Table 2.** The node and its definition for the dimension

*ProfessionalRiskLevelOfInsured=*

Value	Condition Expression
“Level 1”	Insured.playedBy[Person].profRiskLevel=1
“Level 2”	Insured.playedBy[Person].profRiskLevel=2
“Level 3”	Insured.playedBy[Person].profRiskLevel=3
“Level 4”	Insured.playedBy[Person].profRiskLevel=4
“Level 5”	Insured.playedBy[Person].profRiskLevel=5
“High”	Insured.playedBy[Person].profRiskLevel<3
“Medium”	Insured.playedBy[Person].profRiskLevel=3
“Low”	Insured.playedBy[Person].profRiskLevel>3

**Table 3.** The value-based hierarchy for the dimension

The root node	The first level	The second level
ALL	“Low”	“Level 1”
		“Level 2”
	“Medium”	“Level 3”
	“High”	“Level 4”
		“Level 5”

where the *Insured.playedBy[Person].profRiskLevel* is a property path expression that denotes the professional risk level of the insured person.

The main advantage for user-definable value-based hierarchy is the decoupling of business semantics and technical implementation. Business user can modify the value-based hierarchy by business terms without consideration of the physical storage of data mart.

The UI for editing ASM is shown in Fig. 3. Business user can define measures by formulas, and define value-based hierarchy for dimensions using the business terms from BCM.

## 4 Mapping

To enable automatic deployment of data mart for an ASM, information must be provided on where to retrieve the data and what calculations should be done for the dimensions and the measures. Because the business semantics for measures and dimensions are formally defined by the business terms (including property path expressions) from BCM, IT users only need to build the mappings from the business terms to the EDW schema to resolve the technical issues, and then the deployment



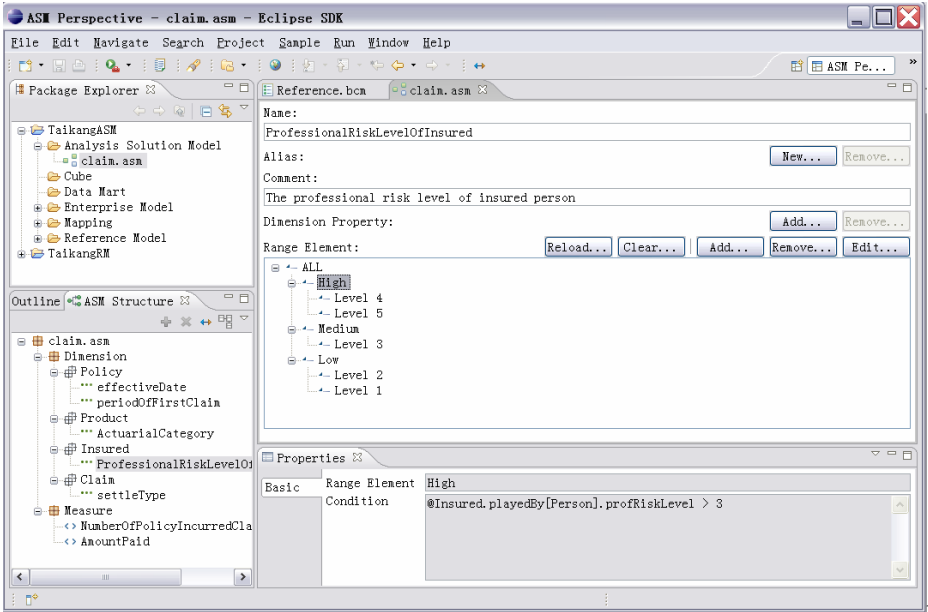


Fig. 3. The ASM Editor

engine will interpret the business semantics and automatically generate the data mart schema and loading data into the data mart.

The mapping from business terms to EDW schema is a set of mapping rules with 4-ary tuples as:  $\langle \text{SOURCE}, \text{TARGET}, \text{CONDITION}, \text{TRANSLATION} \rangle$ , and can be written in a textual form as:

SOURCE :- TARGET WHERE CONDITION WITH TRANSLATION.

The SOURCE can be a single class or a property path. The TARGET is the schema path expression in the EDW schema. A schema path expression is a column in the logical table connected by the join operator. It can be written as:

$$Table_1.fk_1[Table_2], \dots, fk_{n-1}[Table_n].column$$

The CONDITION is the definition of under what condition for the target the mapping is correct, similar to the WHERE clause in SQL, and the TRANSLATION is a function that translate the values for ontology property to the data values for columns in the EDW.

For example, the property path *Insured.playedBy[Person].profRiskLevel* can be mapped to:

```
Insured.playedBy[Person].profRiskLevel :-
F_PLCY_EVT.PLCY_PTCP_ID[D_CUST].CUST_PROF_ID[D_PROFESSI
ON].RISK_LVL
```

The mapping between EDW schema and ontology make the semantics of data in EDW explicit. The intended meaning of the mapping rule is to explain how to retrieve

the data for the business term. For example, the above mapping tells the deployment engine to get the data about the professional risk level of insured person by starting from the fact table F\_PLCY\_EVT, following the foreign keys PLCY\_PTCP\_ID (Policy Participant ID) and CUST\_PROF\_ID (Customer’s Professional ID) to the column RISK\_LVL to get the required data.

The mapping editor is shown in Fig. 4, which supports IT users to map the business terms appeared in ASM definitions to the EDW schema.

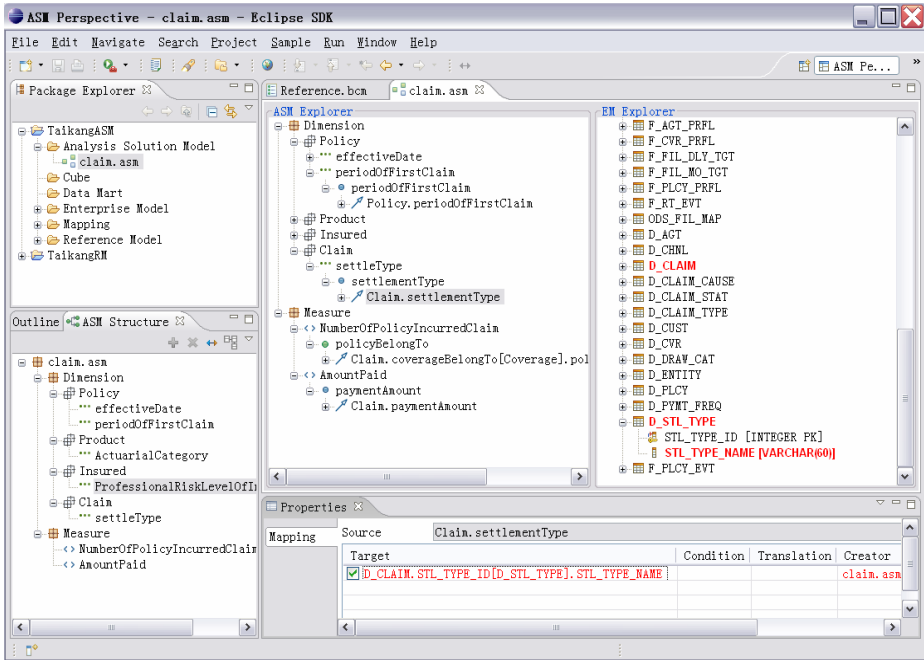


Fig. 4. The Mapping Editor. It consists of three views: the *ASM Explorer* displays the business terms for the definition of the ASM, the *EM Explorer* displays the EDW schema, the *Mapping properties view* enables the editing of the mapping formula.

## 5 Deployment Engine

The deployment engine automatically generates a data mart from the EDW, given the definition of the ASM and the mappings. The data mart is a customization of the EDW in that it is a subset of data from the EDW according to the analysis requirements defined in ASM. The deployment procedure includes two steps: firstly generate schema of target data mart derived from original data warehouse according to the ASM and mapping. The schema will be organized in form of star schema. Secondly, all the needed data will be loaded into the target data mart by the Date Engine. Here we only introduce the basic idea (shown in Fig. 5) and ignore the technical details due to the limitation of paper length.

Currently, we assume that the EDW schema is organized as a star schema or snowflake schema. In such case, the data loading problem for target data mart can be simplified as a problem of pruning unused data, instead of re-structuring data in the EDW. It is a strict assumption because some EDW schemas are 3-NF schemas. However, it is a good starting point for a quick validation for the whole idea.

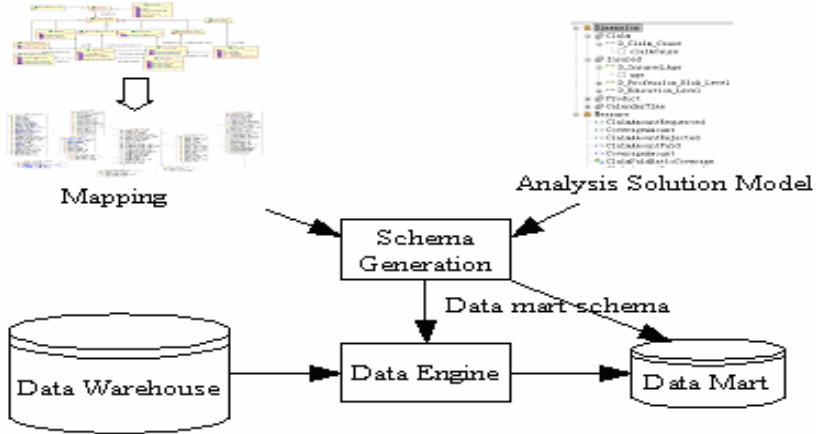


Fig. 5. The deployment of data mart

**Step 1: Generating data mart schema.** The goal of this step is to generate a physical star-schema for the target data mart. For every dimension in the ASM, the system will create a dimension table whose columns are the columns in EDW schema that appeared in the mapping for the business terms that defines the value-based hierarchy for the dimension. The system will create a fact table with one column as the surrogate primary key, some columns for every measure and some foreign key columns to the dimension tables.

**Step 2: Loading data into the data mart.** Once the target data mart schema is generated, given the mapping, the data engine will interpret the definition of ASM and extract relevant data into the dimension tables, aggregate data for the atomic measures and make calculations for the complex measures based on their definitions.

## 6 Tool Deployment and User's Feedback

EIAW is implemented as a plug-in on the Eclipse 3.2 and its internal operation on the BCM and ASM is based on the Integrated Ontology Development Toolkit (IODT)<sup>2</sup>. EIAW was deployed to the Taikang Life Insurance Company as a result of a collaborative project between IBM and Taikang Life. The work on this tool is also invited to present on the IBM Financial Services Solutions Symposium (FS3 2007)<sup>3</sup>.

<sup>2</sup> IODT, <http://www.alphaworks.ibm.com/tech/semanticstk>

<sup>3</sup> <http://www.ibm.com/financialservices/symposium>

The experimental data warehouse is organized as a star-schema, consisting of 5 fact tables and 38 dimension tables. The biggest table is the fact table about policy event with about 53 millions rows of records. The other largest tables are the dimension tables about coverage, customer and policy, with 30, 17 and 8 millions rows of records respectively. The overall data size is about 250 gigabytes.

Currently the built-in BCM is a customization to the business model in the industry reference model: Insurance Information Warehouse (IIW)<sup>4</sup>. The business model in IIW is originally an ER model with 430 entities, 437 relationships and 884 attributes. We manually selected the elements covered by the data in the data warehouse and transformed them to an OWL ontology, which consists of 76 classes, 48 object properties and 67 datatype properties. We created one ASM for claim analysis, agent performance analysis and financial analysis respectively with about average 6 dimensions and 11 measures.

In general, users are very impressed by the definitions of the measures and dimensions using business terms from OWL ontology. They also find that the mapping efforts can be dramatically reduced because only the mapping for resolving technical issues are needed and others are represented as the definitions of measures and definitions. For example, in a Financial Analysis Project, there are above 30 measures defined using only three property paths, so the only mappings are for the three property paths, other than for the 30 measures in the conventional approach. Though they need to define the 30 measures, rather than provide a simple textual name as in the conventional approach, but they indicated that the formal definitions help to clarify the meaning of measures and can improve the quality of multidimensional model. They even further request an approach for consistency checking for multidimensional model, which needs a complete formalization of the extended language and complex reasoning technologies. Another encouraging feedback is that the formal definition with less ambiguity helps for the reuse. Since the purpose of ontology is to be shared and reused for multiple applications, the measures and dimensions defined on the ontology, and the mappings for vocabularies in ontology can also be reused for multiple BI applications.

They also indicate that the tool provides an amazing solution for adapting to analysis changes for data mart deployment. After the mapping is built, they can further modify the multidimensional model, such as adjusting the value hierarchy for dimension and modifying the definitions of measures, and then re-deploy the data mart. They also suggest that the data mart can be generated incrementally if just a minor modification to the multidimensional model is made.

## 7 Related Works

There has been continuous works on designing and modeling of multi-dimensional mode from conceptual level [3,5,7,11]. They represent measure as a function from a set of dimension names to a data value, while dimension as a set of dimension levels with a partial order on the dimension levels to support roll-up and drill-down analysis. However, they do not further define the intended meaning of measures and

---

<sup>4</sup> <http://www-03.ibm.com/industries/financialservices/doc/content/solution/278652303.html>

dimensions in terms of business terms. Actually, the measures and dimension in conventional modeling approach are just textual names with descriptive information in natural language. Instead, measures and dimensions in our tool are formally defined by the business terms from the ontology using an expressive language.

The idea of describing measures and dimensions using vocabularies in conceptual model is not completely new. Muller et al. [8] proposed an approach to use UML as a uniform language for all the business metadata, including the conceptual model, the multidimensional model and the dependencies between these two models. However, the representation of business metadata is too coarse-grained to enable the automatic deployment of data mart. There are also attempts to extend the Description Logics language with multidimensional aggregation. Baader and Sattler [1] explored the extension of different Description Logics (DL) languages by concrete domains and aggregation functions over these domains and studied the decidability of satisfiability problem in these extended languages. Franconi and Sattler [4] further proposed a Data Warehouse Conceptual Data Model which allows for the description of both the relevant aggregated entities of the domain and the relevant dimensions involved in building the aggregated entities, based on DL. In EIAW, the language is much less complex and more application-specific. For example, the property path expression and the measures are not allowed to participate in the definition of class expression. But their works provide a good reference on the formal grounding for our proposed language. We also noticed that “property chain” is introduced in the recent OWL 1.1 proposal, but the property path expression is more expressive by introducing the notion of constraints on domain and range of properties, i.e.  $C.R$  and  $R_{[D]}$ .

There are also growing interests in the introducing of Semantic Web technologies into the area of Data Warehousing and Business Intelligence. Skoutas[13] showed the usage of ontologies to enable a high degree of automation regarding the construction of an ETL design for data warehousing. Sell et al.[12] proposed a Semantic Web based architecture for analytic tools, in which the domain ontology are used to rewrite the conditions of the query to the data warehouse, in order to broaden the results of a query and to support inferences over the results of the queries. In addition, there will be a special issue on Semantic Web and data warehousing published in the International Journal of Semantic Web and Information Systems<sup>5</sup>.

## 8 Conclusions

Similar to the trend that business rules and business processes are isolated from the programming codes; business semantics for data warehousing is also needed to be isolated from the data warehouse based BI application implementation and explicitly represented using a formal language. The explicit business semantics enables business users organize the business knowledge and express their analysis requirements, and enable IT users only build the mappings due to the technical issues.

Based our practices, we think OWL extended by some constructs needed for real applications, such as concrete domain, predefined functions and property path expression, is a good candidate language for expressing the business semantics for

---

<sup>5</sup> The Call For Paper link: <http://www.ijswis.org/cfp/semwebandwarehousing.html>

data warehousing system. However, a complete formalization of the extended language is still an open problem due to the complexity of measure and dimension definitions.

## References

1. Baader, F., Sattler, U.: Description logics with concrete domains and aggregation. In: ECAI 1998. Proceeding of the 13th European Conference on Artificial Intelligence, pp. 336–340 (1998)
2. Berners-Lee, T., Handler, J., Lassila, O.: The Semantic Web. *Scientific American* 184(5), 34–43 (2001)
3. Cabbibo, L., Torlone, R.: A Logical Approach to Multidimensional Databases. In: Proceeding of the 6th International Conference on Extending Database Technology, pp. 183–197 (1998)
4. Franconi, E., Sattler, U.: A Data Warehouse Conceptual Data Model for Multidimensional Aggregation. In: DMDW 1999. Proceeding of Workshop on Design and Management of Data Warehouses, Heidelberg, Germany (1999)
5. Golfarelli, M., Maio, D., Rizzi, S.: The Dimensional Fact Model: a Conceptual Model for Data Warehouses. *International Journal of Cooperative Information Systems* 7(2,3), 215–247 (1998)
6. McComb, D.: *Semantics in Business Systems: The Savvy Manager's Guide*. Morgan Kaufmann, San Francisco (2003)
7. Moody, D.L., Kortink, M.A.R.: From enterprise models to dimensional models: A methodology for data warehouse and data mart design. In: DMDW 2000. Proceeding of the 2nd International Workshop on Design and Management of Data Warehouses (2000)
8. Muller, R., Stohr, T., Rahm, E.: An Integrative and Uniform Model for Metadata Management in Data Warehousing Environments. In: Proceeding of International Workshop on Design and Management of Data Warehouses, pp. 12–28 (1999)
9. O'Neil, B.: Semantics and business metadata. *The Data Administration Newsletter*(TDAN.com) (2007)
10. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL Web Ontology Language Reference, W3C Recommendation (February 10, 2004) (2004), <http://www.w3.org/TR/owl-ref/>
11. Sapia, C., Blaschka, M., Höfling, G., et al.: Extending the E/R Model for the Multidimensional Paradigm. In: Kambayashi, Y., Lee, D.-L., Lim, E.-p., Mohania, M.K., Masunaga, Y. (eds.) *Advances in Database Technologies*. LNCS, vol. 1552, Springer, Heidelberg (1999)
12. Sell, D., Cabral, L., Motta, E., et al.: A Semantic Web based Architecture for Analytical Tools. In: CEC 2005. Proceeding of the Seventh IEEE International Conference on E-Commerce Technology, pp. 347–354 (2005)
13. Skoutas, D., Simitis, A.: Designing ETL processes using semantic web technologies. In: Proceeding of the 9th ACM international workshop on Data warehousing and OLAP, pp. 67–74 (2006)
14. Stam, A.: A Successful Enterprise Data Warehouse? The Four Biggest Reasons for Failure. *BI Review Newsletter* (2007)

# GroupMe! - Where Semantic Web Meets Web 2.0

Fabian Abel, Mischa Frank, Nicola Henze, Daniel Krause,  
Daniel Plappert, and Patrick Siehndel

IVS – Semantic Web Group, University of Hannover, Hannover, Germany  
{abel, frank, henze, krause, plappert, siehndel}@kbs.uni-hannover.de

**Abstract.** Grouping is an attractive interaction metaphor for users to create reference collections of Web resources they are interested in. Each grouping activity has a certain semantics: things which were previously unrelated are now connected with others via the group. We present the GroupMe! application which allows users to group and arrange multimedia Web resources they are interested in. GroupMe! has an easy-to-use interface for gathering and grouping of resources, and allows users to tag everything they like. The semantics of any user interaction is captured, transformed and stored as adequate RDF descriptions. As an example application of this automatically derived RDF content, we show the enhancement of search for tagged Web resources, which evaluates the grouping information to deduce additional contextual information about the resources. GroupMe! is available via <http://www.groupme.org>.

## 1 Introduction

The success of the so-called Web 2.0 has shown that people enjoy features like tagging or collaborative spaces. Community platforms (e.g. wikis, blogs, social bookmarking systems) provide users with high *intercreativity* and form some kind of *collective intelligence*. Although some platforms offer public interfaces to access the community knowledge, sharing knowledge across community boundaries is still limited. One of the reasons for this can be attributed to the fact that Semantic Web technologies are rarely used here. GroupMe! combines approaches from these two areas.

GroupMe! extends the idea of social bookmarking systems like *del.icio.us*<sup>[1]</sup> and systems that allow (re-)organizing web content like *combinFormation* [2] in many aspects: (1) users are able to build groups of arbitrary (multimedia) Web resources by simple drag & drop operations, (2) resources contained in such groups can be (re-)arranged by the users (the visualization of the resources is adapted to content type), and (3) the grouping and tagging activities produce RDF descriptions in an easy and collaborative way (e.g. dropping a Web resource into a group effects the transformation of the resource's community-specific attributes into attributes adhering to common ontologies). RDF content

---

<sup>1</sup> <http://del.icio.us>

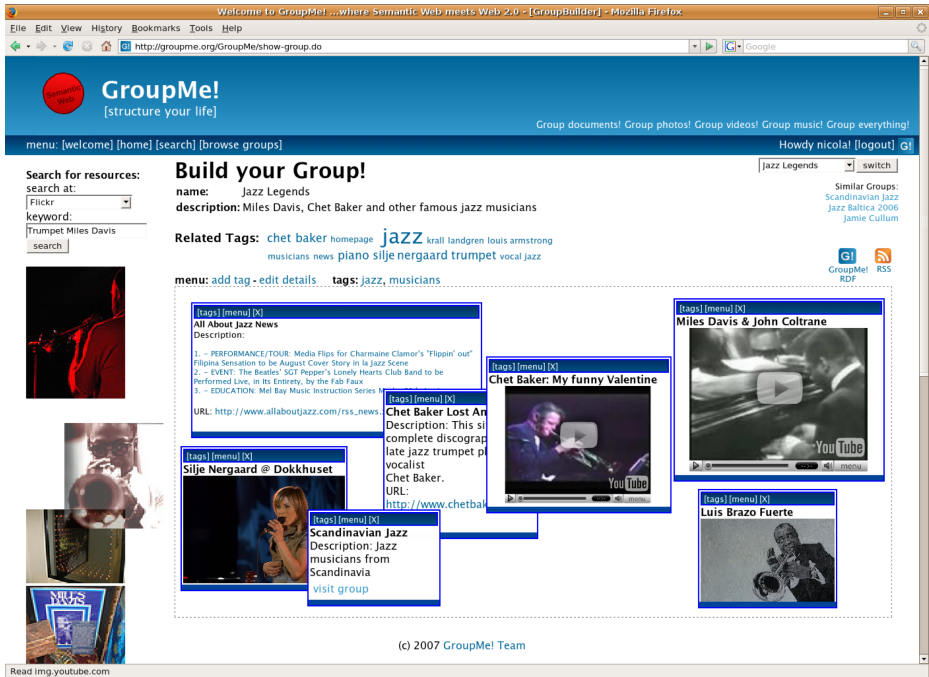


Fig. 1. Screenshot: group builder of the GroupMe! application

that is created in GroupMe! is given back to the Web via an *RESTful* [2] API using well known vocabularies like *FOAF* [2], *RSS* [3], or *DCMI element set* [4], and a GroupMe! vocabulary, which captures new concepts like groups, relation of groups and resources, etc.

The GroupMe! approach provides the possibility to overcome the gap between Web 2.0 and the Semantic Web, and offers with the grouping facility new and interesting strategies for search and content discovery. In the following section, we introduce the GroupMe! application with an example scenario. Afterwards, we discuss a context-aware search strategy that takes grouping information into account. Section 4 briefly discusses technical issues of the GroupMe! architecture, and Section 5 summarizes the paper.

## 2 The GroupMe! Approach - Scenario

Figure 1 shows the group builder functionality of GroupMe!. In the illustrated example the user is building a group which he tags with *Jazz Legends*. The search form on the left hand side in Fig. 1 enables the user to utilize several search

<sup>2</sup> <http://xmlns.com/foaf/spec/>

<sup>3</sup> <http://web.resource.org/rss/1.0/>

<sup>4</sup> <http://dublincore.org/documents/dces/>



engines for gathering resources of interest (in arbitrary format). With a simple *drag & drop* operation he adds suitable resources to the group (right hand side in Fig. 1), and arranges them within the group as he likes. In the depicted scenario the user has already added some resources to the group: two YouTube<sup>5</sup> videos of live performances, two Flickr<sup>6</sup> photos of musicians, a news feed, a website about *Chet Baker*, and a GroupMe! group, which deals with *Scandinavian Jazz*. All these resources are visualized according to their content type, e.g. the news feed displays the three latest news items, videos are embedded, etc. Every time the user adds a resource to a group, GroupMe! produces RDF: Metadata is extracted from the aggregated resources (e.g. a Flickr image title is transformed to the DCMI element *title*, etc.), and the grouping activities are captured as RDF (cf. section 4.2). Produced RDF can immediately be accessed via our API or static links (see buttons “GroupMe! RDF” and “RSS”).

Besides grouping and arranging resources within groups, the user can tag resources and groups. These tags are used to provide enhanced navigation possibilities: by clicking on tags of the group-specific *tag cloud* (see *related tags* in Fig. 1), GroupMe! lists matching resources and groups, and by clicking on a similar group (see top right in Fig. 1) the user is directed to the selected group.

### 3 Folksonomies and Search Strategies

#### 3.1 GroupMe! Folksonomies

The core data of GroupMe! evolves over time by tagging resources and groups, and by grouping resources and (re-)arranging them. Tagging is done by users (*folks*), and results in a collection of concepts (“*taxonomy*”) that is called *folksonomy*. In 3 a folksonomy is formally defined as follows:

**Definition 1 (Folksonomy)**

A folksonomy is a tuple  $\mathbb{F} := (U, T, R, Y, \prec)$ , where:

- $U$ ,  $T$  and  $R$  are finite sets that contain instances of users, tags and resources
- $Y$  defines a relation (tag assignment) between these sets:  $Y \subseteq U \times T \times R$
- $\prec$  defines a user-specific subtag/supertag-relation between tags:  $\prec \subseteq U \times T \times T$

The GroupMe! application uses an adapted version of definition 1. Users are able to tag resources (*tag assignment*), and  $\prec$ -relations can in principle be deduced using ontology learning techniques 4. In GroupMe!, we understand the notion of *resources* in a general sense, e.g. a resource can either be a resource or a more complex object: a set of resources.

**Definition 2 (Group).** A group is a set of resources.

As a group is a resource as well, groups can contain groups (which was e.g. the case in the scenario described in Section 2). With this definition of groups, we extend definition 1 to be able to tag resources and sets of resources (groups):

<sup>5</sup> <http://www.youtube.com>

<sup>6</sup> <http://www.flickr.com>

**Definition 3 (Extended Folksonomy)**

An extended folksonomy is a tuple  $\mathbb{F} := (U, T, \check{R}, Y, \prec)$ , where:

- $\check{R} = R \cup G$  (the union of the set of resources  $R$  and the set of groups  $G \subseteq 2^R$ )
- consequently the definition of  $Y$  is replaced by:  $Y \subseteq U \times T \times \check{R}$ ,  $\prec$  analogously.

According to Fig. 1, the tag assignment for group *Jazz Legends* can be modeled with the following statements:

```
(gm:nicola, gm:jazz, gm:jazzLegendsGroup)
(gm:nicola, gm:musicians, gm:jazzLegendsGroup)
```

Furthermore, resources can be tagged in context of a certain group. This enables us to gain additional knowledge about the tag assignment. The definition of a GroupMe! folksonomy, which considers such a group context, is formalized as:

**Definition 4 (GroupMe! Folksonomy)**

A GroupMe! folksonomy is a tuple  $\mathbb{F} := (U, T, \check{R}, G, \check{Y}, \prec)$ , where:

- $\check{Y}$  defines the extended tag assignment:  $\check{Y} \subseteq U \times T \times \check{R} \times G$ ,  $\prec$  analogously

By introducing the group context in addition to the  $\prec$ -relation, we obtain other kinds of relations between tags which can be used for many purposes (to deduce tags for untagged resources, to derive a "neighborhood" of a tag, to mine frequently occurring neighborhoods, etc.). To continue our example, allocating tags to the resources grouped in Fig. 1 is expressed via:

```
(gm:nicola, gm:trumpet, gm:chetBakerVideo, gm:jazzLegendsGroup)
(gm:nicola, gm:jazz, gm:chetBakerVideo, gm:jazzLegendsGroup)
(gm:nicola, gm:vocalJazz, gm:siljeNergaardPhoto, gm:jazzLegendsGroup)
...
```

**3.2 Search Strategies**

When searching for resources by providing a set of query terms, the GroupMe! application considers not only the tags that are directly assigned to a resource by a user, but also contextual information of resources. Different strategies, which have potential to enhance search strategies as proposed in [5], are possible. Fig. 2 shows an example of such a strategy.

- (1) In the first step, all resources that are directly tagged with one of the query terms are collected. The weight which is associated with these fitting resources depends on the number of users that tagged resource  $r$  with tags contained in  $T_{query}$ , e.g.:

$$directWeight(T_{query}, r) = \sum_{t \in T_{query}} resourceWeight(t, r), \text{ where}$$

$$resourceWeight(t, r) = \frac{\text{number of users who tagged resource } r \text{ with } t}{\text{number of users who tagged resource } r}$$

```

rankResources( $T_{query}$ ):
   $T_{query}$ : tags which represent the search query;
   $\check{R}_{result}$ : set of fitting resources;
   $G_{temp}$ : set of fitting groups; //it is:  $G_{temp} \subseteq \check{R}_{result}$ 
  (1) for each resource  $r \in \check{R}$  :
    if ( $\exists t \in T_{query}$ :  $r$  is tagged with  $t$ ):
       $r.directWeight = directWeight(T_{query}, r)$ ;
       $\check{R}_{result} = \check{R}_{result} \cup \{r\}$ ;
      if ( $r \in G$ ):
         $G_{temp} = G_{temp} \cup \{r\}$ ;
  (2) for each group  $g \in G_{temp}$  :
    for each resource  $r \in g$  :
       $n = |\{g' \in G_{temp} | r \in g'\}|$ ;
       $r.contextWeight += contextWeight(T_{query}, r, g) \cdot \frac{1}{n}$ ;
       $\check{R}_{result} = \check{R}_{result} \cup \{r\}$ ;
  (3) for each  $r \in \check{R}_{result}$ :
     $r.weight = \alpha \cdot r.directWeight + \beta \cdot r.contextWeight$ ;
  (4)  $ranking = order \check{R}_{result}$  by  $r.weight$ ;
  return  $ranking$ ;

```

**Fig. 2.** Search strategy which considers the group context

- (2) Afterwards all *fitting groups* (i.e. those groups that are tagged with at least one of the query terms) are determined, and the resources in these groups are weighted according to their appearances in groups. There are a several ways to compute such *context weight*, e.g.:

$$contextWeight(T_{query}, r, g) = \sum_{t \in T_{query}} resourceWeight(t, r) \cdot groupWeight(t, g),$$

$$\text{where } groupWeight(t, g) = \frac{\text{number of resources in } g \text{ that are tagged with } t}{\text{number of resources in } g}$$

As a resource may appear in several groups  $g \in G_{temp}$ , we compute the average of the corresponding *contextWeights*.

- (3) In the next step, the values of both weights are combined into an *overall weight*.  $\alpha$  or  $\beta$  can be used to emphasize either direct or context weight.
- (4) Finally, all found resources are ranked according to their overall weight and returned as an ordered list.

## 4 The GroupMe! System

At a glance the GroupMe! system provides three core functionalities for users:

- 1. Creation of groups.** Users can add arbitrary Web resource to a group (including other groups), arrange them within the group, and tag both resources and groups.

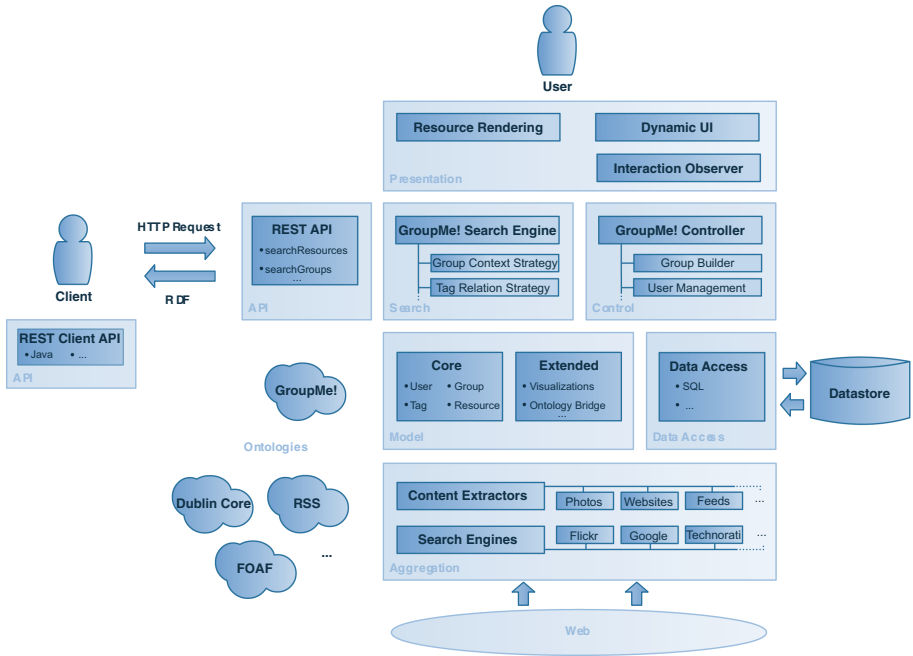


Fig. 3. Technical overview of the GroupMe! application

- 2. **Search and navigation.** Ranking strategies that utilize information about tags and groups are applied for search and navigation facilities.
- 3. **RDF export.** RDF content which is produced within the process of creating groups and arranging resources within groups is made available to the public.

#### 4.1 Architecture

The architecture of the GroupMe! application is outlined in Fig. 3. In technical terms, GroupMe! is a Web application, which adheres to the Model-View-Controller pattern. It consists of four basic layers:

**Aggregation.** The aggregation layer provides functionality to search for resources that should be included into GroupMe! groups (*Search Engines*). *Content Extractors* allow us to process gathered resources in order to extract useful metadata and convert them into RDF resources with semantically well defined descriptions. When e.g. adding a result from the Flickr search engine into a group, a *Photo content extractor* converts Flickr-specific descriptions into a well defined RDF description using Dublin Core vocabulary.

**Model.** The core GroupMe! model is composed of four main concepts: *User*, *Tag*, *Group* and *Resource*. These concepts constitute the base for the GroupMe! folksonomy (cf. section 3). In addition, the model covers concepts concerning the users' arrangements of groups, etc. The *Data Access* layer cares about storing model objects.



Fig. 4. Example: process of RDF content creation

**Application logic.** The logic layer provides various controllers for modifying the model, exporting RDF, etc. The internal GroupMe! search functionality is made available via a RESTful API. It enables third parties to benefit from the improved search capabilities, and to retrieve RDF descriptions about resources, even such resources that were not equipped with RDF descriptions before they were integrated into GroupMe!. To simplify the usage of exported RDF data, we further provide a lightweight Java *Client API*, which transforms RDF into GroupMe! model objects.

**Presentation.** The GUI of the GroupMe! application is based on AJAX [7] principles, and is highly modular and extensible. For example, the visualization of group elements is adapted to the content type (see Fig. 1). When creating or modifying groups, each user interaction (e.g. moving and resizing resources) is monitored and immediately communicated to the responsible GroupMe! controller with the effect that e.g. the actual size or position of a resource within a group is stored.

## 4.2 Ontologies and Content Creation

Almost every user interaction with the GroupMe! systems implies the creation of RDF content. Figure 4 illustrates the general process of RDF content creation. On the left the original resource – a Flickr photo in this example – is presented. It is described with a community-specific vocabulary instead of a well defined ontology. Hence, when integrating the resource into GroupMe!, this deficit has to be compensated. To do so, we use ontologies consistent with the type of resource. The photo in Fig. 4 can for the most part be described by applying Dublin Core metadata elements. For example, *dc:subject* represents the tag that was provided by Flickr. Which ontology to apply depends highly on the resource type and further on the content provider. To capture the process of grouping and tagging we rely on the *GroupMe! ontology* [8] which essentially models the *GroupMe! folksonomy* as defined in definition 4.

**User ( $U$ ):** GroupMe! users are simply modeled as *rdfs:subClassOf* of *foaf:Person*.

**Resources ( $\check{R}$ ):** Resources have at least *dc:title* and *resourceURL* properties. Additionally they can be equipped with attributes of any other domain ontology. Grouping of resources is modeled via the object property *isInGroup*,

<sup>7</sup> <http://www.adaptivepath.com/publications/essays/archives/000385.php>

<sup>8</sup> <http://groupme.org/rdf/groupme.owl>

which points to the *Group* instances the resource is included in. For tagging of resources the GroupMe! ontology provides an object property named *tagAssignment*, which refers to *TagAssignment* instances.

**Groups (*G*):** *Group* is a subclass of *Resource* and extends its superclass with an inverse property of *isInGroup*, namely *hasResource*.

**Tag (*T*):** The class *Tag* defines a functional property *keyword*. As future versions of GroupMe! should also aim on bridging from folksonomies to taxonomies, *Tag* is also equipped with a property *relatesToConcept*, which should refer to such concepts of domain ontologies that are denoted by the corresponding *Tag* instance.

**TagAssignment (*Y*):** This concept implements the assignment of a tag by a user within the context of a group and consequently has four object properties: *user*, *tag*, *resource* (inverse of *tagAssignment*), and *group*.

The result of applying the GroupMe! ontology is shown shortened in the right box of Fig. 4. Such semantically enriched resources, which evolve naturally while users are interacting with the GroupMe! system, can in turn be processed by other systems via the GroupMe! API.

## 5 Conclusions

The GroupMe! application is at the edge between Web 2.0 and Semantic Web and enables users to easily group and arrange multimedia resources they are interested in. We believe that this kind of interaction is enjoyable to use and will, in combination with the automatic capturing of the semantics of the users interactions, support the wide-spread use of RDF.

## References

1. Kerne, A., Koh, E., Dworaczyk, B., Mistrot, J.M., Choi, H., Smith, S.M., Graeber, R., Caruso, D., Webb, A., Hill, R., Albea, J.: combinFormation: A mixed-initiative system for representing collections as compositions of image and text surrogates. In Marchionini, G., Nelson, M.L., Marshall, C.C., eds.: JCDL, ACM (2006) 11–20
2. Fielding, R.T., Taylor, R.N.: Principled design of the modern web architecture. In: Proc. of ICSE '00, New York, USA, ACM Press (2000) 407–416
3. Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: BibSonomy: A social bookmark and publication sharing system. In de Moor, A., Polovina, S., Delugach, H., eds.: Proc. of ICCS '06, Aalborg (2006) 87–102
4. Cimiano, P., Pivk, A., Schmidt-Thieme, L., Staab, S.: Learning taxonomic relations from heterogeneous sources of evidence. In: Ontology Learning from Text: Methods, Evaluation and Applications. Frontiers in AI. IOS Press (2005) 59–73
5. Bao, S., Xue, G., Wu, X., Yu, Y., Fei, B., Su, Z.: Optimizing web search using social annotations. In: Proc. of WWW '07, New York, USA, ACM Press (2007) 501–510

# CHIP Demonstrator: Semantics-Driven Recommendations and Museum Tour Generation

Lora Aroyo<sup>1,2</sup>, Natalia Stash<sup>1</sup>, Yiwen Wang<sup>1</sup>,  
Peter Gorgels<sup>3</sup>, and Lloyd Rutledge<sup>4</sup>

<sup>1</sup> Eindhoven University of Technology, Computer Science  
{n.v.stash,y.wang}@tue.nl

<sup>2</sup> VU University Amsterdam, Computer Science  
l.m.aroyo@cs.vu.nl

<sup>3</sup> Rijksmuseum Amsterdam  
p.gorgels@rijksmuseum.nl

<sup>4</sup> Telematica Institute  
Lloyd.Rutledge@cwii.nl

## 1 Introduction

The main objective of the CHIP project is to demonstrate how Semantic Web technologies can be deployed to provide personalized access to digital museum collections. We illustrate our approach with the digital database ARIA of the Rijksmuseum Amsterdam<sup>1</sup>. For the semantic enrichment of the Rijksmuseum ARIA database we collaborated with the CATCH STITCH project<sup>2</sup> to produce mappings to Iconclass<sup>3</sup>, and with the MultimediaN E-culture project<sup>4</sup> to produce the RDF/OWL of the ARIA and Adlib databases. The main focus of CHIP is on exploring the potential of applying adaptation techniques to provide personalized experience for the museum visitors both on the Web site and in the museum. This resulted in three demonstrator components:

- Artwork Recommender - a Web-based rating dialog to build a user profile, based on semantics-driven recommendations.
- Tour Wizard - a Web-based tool using the user profile to generate automatically personalized museum tours for each user, and to (semi)-automatically generate various personalized routes through the digital Rijksmuseum collection.
- Mobile Tour - a PDA-based tool, which uses the results from the Tour Wizard and helps users navigate and discover artworks in the physical Rijksmuseum environment.

---

<sup>1</sup> <http://rijksmuseum.nl/aria/>

<sup>2</sup> <http://www.cs.vu.nl/STITCH/>

<sup>3</sup> <http://www.iconclass.nl/libertas/ic?style=index.xml>

<sup>4</sup> <http://e-culture.multimediana.nl/>

The online version of the CHIP demonstrator as well as a tutorial with a brief walk-through of the personalization functionality can be found at: <http://www.chip-project.org/demo/>.

Further, we give a short introduction to the basic functionality of the Web-based parts of the CHIP demonstrator. Please note that the CHIP project collects feedback, on the functionality and usability of the demonstrator, on a regular basis from studies with museum visitors. Thus, the demonstrator changes over time as we are incorporating more functionalities and improvements to the interface.

## 2 Usage Scenario: You Rate - We Recommend

In fig. 1 we illustrates how we employ semantics in building user profiles and using them for generating recommendations to the user, as a way of guiding users through the museum collection. In the *Artwork Recommender*, the user rates an artwork and several properties:

- artwork *Night Watch* - 4 stars (i.e. ‘I like *Night Watch*’);
- creator property *Rembrandt* - 4 stars (i.e. ‘I like *Rembrandt*’);
- theme property *Landscape* - 4 stars (i.e. ‘I like *landscape*’);
- theme property *Self-portrait* - 1 star (i.e. ‘I hate *self-portrait*’).

The User Profile stores the user’s ratings for generating recommendations. Now, let’s see how does the artwork and topic recommendation in CHIP work:

- Find all *Night Watch*-related properties, e.g. creator, creation place, creation year, material and themes
- Find all *Rembrandt*-related properties, e.g. style, teacher-of and student-of.

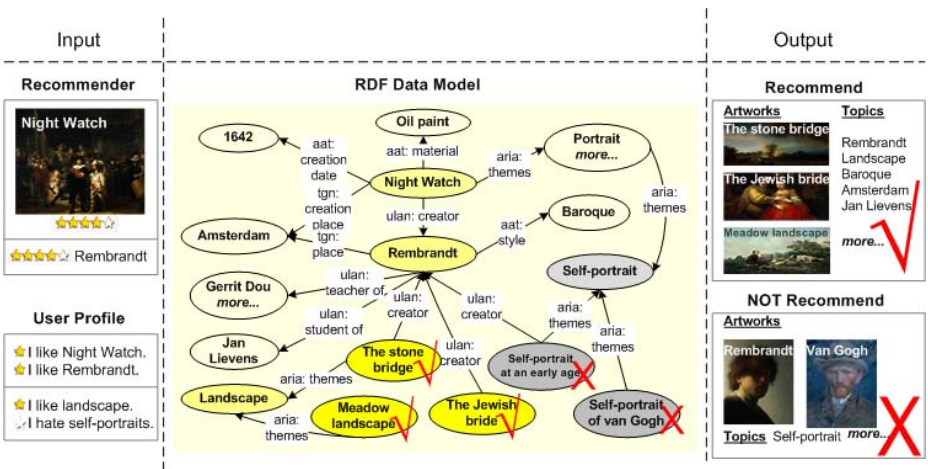


Fig. 1. Exploring semantic links in the Rijksmuseum Amsterdam collection



- Find all artworks with these properties, e.g. *The Jewish Bride* and *The Stone Bridge* by Rembrandt
- Include all artworks with property *Landscape*, e.g. *The Stone Bridge* and *Meadow Landscape*
- Exclude all artworks with property *Self-portrait*, e.g. *Self-portrait of Van Gogh* and *Self-portrait at an early age*

This results in two sets of recommendations:

- Result: recommend all artworks with the above positively rated properties. All recommended artworks are ordered by the number of matching properties, e.g. *The Stone Bridge* is the first one because it has both *Rembrandt* and *landscape*.
- Result: recommend all topics with the above positively rated properties, e.g. *Rembrandt*, *Landscape* and *Baroque*

Two more usage scenarios are give in the online tutorial.

### 3 CHIP Architecture

The demo is based on a Sesame [1] RDF store with SeRQL-based access to user modeling, recommendation and tour generation components. The tour generation component consists of two main parts: (1) a semantic-search facility for the user to search for themes or topics of a possible tour (e.g. a search for Rembrandt will result in a sub-set of Rembrandt artworks which are of interest to this user according to her user profile); and (2) my tours visualization on a historical timeline, museum map or as a list of artworks. In the latter the user can also manually create a tour by giving it a name and then continuing with the search option to find single artworks to include in the currently created tour holder. In fig. 2 we show the current CHIP architecture and its sub-components.

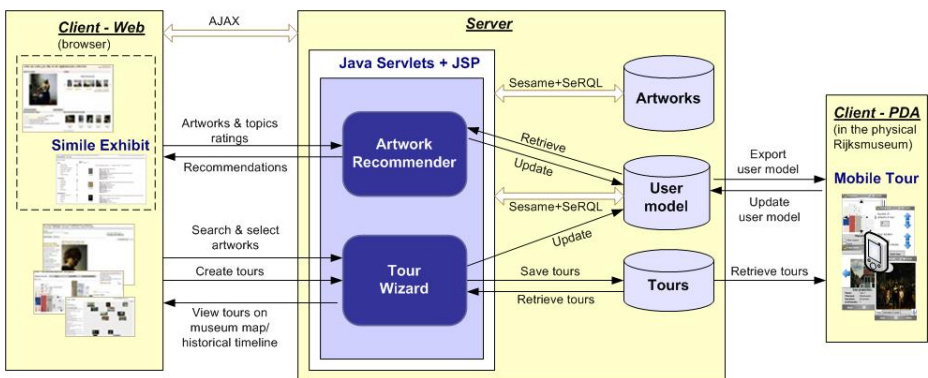


Fig. 2. CHIP Demonstrator Architecture

### 3.1 Rijksmuseum Amsterdam Collection and Shared Vocabularies

Currently, the demonstrator hosts four thesauri, namely the three Getty vocabularies<sup>5</sup>, i.e., the Art and Architecture Thesaurus (AAT), Union List of Artists Names (ULAN) and the Thesaurus of Geographical Names (TGN), as well as the subject classification Iconclass<sup>6</sup>. We use mappings to IconClass provided by the STITCH project<sup>7</sup>. We use the Getty thesauri conversion from their original XML format into an RDF/OWL representation done by the MultimediaN E-culture project<sup>3,4</sup>. The Getty thesauri are licensed<sup>7</sup>. Following this approach we use mappings of the ARIA terminology to the AAT, ULAN, TGN and IconClass concepts. For example, the concepts for places in ARIA refer to location terms in TGN; styles in AAT are linked to artists in ULAN; birth places of artists in ULAN refer to location terms in TGN; subject themes in ARIA refer to subjects in IconClass; names of artists in ARIA refer to ULAN artists, etc. See fig. 3. We use the official ARIA collection of the Rijksmuseum in Amsterdam containing images of some 750 master pieces maintained at the Rijksmuseum Amsterdam website. However, we are now preparing for a migration to the main Oracle database of about 70,000 objects, extending the current RDF/OWL with not only more artworks but also shop, news and user comments items. The current client interface is developed in HTML+CSS and Ajax<sup>5</sup>.

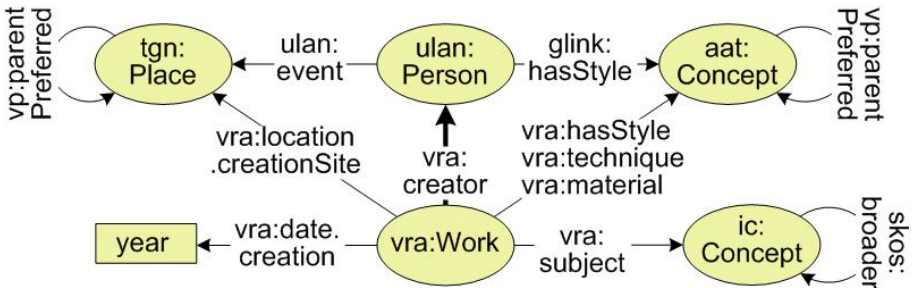


Fig. 3. CHIP Data Model and Vocabularies

## 4 Build Your Profile with Artwork Recommender

The user can start the exploration by first building a user profile in the Artwork Recommender component. This is driven by a rating dialog<sup>6</sup> for artworks from the Rijksmuseum Amnsterdam collection. The user can express her opinion using five stars, where the meaning of each star is shown when you hover the cursor over it. Next to a rating the user can indicate for each artwork whether to be used in

<sup>5</sup> [http://www.getty.edu/research/conducting\\_research/vocabularies](http://www.getty.edu/research/conducting_research/vocabularies)

<sup>6</sup> <http://www.iconclass.nl/libertas/ic?style=index.xsl>

<sup>7</sup> The partners in the project have acquired licenses for the thesauri. People using the demonstrator do not have access to the full thesauri sources, but can use them to annotate and/or search the collections.

further recommendations or not, by using the checkbox *Not interested in*. Rated artworks checked as *not interested in* will not influence the recommendations. The user can continue the process of rating artworks as long as she is satisfied with the state of the user profile shown on the right, or as long as the set of recommended artworks shown in the lower right part of the screen seem relevant. The user can skip rating artworks by pressing the *Next artwork* button. There is no average number of artworks the user needs to rate. In order to kick off the recommendations the user needs to give at least one positive (3, 4 or 5 stars) rating. The system would not be able to recommend artworks and topics based only on negative (1 or 2 stars) ratings.

Important here is that we recommend not only artworks but also topics (based on the semantic description of each artwork you have already rated). The user can provide her positive or negative feedback to each recommendation (both topics and artworks) by rating the empty set of stars associated with it. This would be recorded then in the user profile in order to increase the level of certainty for related properties and artworks.

The screenshot displays the 'CHIP Personalized Access to the Rijkmuseum Amsterdam Collection' interface. At the top, there are navigation links: 'Tutorial', 'Project Page', 'View Full Profile', and 'Log Out (log)'. Below this, the user is logged in as 'Art Recommender' and 'Tour Wizard'. The main content area is divided into several sections:

- Artwork To Rate:** A large image of a woman holding a platter with a man's head. Below it are controls to 'Rate this artwork' (with a star rating system), a checkbox for 'Not interested', and a '+ Add to favor(s)' button. Navigation buttons for 'Previous artwork' and 'Next artwork' are also present.
- Your Profile:** A section titled 'Your Profile' containing three columns of user preferences:
  - Topics I Like:** 'Creation site (1)' (5 stars, Haarlem), 'Material (2)' (5 stars, Baroque; 5 stars, Oil paint), and 'Themes (2)' (5 stars, Food and Drink; 5 stars, Still lifes with objects).
  - Neutral Topics:** 'Artists (1)' (5 stars, Steen, Jan), 'Themes (2)' (5 stars, Silver; 5 stars, Table- and kitchenware).
  - Topics I Dislike:** 'Artists (1)' (1 star, Heda, Willem Claesz.), and 'Themes (2)' (1 star, Group; 1 star, Portraiture).
- Artworks I Rated (6):** A horizontal row of six small artwork thumbnails, each with a star rating and a checkbox. The first two have 'X' marks, indicating they are not interested.
- Topics We Recommend (51):** A list of recommended topics with star ratings and question marks: 'Brush technique', 'Still lifes with food', 'The Glory of the Golden Age', 'Amsterdam', and 'Light source'. A link 'See all recommended topics...' is provided.
- Artworks We Recommend (424):** A horizontal row of five artwork thumbnails, each with a star rating and a question mark. A link 'See all recommended artworks ...' is provided below.

Fig. 4. Screenshot of the CHIP Recommender

If the user is logged with a FOAF profile in the option to view the full user profile (right top) will show the user's personal and social network data. In the full profile we also store the history interaction data about tours that this user has created and/or followed both on the Web and in the museum with the PDA-based mobile tour. Current investigations focus on including also social filtering

[7] in order to include social aspects in recommendation and to optimize specific cold-start problems.

In this interactive way [8] the user has the opportunity to quickly get a feeling of the Rijksmuseum collection and give her opinion on the paintings presented. Based on a user rating the system finds artworks from the Rijksmuseum collection that are the most probable candidates for this user to like. See fig. [4] for a screenshot of the recommender user interface.

## 5 Create and Personalize Your Tours with Tour Wizard

The main rationale behind the CHIP demonstrator is based on the assumption that users more and more will spend time preparing their visits to museums and also reflecting on them after the visit. This is driven by the observation that users have an increasing choice of digitalized collections and related information. This leads us to our main goal to allow the **users to be their own curators**, e.g. selecting the artworks they want to see, influencing the order, the overall theme of the tour and the time to be spent in the museum. In order to realize this we maintain a common user profile for the user on the Web and on the mobile device, and in this way we keep track of users interaction history both in the virtual and in the physical museum.

In the Tour Wizard you have two main screens (tabs):

- **My Tours:** Here the user can see an overview of the current tours and can create a new tour if desired. She can (1) select a tour and then (2) search for a topic, artist or an artwork to include further new items in the currently selected tour. **Note** that if you go straight to the search option you will not

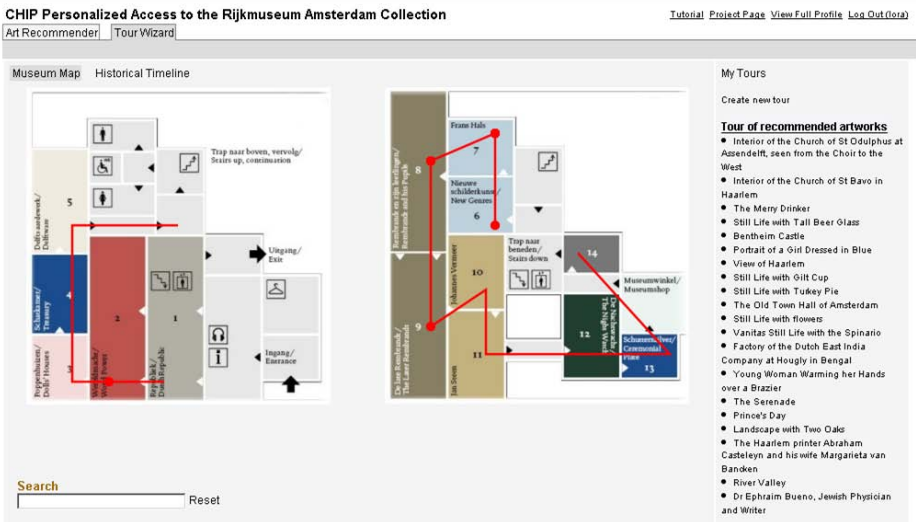


Fig. 5. Screenshot of *My Tours* on the museum map

CHIP Personalized Access to the Rijksmuseum Amsterdam Collection

Art Recommender | Tour Wizard

Tutorial | Project Page | View Full Profile | Log Out (tour)

Museum Map | Historical Timeline

Timeline © SHMLL

1620 1630 1640 1650 1660 1670 1680

1500 1600 1700 1800

Search  Reset

My Tours

Create new tour

**Tour of recommended artworks**

- Interior of the Church of St Odulphus at Assendelft, seen from the Choir to the West
- Interior of the Church of St Bavo in Haarlem
- The Merry Drinker
- Still Life with Tall Beer Glass
- Bentheim Castle
- Portrait of a Girl Dressed in Blue
- View of Haarlem
- Still Life with Gilt Cup
- Still Life with Turkey Pie
- The Old Town Hall of Amsterdam
- Still Life with Flowers
- Vanitas Still Life with the Spinnario
- Factory of the Dutch East India Company at Hooghly in Bengal
- Young Woman Warming her Hands over a Brazier
- The Serenade
- Prince's Day
- Landscape with Two Oaks
- The Haarlem painter Abraham Casteleyn and his wife Margarieta van Banden
- River Valley
- Dr Ephraim Bueno, Jewish Physician and Writer

Fig. 6. Screenshot of *My Tours* on the historical timeline

be able to add new items to your tours; you will be able to only search for topics, artworks or artists and browse through the search results. Among the already existing tours we generate for each user a **Tour of Recommended Artworks**, which contains 20 Rijksmuseum master pieces selected according to user's current user profile. If further interaction with the demo appears, the user profile will be altered and in this way the tour of recommended artworks will be updated as well. Each tour can be viewed by the user as a list and on a **Museum Map** (see fig. 5) or on a **Historical Timeline** (see fig. 6).

- **Search Results:** Here the user can search the Rijksmuseum collection for topics, artists, locations, styles, artworks, etc. and browse the search results. Ideally these results will be altered (filtered with) according to the user profile, so that she will see the artworks related to her search query, which are of relevance and interest to the user. However this option is still in work-in-progress, so it might not work properly yet. **Note** that if you first do search and then (while browsing the results) you want to add some of them to an existing tour, you will not be able to do so. Unfortunately, the current interface implementation restricts the user to **first select a tour and then perform a search for artworks to be included in this tour**. We are working on improving this interface limitation, so that you are able to add at any point of time an artwork from the search result (or even from your Artwork Recommender interface) to an existing tour.

*Acknowledgements.* The CHIP (Cultural Heritage Information Personalization) project is funded by the Dutch Science Foundation funded program CATCH (Continuous Access to Cultural Heritage) in the Netherlands.

## References

1. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A generic architecture for storing and querying rdf and rdf schema (2002)
2. van Gendt, M., Isaac, A., van der Meij, L., Schlobach, S.: Semantic web techniques for multiple views on heterogeneous collections: A case study. In: Gonzalo, J., Thanos, C., Verdejo, M.F., Carrasco, R.C. (eds.) ECDL 2006. LNCS, vol. 4172, Springer, Heidelberg (2006)
3. van Assem, M., Menken, M., Schreiber, G., Wielemaker, J., Wielinga, B.: A method for converting thesauri to rdf/owl. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 17–31. Springer, Heidelberg (2004)
4. Schreiber, G., Amin, A., van Assem, M., de Boer, V., Hardman, L., Hildebrand, M., Hollink, L., Huang, Z., van Kersen, J., de Niet, M., Omelayenko, B., van Ossenbruggen, J., Siebes, R., Taekema, J., Wielemaker, J., Wielinga, B.: Multimedial e-culture demonstrator. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 951–958. Springer, Heidelberg (2006)
5. Garrett, J.J.: Ajax: A new approach to web applications (February 2005), <http://www.adaptivepath.com/publications/essays/archives/000385.php>
6. Wang, Y., Aroyo, L., Stash, N., Rutledge, L.: Interactive user modeling for personalized access to museum collections: The rijksmuseum case study. In: UM 2007. Proceedings of the International User Modeling Conference, Corfu, Greece, Springer, Heidelberg (2007)
7. van Setten, M.: Supporting people in finding information: Hybrid recommender systems and goal-based structuring. Telematica Instituut Fundamental Research Series, No.016 (TI/FRS/016). Universal Press (2005)
8. Rutledge, L., Aroyo, L., Stash, N.: Determining user interests about museum collections. In: WWW 2006. Proceedings of the International WWW'06 Conference (poster), Springer, Heidelberg (2006)

# Semantics-Based Framework for Personalized Access to TV Content: The iFanzo Use Case

Pieter Bellekens<sup>1,4</sup>, Lora Aroyo<sup>1,2</sup>,  
Geert-Jan Houben<sup>1,3</sup>, Annelies Kaptein<sup>4</sup>, and Kees van der Sluijs<sup>1</sup>

<sup>1</sup> Eindhoven University of Technology, Computer Science  
`{p.a.e.bellekens,k.a.m.sluijs}@tue.nl`

<sup>2</sup> VU University Amsterdam, Computer Science  
`l.m.aroyo@cs.vu.nl`

<sup>3</sup> Vrije Universiteit Brussels, Computer Science  
`Geert-Jan.Houben@vub.ac.be`

<sup>4</sup> Stoneroos Digital Television, Hilversum  
`annelies.kaptein@stoneroos.nl`

## 1 Introduction

The ICT landscape is developing into a highly-interactive distributed environment in which people interact with multiple devices (e.g. portable devices such as mobile phones and home equipment such as TV's) and multiple applications (e.g. computer programs such as Web browsers and dedicated Web services) [1]. Globally, the industry is being driven by the shift away from old models - from physical space to digital space. New methods emerge for getting content such as TV programs via the Web. Almost half of the people want to watch TV content on their PC's; they want to make a bridge between a TV and a PC, perhaps even sitting in a home office [2]. The information overload is enormous and the content presented is hardly adapted to the prior knowledge, to the preferences and to the current situation of the user.

*Personalization* in information retrieval and information presentation has therefore become a key issue and a key ingredient of the so-called "Web 2.0" applications. However, such personalization is still local and cannot be used in the context of other information services: e.g. personalized information is only valid within one Web application such as an online TV Guide [3]. Moreover, the online TV Guide cannot cater for different "modes" of a user, e.g. when watching a program himself, or when watching together with friends [3]. Thus, main aspects we cover in this work are **Data integration** of distributed collections, a **Context modeling** framework for temporal and spatial-specific viewpoints, **User modeling** in a contextualized form, and **Personalized presentation** of the combined information about data, context and user.

In this paper we present SenSee, a semantics-based framework for providing personalized access to TV content in a cross-media environment. It allows for an integrated view on data harvested from heterogeneous and distributed Web sources. The ultimate goal is to support individual and group TV viewers

---

<sup>1</sup> <http://www.tvguids.nl>



(operating multiple devices, e.g. PC, set-top box and mobile) in finding programs that best suit their interests. Personalization here has to consider both data-integration issues (how is information from different applications and devices related?) as well as context-modeling issues (in which space/time/mode are statements about a user valid?). iFanzly is a personalized TV guide application using the SenSee framework. iFanzly consists of a Web-based front-end serving as a Web-based remote control point to the set-top front-end. Future extension is considered on a mobile platform.

SenSee integrates multiple data sources, such as BBC data from BBC Backstage<sup>[2]</sup>, XMLTV<sup>[3]</sup> and IMDB<sup>[4]</sup>. Moreover, these sources are interconnected and mapped to external vocabularies, like OWL Time<sup>[4]</sup>, Geo Ontology, TV Anytime genre classification<sup>[5]</sup> and WordNet<sup>[6,7]</sup>. The resulting abundance of data is controlled in the user interface by offering the user a faceted browsing view on the data, i.e. they can search and browse the data based on facets for time, location and genre. We daily retrieve the metadata from the different multimedia sources on the Web. How we integrated this data is described later, in section 3.

Large part of this framework, primarily supporting the set-top box front-end, has been developed within the context of the Passepartout project<sup>[1]</sup> in collaboration with Philips Applied Technologies and Stoneroos Interactive Television<sup>[8]</sup>. Currently Stoneroos, VU University Amsterdam and Eindhoven University of Technology are working towards the commercial deployment of the framework in the form of the personalized electronic program guide iFanzly<sup>[9]</sup>.

This is work in progress so expect to find changes over time, both on the platform side as on the interface side. However, the basic functionality is ready and available at <http://wwwis.win.tue.nl:8888/SenSee>, as well as a brief tutorial on how to use the system. The Web client is built on top of GWT<sup>[10]</sup>, which allows it to run on most modern JavaScript enabled Web-browsers. However, so far we have only tested it in Firefox 2 and Internet Explorer 7. Note that between 2:00h and 4:00h (GMT +1:00) the servers retrieve, parse and transform broadcast information, so expect the application to be slower in that period of time.

## 2 Architecture

Figure 1 gives an overview of the architecture of the SenSee framework (i.e. excluding the front-end interfaces) complying with the main requirements to be scalable, extensible and flexible. For more details on the architecture see 3.

- 
- <sup>2</sup> <http://backstage.bbc.co.uk/>
  - <sup>3</sup> <http://xmltv.org/wiki/>
  - <sup>4</sup> <http://imdb.com>
  - <sup>5</sup> <http://www.tv-anytime.org/>
  - <sup>6</sup> <http://wordnet.princeton.edu/>
  - <sup>7</sup> <http://www.w3.org/2001/sw/BestPractices/WNET/wn-conversion>
  - <sup>8</sup> <http://www.stoneroos.nl/>
  - <sup>9</sup> <http://www.stoneroos.nl/portfolio/case-study/Passepartout-personalised-EPG>
  - <sup>10</sup> <http://code.google.com/webtoolkit/>



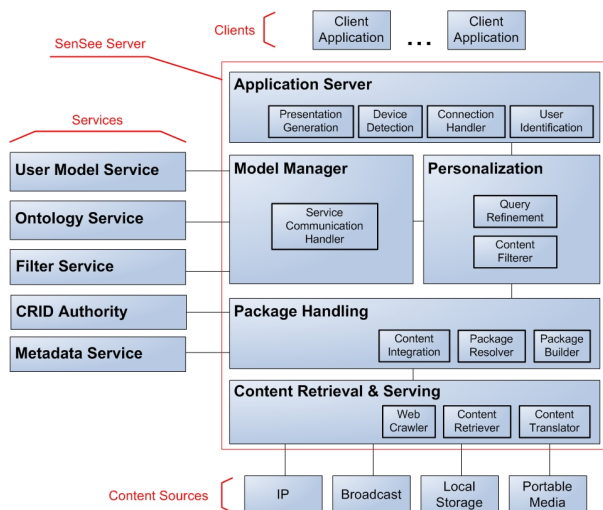


Fig. 1. SenSee Architecture

A key element in this framework is the use of Semantic Web languages RDF(S) /OWL to model all input data and its relationships, so that we can reason and navigate over it in a uniformly connected way. In this case it means navigating via shared facets like time, location and genre.

For the scalability we rely on the RDF storage, inferencing and querying framework Sesame. All data handled within the SenSee platform is stored and retrieved by Sesame, including all metadata sources as well as the ontological sources, the user model, and the context models.

Various client applications can use the SenSee platform. These are not only front-end interfaces (e.g. the iFanzy settop box and Web-based applications)<sup>11</sup>, but could also be sensed-devices like the sensor-enhanced pillow<sup>5</sup> we have also developed in collaboration with V2 and CWI.

### 3 Data Integration

Program information from BBC broadcasts are retrieved directly from BBC Backstage in TV-Anytime XML format. We have transformed this into OWL/RDF and used SKOS to describe relations between concepts in the loosely structured vocabulary (i.e. by using the relations skos:broader, skos:narrower and skos:related). Furthermore we mapped the TV-Anytime time definition to the OWL-TIME notion of time. So, for every crawl of the data source we transform the input XML data to RDF-instances of our schema and we transform the data by using XPath. Another broadcast source is XMLTV program information grabbed from several program information websites. XMLTV covers various

<sup>11</sup> <http://en.wikipedia.org/wiki/iFanzy>

countries worldwide<sup>12</sup>. For demonstration purposes we have chosen to focus on the broadcast information from UK and the Netherlands. This choice does not imply any restrictions in the generality of the framework. Both data sources are crawled on a daily basis.

Movie-related information we gather from the IMDB dataset (text dump of the IMDB database; data model translated by us in RDF(S)). This text dump however does not contain photos and movie trailers URLs. Thus, we made scripts that on access of movie details retrieve such information live from the Web with screen scrapers and then cache that data in the database. The IMDB information is used for detailed descriptions of movies in the regular broadcasted TV programming as well as in the option "movie on demand", e.g. on a pay-per-view basis. As there is currently no commercial pay-per-view party involved in the project we show-case only that it is possible to integrate such a movie source with no time-dimension, e.g. broadcast time. For the pay-per-view option we use the 10.000 most popular movies in the IMDB dataset (determined by the number of IMDB votes). The information coming from BBC Backstage and the XML-TV sources can be seen as a sliding window of 8 effective days, where the first day is the current day. With other words, if it is now Wednesday, we have metadata from today until and including next Wednesday.

An overview of all the datasets we use within the demo:

**Table 1.** Size of data sources

Data source	#triples
User Model (schema)	319
IMDB schema	408
TV Anytime Genre Classification	3.971
Geo Ontology	59.835
Time Ontology	1.547
Country Codes	2.023
WordNet	1.942.887
BBC dataset (random pick)	91.447
XMLTV dataset (random pick)	1.278.718
IMDB dataset	7.969.199

## 4 Faceted Browsing

Bringing together TV content available from a high number of broadcasting channels and a number of shared vocabularies for its semantics-rich representation, requires a special attention of the search and browse interface presented to the user. Thus, we provide two ways to access and navigate within TV content, i.e. by basic search and by facet browsing.

Figure 2 shows a screenshot of the Web interface search panel. On the left one can browse selected *facets*, e.g. time, genre and location. *Time* is visualized as

<sup>12</sup> see <http://xmltv.org/wiki/xmltvworlddomination.html> for an overview

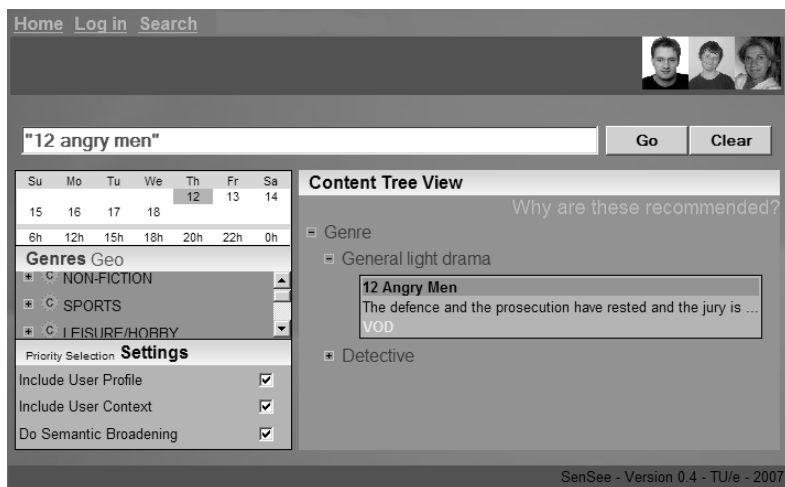


Fig. 2. SenSee Screenshot

a calendar showing the current day, and the following six days. Users can select a day and further specify the time period. Broadcast information is time-bound as opposed to on-demand movies, so a time-selection will only filter broadcast results. *Genre* is visualized as a TV-Anytime classification tree. Users can select one or more genres for the program they are searching for. *Location* information is visualized as a IMDB-location tree. In the IMDB dataset every movie has filming locations. Such location is composed as a hierarchical string. For instance, for the movie *The Godfather* one of the filming locations is **Bronx, New York City, New York, USA**. We parsed these strings building a hierarchy, and created a structure modelled similar to the RFDS subclass structure, i.e. **Bronx -subComponentOf-> New York City -subComponentOf-> USA**. When we encountered equally named location names with different parents, we created different unique nodes for them. For instance there is an **Athens** in both the **USA** and in **Greece**, but the locations are different. It appeared that locations were uniquely named in the database (i.e. there were no derivations like **USA** in one place and **United States of America** in another). Users can select one or more locations in the tree, similar as we do for genres, which limits movies that were shot in the selected locations. Locations are in a separate store with inferencing, so we compute the closure. If a user selects a location that has sub-components, those will also be returned as a result.

The *basic search* field accepts a Google-like string input. By default the input strings are matched to title, keywords and actor names in all sources available. The *advanced search* triggers the following actions before the query execution:

- **Keywords broadening:** configured to query for synonyms in WordNet, i.e. terms appearing in the same set as the input term. We currently limited the keyword expansion to 5 new terms per keyword. The output is a set of keywords.

- **Keyword conceptualization:** all facet-sources (e.g. time, genre and location) are parsed for match for every keyword in the broader set. This matching has three modes, i.e. strict, loose and free. In the *strict match* the keyword should be syntactically equal to the concept (ignoring case), e.g. the keyword `News` and the genre concept `News` match. *Loose match* looks for sub-strings, e.g. the keyword `News` and the genre concept `Daily News` match, where it would not match in strict mode. *Free match* does pattern matching and would find the term `day` as a match with `Friday`, where it would not match in loose match. The output is a set of concepts.
- **Concepts broadening:** for a given object property we traverse the corresponding graphs and find related concepts (including inverse relationships). For example, the current TIME ontology configuration is to consider the inverse `time:after` relationship, i.e. we find all concepts that are connected to a given time concept via the `time:after` relationship. Given, for instance, the `time:Noon` concept, we will also find the `time:afterNoon` concept. For the TV-Anytime genre classification we consider the `skos:broader` relationship, resulting in all concepts that are *narrower* than the input concept. For example, for the `tva:3.2 (Sports)` concept we will also find the `tva:3.1.1.9 (Sports News)`, as all programs of the genre `Sports News` are also about `Sports`.

These actions lead to execution of a number of queries in a time that depends on *number of keywords*, the *number of ontologies*, the *size of ontologies* and the *configured number of extension steps*. The more generic a search keyword is, the more extensive the set of additional terms and concepts is. In the current implementation, these broadening and conceptualization actions take on average about 4 or 5 seconds to execute on a reasonable modern single server machine. Currently, we perform various optimizations of the demonstrator, for example by using keyword indexes that can considerably speed-up the process.

Results can be grouped in several facets. The user can determine which facets to be used and in what order the results will be grouped (by dragging and dropping the facet names in the desired order). The grouping is arranged in a tree, where the top priority facet will make the first level of the tree. If `Genre` is of highest importance to the user, the first level tree will contain the genres of the first N results in alphabetic order. Grouping occurs in a similar fashion for the extra facets. Note that programs can have several genres, and that results will therefore occur more than once in the tree.

## 5 Personalization and Adaptation

In our previous work with a sensor-enhanced pillow [5] in combination with the set-top box front-end, we have collected bio-sensor data in order to calibrate the user profile. In the current Web application use case we gather user data by monitoring the user behavior, or as we have shown in previous research by re-using data from other application [6]. The current implementation does not

include a user interface to create user profiles yet. For the purposes of this demo we created three different user and context profiles. The current time is determined dynamically, however other context information, such as current location, is for demo purposes pre-set in the context user profiles. One user preference is language. Currently we support English, Dutch and Swedish. If several users log at the same time their profiles are combined and the common language is chosen. The default language is English, however if all logged users prefer another language that language will be chosen (i.e. in general we compute an intersection of the profiles).

The key personalization feature is adapting the search results with respect to the user profile, i.e. content with user preferred terms and genres is ranked higher. Negative preferences are also taken into account. Context information is applied in a similar fashion. It can be configured in the concrete application and adapted to the context needed there. In the current demo we assume that people would prefer to look at certain types of TV programs that relate to the location they are in. For example, traffic information, news and police reports for a location in London if the user is situated there.

## 6 Lessons Learned and Future Work

Several problems and issues have been tackled during the work on this project. In this section we present a brief reflection on them and draw lessons learned.

**Sesame performance and scalability:** In close collaboration with Aduna<sup>13</sup> we used Sesame as the backbone RDF storage and querying. The new alpha (and later beta) allowed us to use the newest Sesame features related to context modeling. However, this came in a package with a *poor query optimization* (for now), especially critical when using large datasets like IMDB. The overall result is a slower and less scalable demonstrator. By cross testing queries in Sesame 1 and 2, and regarding that the query and data model should lead to better optimization in the second version lead us to believe that some queries will improve in evaluation time up to a factor 10. Similarly, by using Googles GWT toolkit for programming the front-end we depend on its portability and efficiency (which is for example currently rather slow and inflexible with tree-rendering).

**Code optimization:** Currently we undergo various optimizations of the system, e.g. free text indexing for matching keywords. The application now runs on a single (single-core) server because of resource limitations, while the application lends itself quite well for parallelization, especially if we move specific functionality to different machines. So if the application feels a bit sluggish now (especially with multiple users), expect it to greatly speed up in the following months.

**Use of live data:** Initially we aimed at working with all live data. For instance, we did not want to store the IMDB-text dump (as it is not refreshed that often),

<sup>13</sup> <http://www.aduna-software.com>

but query it live all of the time and scrape its HTML-pages. However, after experiencing many layout changes, and thus rewrites of the screen scrapers, we had to give that up because of the amount of work it gave us. We are using live metadata from the BBC Backstage website.

**User interface:** The current faceted-based presentation of the search result is a good demonstration of combining multiple perspectives in one view. However, we are exploring options to use timeline<sup>14</sup> for the temporal aspect and a map<sup>15</sup> for the location facet. Furthermore, we aim at realizing combinations of different facets, e.g. conventional paper TV guides typically combine the channel facet with the time facet.

**Recommendations:** Recommendation functionality (currently implemented for an empty query) needs features currently not available in our backbone database's query engine, such as sorting, aggregation, query nesting and update queries. Further developments in the context of recommendations will involve social aspects and focus more on group recommendations, content sharing, etc.

## References

1. Bjorkman, M., Aroyo, L., Bellekens, P., Dekker, T., Loef, E., Pulles, R.: Personalised home media centre using semantically enriched tv-anytime content. In: EuroITV 2006 Conference, pp. 156–165 (2006)
2. Aroyo, L., Bellekens, P., Bjorkman, M., Houben, G.J.: Semantic-based framework for personalised ambient media. In: Multimedia Tools and Applications (in print, 2007)
3. Aroyo, L., Bellekens, P., Bjorkman, M., Houben, G.J., Akkermans, P., Kaptein, A.: Sensee framework for personalized access to tv content. In: Interactive TV: a Shared Experience, Amsterdam, the Netherlands, pp. 156–165. Springer, Heidelberg (2007)
4. Hobbs, J.R., Pan, F.: An ontology of time for the semantic web. *ACM Transactions on Asian Language Information Processing (TALIP)* 3(1), 66–85 (2004)
5. Aroyo, L., Nack, F., Schiphorst, T., Schut, H., KauwATjoe, M.: Personalized ambient media experience: move.me case study. In: *IUI 2007. Proceedings of the 12th international conference on Intelligent user interfaces*, pp. 298–301. ACM Press, New York (2007)
6. van der Shuijs, K., Houben, G.J.: A generic component for exchanging user models between web-based systems. *IJCELL Journal* 16(1/2), 64–76 (2006)

<sup>14</sup> <http://simile.mit.edu/timeline/>

<sup>15</sup> <http://www.google.com/apis/maps/>

# Revyu.com: A Reviewing and Rating Site for the Web of Data

Tom Heath and Enrico Motta

Knowledge Media Institute, The Open University,  
Walton Hall, Milton Keynes, MK7 6AA, United Kingdom  
{t.heath,e.motta}@open.ac.uk

**Abstract.** Revyu.com is a live, publicly accessible reviewing and rating Web site, designed to be usable by humans whilst transparently generating machine-readable RDF metadata for the Semantic Web, based on their input. The site uses Semantic Web specifications such as RDF and SPARQL, and the latest Linked Data best practices to create a major node in a potentially Web-wide ecosystem of reviews and related data. Throughout the implementation of Revyu design decisions have been made that aim to minimize the burden on users, by maximizing the reuse of external data sources, and allowing less structured human input (in the form of Web2.0-style tagging) from which stronger semantics can later be derived. Links to external sources such as DBpedia are exploited to create human-oriented mashups at the HTML level, whilst links are also made in RDF to ensure Revyu plays a first class role in the blossoming Web of Data. The site is available at <<http://revyu.com>>.

## 1 Introduction

Revyu.com is a live, publicly usable (and used!) reviewing and rating Web site developed using Semantic Web technologies and standards, and according to Linked Data principles [1] and best practices [2]. Reviews and ratings are widely available on the Web and are one major form of Web2.0-inspired 'user-generated content'. However, despite the availability of reviews through APIs such as Amazon Web Services, this data remains largely in isolated 'silos', and described in formats that hinder its integration and interlinking with data from other sources. This presents considerable barriers to the aggregation of all reviews of a particular item from across the Web. As has been recognised by previous authors [3, 4], the Semantic Web, or Web of Data, provides a technological platform with which to overcome this problem. Revyu takes a significant and concrete step towards this, by exposing reviews using standards such as RDF and SPARQL. In doing so it helps to seed an ecosystem of interlinked reviews, and to bootstrap the Semantic Web as a whole.

## 2 Revyu Overview

Revyu allows people to review and rate things simply by filling in a Web form. This style of interaction with the site will be familiar to those who have written reviews on

sites such as Epinions<sup>1</sup> or Amazon<sup>2</sup>. Whilst this functionality is not especially novel, as a reviewing application Revyu improves significantly over other work in the area in the following ways: it goes well beyond the closed world 'silos' of sites such as Epinions and TripAdvisor, by exposing reviews in a reusable, machine-readable format; it improves upon the APIs of sites such as Amazon by using a more flexible data format (RDF), allowing more versatile queries via SPARQL, and linking to external data sources; lastly the site takes an open world view of the reviewing process by not constraining users to reviewing items from a fixed database. Anything a user can name can be reviewed, whilst links supplied with the review can disambiguate items thanks to inverse functional properties such as *foaf:homepage*. Consequently reviewers are not restricted to reviews and ratings in one domain, as is the case with Golbeck's FilmTrust [4]. As of August 2007 Revyu has been live for 10 months, attracting 412 reviews from 112 reviewers.

Revyu is built from the ground upwards on Semantic Web technologies. By following Linked Data principles [1] and best practices [2] the site ensures that reviews it hosts can be fully connected into a Web of Data. This approach manifests itself in a number of ways. All site content, in addition to being available in HTML, is also published in RDF/XML that is interlinked with the corresponding HTML pages but available as separate crawlable documents. As we have described elsewhere, this creation and publication of RDF is invisible to the reviewer, enabling novice users to contribute data to the Semantic Web through a familiar, Web2.0-style mode of interaction [5]. To date this approach has yielded over 13,000 RDF triples publicly available on the Semantic Web. Whilst not a large figure by many standards, it is significant that these triples have been generated primarily from direct user input, rather than by data mining, extraction from natural language, or conversion of existing databases.

In addition to review data, RDF describing reviewers, reviewed items, and tags they assign to these is published on the site. These descriptions use the FOAF [6] and Tag [7] ontologies, as well as properties and classes from RDFS and OWL. This data can also be retrieved programmatically via the Revyu SPARQL endpoint<sup>3</sup>, allowing third parties to access Revyu data for reuse in their own applications. Whilst in some ways analogous to Web2.0 APIs that provide remote query capabilities, SPARQL endpoints afford many advantages to the developer: for example, common libraries can be used to query multiple RDF graphs yet return the results as one resultset, effectively allowing joins over multiple data sources. In the following section we will detail the technical infrastructure underlying Revyu, and discuss decisions made in implementing the system.

### 3 Revyu Architecture and Implementation

Revyu is implemented in PHP, and runs on a regular Apache web server. The RDF API for PHP (RAP) [8] provides RDF processing capabilities, whilst RDF data is persisted to a de-normalised MySQL database following the RAP database schema.

---

<sup>1</sup> <http://www.epinions.com/>

<sup>2</sup> <http://www.amazon.com/>

<sup>3</sup> <http://revyu.com/sparql/welcome>



The Revyu SPARQL endpoint relies on the RAP SPARQL engine, which operates against the same MySQL-based triplestore.

From the outset Revyu was designed to adhere to the four 'commandments' of Linked Data outlined by Berners-Lee [1]: using URIs as names for things, using HTTP URIs so people can look up those names, providing useful information when someone looks up a URI, and linking to other URIs so more things can be discovered.

All things represented on Revyu are assigned URIs: reviews, people, reviewed things, tags assigned to things, and even the bundles that represent tags assigned by one person at one point in time. Providing URIs for all these things gives many items a presence on the Semantic Web which they would not have otherwise, and enables any third party to refer to these items in other RDF statements. This opens the way for links between Revyu and other data sets, thereby helping to lay the foundations for a Web of Data.

All URIs in the Revyu URI-space can be dereferenced. Attempts to dereference the URIs of non-information resources receive an HTTP303 "See Other" response containing the URI of a document that describes the resource. This adheres to the W3C Technical Architecture Group's finding on the `httpRange-14` issue [9], and serves to reinforce the distinction between a resource and a description of that resource. Content negotiation is also performed on Revyu URIs, whereby the user agent receives a description of the resource in either HTML or RDF depending on the value of the *Accept* header sent in the initial HTTP request.

## 4 Deriving Semantics from Tagging Data

When creating Revyu, a significant decision was taken to not require users to classify the items they were reviewing, but instead to associate keyword tags with the item. This decision was taken for several reasons: firstly there was seen to be a lack of sufficiently comprehensive classifications of items that users may want to review; secondly, requiring all users to subscribe to a single classification scheme for reviewed items seemed unnecessarily constraining and against the spirit of the Semantic Web; thirdly, providing a usable interface through which non-specialists could classify items using arbitrary types discovered in ontologies on the Semantic Web was seen as unfeasible; and lastly, the coverage provided by ontologies readily available on the Web was deemed insufficient to describe all items that might be reviewed, therefore potentially resulting in a more closed world of reviewed items.

The recent availability of Yago [10] class definitions via DBpedia [11] has gone some way to addressing these issues, and we will be investigating use of these classes in future work. However we believe that tagging retains the appropriate balance of usability whilst also providing sufficient data from which stronger semantics can be derived. At present we use tagging data in two ways: to identify basic semantic relationships between tags and to derive type information about a reviewed item.

Tags that are frequently associated with the same item are assumed to be related in some way. In the HTML pages about each tag, tags that co-occur above a certain threshold are displayed to the user. This threshold is set low for HTML output, as human readers of the page are unlikely to infer erroneous information based on these relationships. In contrast however, relationships exposed in RDF descriptions of tags

(using the *skos:related* property) are based on a more conservative threshold, in order to avoid erroneous inferences based on these assertions. In ongoing work we are investigating the derivation of more precise relationships (such as superclass/subclass) between tags, based on tagging data.

We currently derive type information from tagging data in two domains, books and films, relying on external data sources to help ensure accurate results. Firstly, where items are tagged 'book' we parse Web links provided by the reviewer that relate to the item, and attempt to extract ISBN numbers embedded in these links. Where we are able to extract an ISBN number in this fashion we conclude that the reviewed item is in fact a book, and assert a corresponding *rdf:type* statement into the triplestore.

If an item has been tagged 'film' or 'movie', we execute a query against the DBpedia SPARQL endpoint<sup>4</sup> in order to find any entries of type *yago:Film* that have the same name as the reviewed item. If a match is found then we conclude this item is in fact a film, and add an *rdf:type* statement to this effect to the triplestore. These type statements for both books and films are exposed in the RDF descriptions of items on Revyu, and also used as the basis for showing additional relevant data in the HTML pages about an item, as detailed in the following section.

## 5 Production and Consumption of Linked Data

Validating Revyu data against external sources not only allows the derivation of more reliable type information than would be possible using tags alone, it also allows items on Revyu to be linked with others from heterogeneous external data sources such as DBpedia<sup>5</sup>, Open Guides<sup>6</sup>, and FOAF data. Where matches are found, we use the *owl:sameAs* property to assert that two URIs identify the same resource. Publishing these links in RDF helps create a Web of Data rather than simply isolated islands of RDF; Revyu data is *in the Web*, not just *on the Web*.

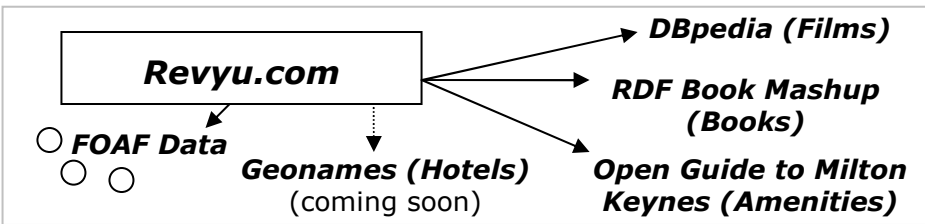


Fig. 1. Links from Revyu.com to external data sets

We actively exploit the links we set between Revyu and external data sources, to enhance the experience of our users without placing an additional burden on reviewers by requiring them to supply additional information about the reviewed item. For example, where *owl:sameAs* statements exist linking films on Revyu to their entry in DBpedia, we retrieve additional information about the film, such as the URI

<sup>4</sup> <http://dbpedia.org/sparql>

<sup>5</sup> <http://dbpedia.org/>

<sup>6</sup> <http://openguides.org/>

of the films promotional poster, and the name of the director. This information is displayed on the Revyu HTML page about the film (as shown in Fig. 2), thereby enhancing the value of the site for users without requiring this information to be manually entered into Revyu. Similarly we use *owl:sameAs* links between Revyu and the RDF Book Mashup [12] as the basis for retrieving book cover and author information which is also then displayed on the Revyu HTML page about the book (see <sup>7</sup> for an example).

In the RDF descriptions of items we take a slightly different approach to that taken with HTML output, choosing to simply expose the links between items without republishing RDF data from external sources. This approach could be described as using Semantic Web data to produce Web2.0-style mashups at the human-readable, HTML level, whilst also mashing up (i.e. linking) data at the RDF level. Not only does this Linked Data approach to mashups reduce issues with licensing of data for republication, it is also a more Web-like approach; duplicating data is of much lesser value than linking to it, and the user agent of the future should be able to 'look ahead' to linked items and merge data accordingly.

It should be noted that we do not claim that the Revyu Web2.0-style mashups represent something that could not have been achieved using conventional Web2.0 approaches. However, the following features distinguish our approach: the simultaneous publishing of data-oriented and human-oriented mashups, so that the data integration effort we have invested is not lost but can be reused by other parties; the ability to easily integrate additional heterogeneous sources using RDF; and the substantially reduced development costs in producing human-oriented mashups through use of Semantic Web technologies.

Whilst to date we have waited for new film reviews on Revyu and then attempted to automatically match them with entries in DBpedia, we are currently preparing for import into Revyu 'skeleton' records covering 12,000 films described in DBpedia. These records simply include the title of the film, a statement indicating that this item is of type '*Film*', a number of keyword tags, and links to the corresponding item on DBpedia. Not only will this provide a foundation on which new reviews can be created, it will also ensure that all films being reviewed in the future will already be interlinked with the corresponding DBpedia entry, and thus the Web of Data.

This skeleton record approach has already been followed when linking Revyu to data from the Open Guide to Milton Keynes<sup>8</sup>, a member of the Open Guides family of wiki-based city guides that expose data in RDF. Milton Keynes is a city in south east England, and home of The Open University. Whilst some amenities in the city, such as pubs and restaurants, were already reviewed on Revyu, many more were listed in the Open Guide due to its longer history. Therefore, after identifying items existing in both locations and making the appropriate mappings to avoid duplication, we created skeleton records in Revyu for the remaining items, setting links back to their Open Guide URIs. This has enabled latitude and longitude data for many items to be retrieved from RDF exposed by the Open Guide, and used to show a Google Map of the items location (see <sup>9</sup> for an example). The same approach can also be used to expose address, telephone, and opening time information held in the Open Guide.


<sup>7</sup> <http://revyu.com/things/the-unwritten-rules-of-phd-research/about/html>

<sup>8</sup> <http://miltonkeynes.openguides.org/>

<sup>9</sup> <http://revyu.com/things/ye-olde-swan-woughton-on-the-green-milton-keynes/about/html>

**Broken Flowers**

---




---


directed by [Jim Jarmusch](#)

---

Broken Flowers is a 2005 comedy-drama film directed and written by

**About tom (Tom Heath)**

---




---

[tom's Home Page](#)

---

tom's location:  
[Borough of Milton Keynes](#)

---

tom's Interests

---

[Semantic Web](#)  
[Word of Mouth](#)  
[Trust](#)  
[Beer](#)

**Fig. 2.** Excerpts from the Revyu HTML page about the film Broken Flowers, showing the film poster, director information, and summary drawn from DBpedia<sup>10</sup>

**Fig. 3.** Excerpts from the first author's Revyu profile page, showing data sourced automatically from his external FOAF file<sup>11</sup>

Similar principles are also applied to user information, such that people registering with the site are not required to provide copious information to populate their user profile. Instead, where they have an existing FOAF description in an external location they may provide its URI, in which case Revyu dereferences this URI and queries the resulting graph for relevant information (such as a photo, location, home page address, and interests), which is then displayed on their profile page, as illustrated in Fig. 3. This approach reduces the burden on the user by not requiring them to manage multiple redundant sets of personal information stored in different locations.

<sup>10</sup> <http://revyu.com/things/broken-flowers-film-movie-bill-murray-jim-jarmusch-sharon/about/html>

<sup>11</sup> <http://revyu.com/people/tom/about/html>

Furthermore, where the user has assigned themselves a URI in their FOAF description, Revyu sets *owl:sameAs* links asserting that this URI identifies the same resource as the user's Revyu URI. Users can also state that they know other Revyu reviewers, at which point this relationship is recorded in the triplestore using the *foaf:knows* property, and exposed (privacy settings permitting) in the user's RDF description on the Revyu site. This ensures that social networking data created in one location is not automatically rendered inaccessible to other services.

## 6 Future Work and Conclusions

In addition to encouraging further user participation in order to increase the value delivered by the site, we plan to integrate Revyu with a number of additional data sets. Most notably we are preparing to create skeleton records in Revyu of 70,000 hotels worldwide, linked to their corresponding entry in the *Geonames* dataset. The same approach will also be used to link Revyu with data from other Open Guides, such as London and Boston. Additional data will be integrated as further relevant sources become available.

It should be noted that our aim in linking to external datasets is not to constrain, but merely to seed, users conceptions of what can be reviewed. As we integrate further data sets we hope to achieve a more automated linking process by investigating generic similarity matching techniques for operation on the wider Semantic Web.

Whilst frequently suggested as an additional feature, at present there are no concrete plans to import external review data into Revyu, for a number of reasons. Firstly, to the best of our knowledge Revyu is the only site serving reviews as Linked Data according to current best practices, which limits our abilities to interlink Revyu with external review data sets; secondly, little review data is available under a suitable license; lastly, our ongoing research is predicated on the ability to combine review data with social networks, requiring some global identifier (such as *foaf:mbox\_sha1sum*) to be available for each reviewer. This is rarely the case with traditional reviewing sites. By providing reviews in a reusable format that is easily integrated and interlinked with other data, Revyu provides core data for our ongoing work into information seeking, recommendation, and trust in social networks on the Web.

In conclusion, in this paper we have described Revyu, a human usable reviewing and rating Web site built on Semantic Web technologies, and fundamentally designed to contribute to the realization of a Web of Data. Whilst superficially not unique in functionality, the site is rare in its status as a publicly available service in daily use that is oriented towards human users, yet also embodies current best practices in developing for the Semantic Web.

## Acknowledgements

This research was partially supported by the Advanced Knowledge Technologies (AKT) and OpenKnowledge (OK) projects. AKT is an Interdisciplinary Research

Collaboration (IRC) sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01. OK is sponsored by the European Commission as part of the Information Society Technologies (IST) programme under grant number IST-2001-34038. Peter Coetzee did a superb job of turning data into skeleton records for import into Revyu. Lastly, the Open Guides and DBpedia communities, and the RDF Book Mashup team deserve our special thanks.

## References

1. Berners-Lee, T.: Linked Data (2006), <http://www.w3.org/DesignIssues/LinkedData.html>
2. Bizer, C., Cyganiak, R., Heath, T.: How to Publish Linked Data on the Web (2007), <http://sites.wiwi.fu-berlin.de/suhl/bizer/pub/LinkedDataTutorial/>
3. Guha, R.: Open Rating Systems. In: Proc. 1st Workshop on Friend of a Friend (2004)
4. Golbeck, J., Hendler, J.: FilmTrust: Movie Recommendations using Trust in Web-based Social Networks. In: Proc. IEEE Consumer Communications and Networking Conference (2006)
5. Heath, T., Motta, E.: Ease of Interaction plus Ease of Integration: Combining Web2.0 and the Semantic Web in a Reviewing Site. *Journal of Web Semantics*, 5 (to appear)
6. Brickley, D., Miller, L.: FOAF Vocabulary Specification 0.9 (2007), <http://xmlns.com/foaf/0.1/>
7. Newman, R., Russell, S., Ayers, D.: Tag Ontology (2005), <http://www.holygoat.co.uk/owl/redwood/0.1/tags/>
8. Oldakowski, R., Bizer, C., Westphal, D.: RAP: RDF API for PHP. In: ESWC 2005. Proc. 1st Workshop on Scripting for the Semantic Web, 2nd European Semantic Web Conference (2005)
9. W3C Technical Architecture Group: httpRange-14: What is the range of the HTTP dereference function? (2005), <http://www.w3.org/2001/tag/issues.html>
10. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A Core of Semantic Knowledge - Unifying WordNet and Wikipedia. In: WWW 2007. Proc. 16th International World Wide Web Conference (2007)
11. Auer, S., Lehmann, J.: What have Innsbruck and Leipzig in common? Extracting Semantics from Wiki Content. In: ESWC 2007. Proc. 4th European Semantic Web Conference (2007)
12. Bizer, C., Cyganiak, R., Gauss, T.: The RDF Book Mashup: From Web APIs to a Web of Data. In: ESWC 2007. Proc. 3rd Workshop on Scripting for the Semantic Web, at 4th European Semantic Web Conference (2007)

# Potluck: Semi-ontology Alignment for Casual Users

David F. Huynh, Robert C. Miller, and David R. Karger

MIT Computer Science and Artificial Intelligence Laboratory

32 Vassar St., Cambridge, MA 02139, USA

{dfhuynh, rcm, karger}@csail.mit.edu

**Abstract.** Potluck is a web user interface (Figure 1) that lets casual users—those without programming skills and data modeling expertise—repurpose heterogeneous Semantic Web data. It lets users merge, navigate, visualize, and clean up data all at the same time, using direct visual manipulation. This iterative process of integrating the data while constructing useful visualizations is desirable when the user is unfamiliar with the data at the beginning—a common case—and wishes to get immediate value out of the data without having to spend the overhead of completely and perfectly integrating the data first.

**Keywords:** mash up, drag and drop, faceted browsing, simultaneous editing, ontology alignment, end-user programming, semantic web, RDF.

## 1 Introduction

A central theme of the Semantic Web is of casual users collecting, merging, and repurposing information from multiple sources. But most tools that support this vision—descriptive logics, reasoners, and graph-based visualizers—posit data that has extensive schema information, and are far too sophisticated for casual users. We describe Potluck, a tool that allows casual users—who may not even know what an ontology is—to collect *messy* semantic web data from multiple sources, merge it into a coherent ontology, and visualize and navigate it effectively.

Potluck provides an *instant gratification* demonstration of the Semantic Web's benefits. Users who find multiple useful sources of semantic web data can instantly merge them into a single blob in their web browser. That the data is structured means faceted navigation and structured visualization of the data are immediately available. Then, rather than thinking hard about proper ontologies or writing functional descriptions of data transformation, users visually manipulate the data until it *looks right* for their purposes. Through direct manipulation, they align ontologies (unknowingly), clean up instances, create useful facets for navigation, and produce useful merged views for visualization. The end result can be saved as RDF that another user can consume for yet other purposes. Potluck eliminates the artificial separation between understanding and cleaning the collected data and using it to solve the user's problem, letting her tackle both at once.

Unlike automated alignment tools that rely on extensive ontological markup, Potluck works with “raw” RDF—it needs no schematic information beyond what is implicit in the properties used in the data. This offers several benefits:

- Far more raw RDF is available than “cooked”. For example, DBpedia contains almost 100M triples, but almost no schema information. Users can use Potluck to interact with a subset of this data.

- By demonstrating that users can interact with raw RDF, Potluck shows that publishing Semantic Web data is easy: authors can publish data instances without having to picking the right ontologies.
- Even providers who *want* to create nice ontologies are likely to do it “wrong” for some users. Potluck lets these users correct the “errors” themselves.

Potluck is therefore an effective deployment tool for the Semantic Web, answering the well known “chicken and egg” complaint that good semantic web tools and good semantic web data are stuck waiting for each other.

## 2 Scenario

Before describing the user interface of Potluck, we motivate it with a scenario that illustrates various idiosyncrasies of personal mash-up construction. Let us be optimistic that within a decade, the Semantic Web will be prevalent and RDF data will be everywhere. This scenario argues that even in this future world, users will *still* face problems making mash-ups between data sources.

In 2017, a historian named Henry is documenting the first cases of a rare genetic disease called GD726. These first cases occurred in the Valentine family in the 1820s. He wants to include in his final report a genealogical tree of the Valentine family, annotated with the disease’s infliction, as well as a comprehensive table of the Valentines’ data in an appendix.

Like most historians, Henry is not a programmer but he is experienced in managing data in his work. The proliferation of RDF means that he does not need programming skills to scrape HTML himself: all information needed for his research has been converted into RDF by various independent organizations and individuals. Henry thinks it would be trivial to pool the RDF together and call it done.

Henry tracks down various birth certificate and death certificate issuing offices where the Valentines lived for their RDF data. Some offices use `dc:date` in their data to mean “birth date,” some to mean “death date,” and some “certificate issuing date.” Thus, considering all `dc:dates` the same would be disastrous.

Henry also tracks down hospital records, which contain `hospital:tod` (short for “time of death”). Hence, `hospital:tod` is equivalent to some of the `dc:dates`. It would be hard to match `hospital:tod` with `dc:date` based on string analysis alone, yet match for some of the cases only.

The records all have geographical location names, but these names are not fully qualified. Those responsible for digitizing them thought that since all locations were within their country, the country name was redundant. Consequently, Henry needs to append the country name to all location names in order to map them.

People’s names are encoded in two different forms: “first-name last-name” in some data sets and “last-name, first-name” in others. Nick names are also present (e.g., “Bill” instead of “William”, and “Vicky” instead of “Victoria”).

The hospital records also pose problems. While most of their admittance dates are in ISO 8601 format, a few are of the kind “Easter Day 1824.” Such sloppiness has been observed in industrial and institutional databases, and should be expected on the Semantic Web.



Despite all these problems, there is one good thing about the data: Henry can reliably get the mother and father of each Valentine through the **gen:mother** and **gen:father** predicates, which seem to be very widely adopted. This helps Henry construct a genealogical tree visualization. However, as males and females both have equal chance of passing on GD726, Henry wants to treat **gen:mother** and **gen:father** the same while tracing the disease through the family. Unfortunately, adding an **owl:sameAs** equivalence between those two predicates will break his genealogical tree.

Potluck - Browse - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://dfhuyrh.csail.mit.edu:6666/potluck/system/template/browse

## Potluck

Data origins: MIT CSAIL Faculty web site and CCNMTL Staff web site

VIEW AS: TABLE • MAP

expand all details • collapse all details

Facets

origin

- 33 ✓ CCNMTL Staff web site
- 92 ✓ MIT CSAIL Faculty web site

drag and drop fields to create facets

drag and drop fields to create columns

- Alan Edelman**

label: Alan Edelman

origin: MIT CSAIL Faculty web site

email: edelman@csail.mit.edu

floor: 6th floor

group: unknown

last-name: Edelman

office: 32-G606

phone: 253-1355

photo: <http://www.csail.mit.edu/PI/72dp/edelman.jpg>

position: faculty

tower: Gates

url: <http://www.csail.mit.edu/rmt.php?PeopleID=75>

type: Person
- A. Maurice Matiz**

label: A. Maurice Matiz

origin: CCNMTL Staff web site

bio: Maurice is a co-founder of the Columbia Center for New Media Teaching and Learning and currently pla ... (82) in Computer Science from the School of Engineering and Applied Science at Columbia University.

building: Butler Library

campus: Morningside

email: maurice@columbia.edu

first: Maurice

group: Management; Technology

imageURL: <http://ccnmtl.columb.edu/headshots/Matiz.jpg>

last: Matiz

phone: 212 854-2665

room: 505A

title: Vice Executive Director, Director of Technology

type: Staff
- Alan Willsky**

label: Alan Willsky

origin: MIT CSAIL Faculty web site

email: willsky@mit.edu

floor: 5th floor

group: stochastic systems group

Done

**Fig. 1.** Potluck’s user interface shows data mixed from two different sources. Fields are rendered as draggable “field tags,” color-coded to indicate their origins.

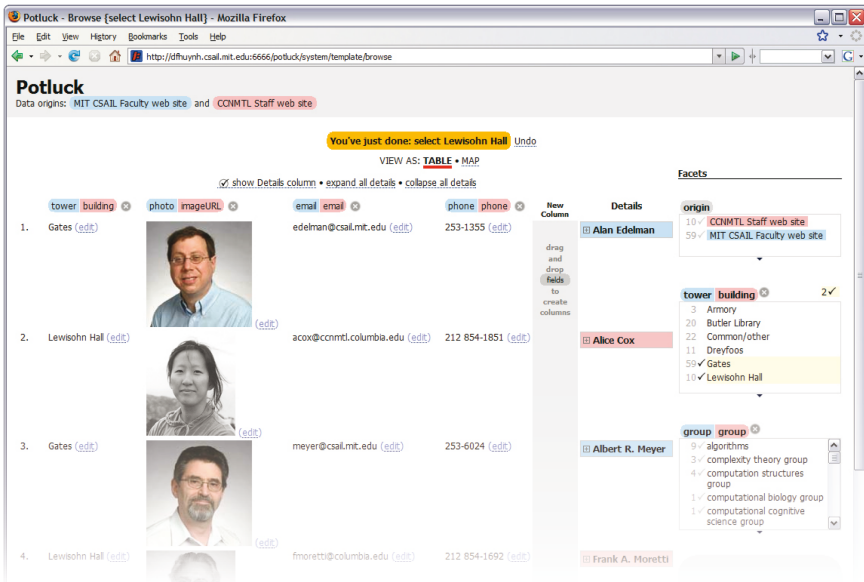
While all parties involved in this scenario acted logically and responsibly, Henry still ends up with a mess of RDF. To fix up the data, Henry must be able to:

- Merge `dc:dates` into *several* groups (the birth dates and the death dates) even though they all use the same predicate URI.
- Merge `gen:mother` and `gen:father` together in some situations while keeping them separate in other situations. This precludes the simple approach of adding `owl:sameAs` statements in the data model to implement equivalences.
- Edit the data efficiently to unify its syntax.
- Fix up the data iteratively as he learns more and more about the data.

### 3 User Interface

We now describe Potluck’s user interface, showing how it addresses the problems in the scenario above. The reader is encouraged to view a screencast to understand Potluck’s interactivity: <http://people.csail.mit.edu/dfhuynh/research/media/iswc2007/>. Potluck starts up with a text box where the user can paste in several URLs of Exhibit-powered web pages and click a button to yield the results in Figure 1, which lists data records from the original web pages. The records are interleaved by origins—the pages from which they have been extracted—to ensure that some records of each data set are always visible.

Fields are rendered as *field tags*: `label`, `position`, and `title`. Field tags are color-coded to indicate their origins: blue from one source and pink from another. Three core fields, `label`, `type`, and `origin`, are assigned to all records and their tags are colored gray. Fields from different origins having the same name are considered different—a crucial choice for `dc:date` in the above scenario.



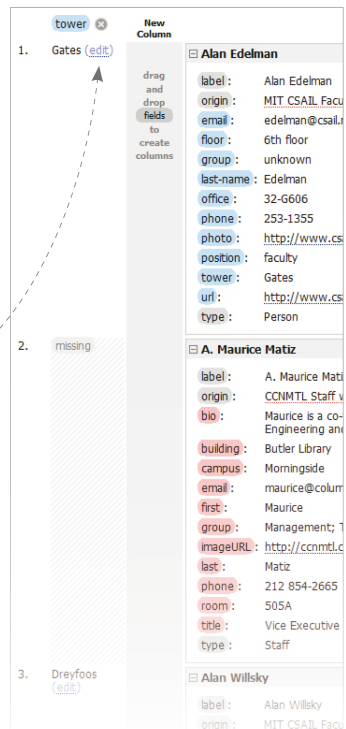
**Fig. 2.** A screen shot of Potluck showing several columns and facets of merged fields. The records’ details have been collapsed to make space for the columns.

**Creating columns and facets.** A field tag can be dragged and dropped onto the gray column to the left (Figure 1) to create a new column listing that field, or onto the gray box to the right to create a facet for filtering by that field. Figure 3 shows a newly created column. A column or facet can be moved by dragging its field tag and dropping the tag between other columns or facets. Deleting a column or facet (by clicking its ✕) removes the column or facet from the display but does not delete the corresponding field’s data.

**Merging fields.** A field tag can be dropped onto an existing column or facet to make that column or facet contain data for both the original field and the newly dropped field. Such an operation creates a *merged field*, whose field tag is rendered as a visual juxtaposition of the original tags in a pill-shaped form `position title`. Figure 2 shows several columns and facets of merged fields. Merged field tags can be dragged and dropped like elemental field tags can to create new columns and facets, or to merge into other existing columns and facets.

Creating a merged field does not disturb the elemental fields. Thus, in the scenario, it would be easy to have `gen:mother` and `gen:father` merged together for one purpose while keeping them separate for another purpose, all at the same time. Furthermore, the merging operation is *not* transitive, so that, say, merging fields `mother` and `father` together (to mean `parent`) and then `mother` and `grandmother` together (to mean `female ancestor`) does *not* force all three fields to be merged into `mother/father/grandmother`.

**Simultaneous editing.** The edit link next to each field value opens up the Simultaneous Editing dialog box where the values of that field can be edited *en masse* (Figure 4). The concept of simultaneous editing originated from LAPIS [6], a text editor that displays several keyboard cursors simultaneously on a text document, generalizes the user’s editing actions at one cursor, and applies them to the text at the rest of the cursors. Based on the user’s mouse clicks, LAPIS guesses how to divide the text document into records (often into lines or paragraphs) and where the cursors should be placed within those records (e.g., after the second word of the third sentence in each paragraph). Whereas LAPIS has to guess what a record is for the purpose of simultaneous editing, Potluck already has the field values conveniently separate. Potluck groups field values into columns by structural



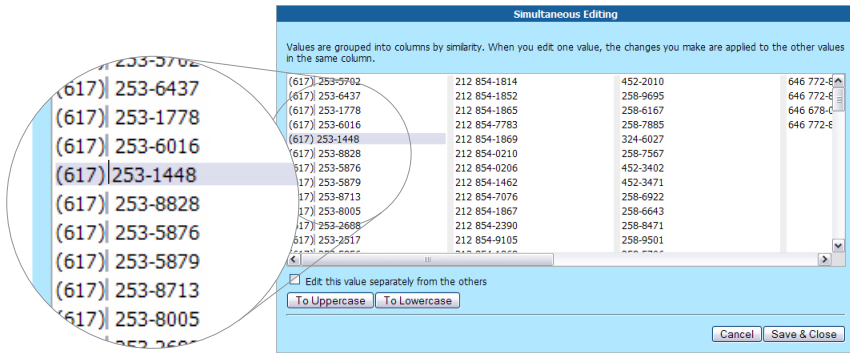
**Fig. 3.** Potluck renders a new column to the left when `tower` is dropped into the New Column drop target. Since the second record is not from the same origin as the dropped field, its cell in that column shows `missing`.

similarity, e.g., the phone numbers in the second column all have area code 212. These columns serve to visually separate out values of different forms, call out outliers (such as “Easter Day 1824” in the scenario), and let the user edit different forms differently. The user can click on any field value to give it keyboard focus, and editing changes made to it are applied to other values in the same column in a similar way. The multiple cursors in Figure 4 give visual feedback of the simultaneous editing operations in progress.

Simultaneous editing is useful for correcting inconsistencies between data sets that occur many times, such as prefixing area codes to phone numbers; for reformatting a field, such as changing “first-name last-name” into “last-name, first-name”; and for making a new field out of an existing field, such as extracting building numbers (32) from within office numbers (32-582).

**Faceted browsing** [8] lets a set of records be filtered progressively along several dimensions in any arbitrary order. It is useful in Potluck as Potluck often handles multidimensional data. Exploration through such data is needed for selecting out just the desired subset and for isolating records that need cleaning up.

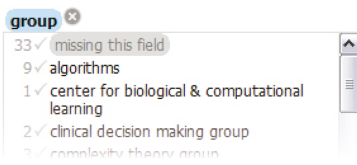
Potluck extends faceted browsing for the mash-up task in which data arrives from many sources. First, if within a facet there are records missing the corresponding field, the facet explicitly shows a choice for filtering to them (Figure 5). This visual element, not present in conventional faceted browsing interfaces, also serves to remind the user that, if that field is an elemental field instead of a merged



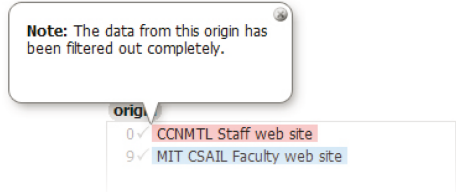
**Fig. 4.** The Simultaneous Editing dialog box lets the user change several similar values simultaneously by editing any one of them

field, the field is not present for records in other data sets. Second, whenever a facet choice causes all records from an origin to be filtered out completely, that origin remains in the origin facet and a warning message is displayed (Figure 6).

**Visualizations.** Potluck currently provides two visualizations: a tabular view and a map view. Figure 7 shows the map view in which any field containing street addresses or latitude/longitude pairs can be dropped onto the map view to plot the records. The map markers can also be color-coded using drag and drop. Faceted browsing is supported concurrently so that the user can construct a map while browsing through the data at the same time.



**Fig. 5.** If inside a facet there are records missing the corresponding field, the facet shows `missing this field` as a choice to get to those records

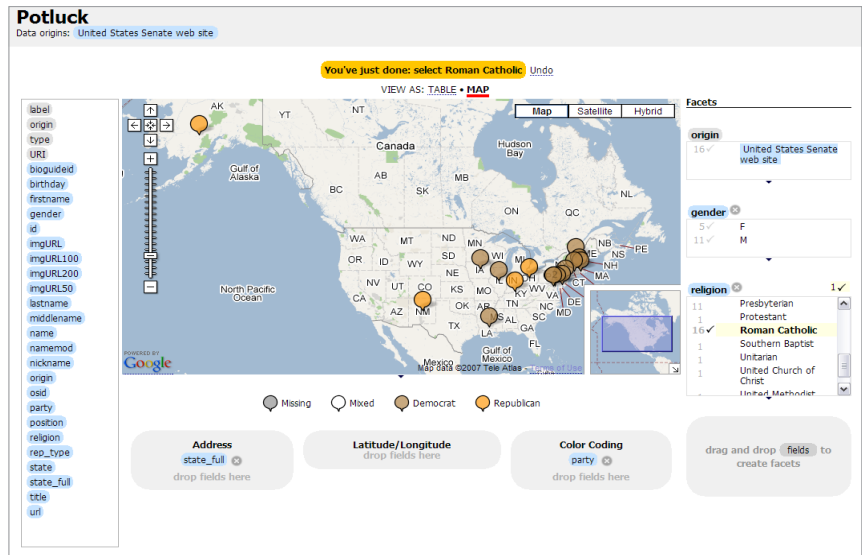


**Fig. 6.** The origin facet does not remove choices for which there are no records. Moreover, it pops up messages to call the user's attention to those filtered out origins.

## 4 Related Work

We discuss related work at greater length in our ISWC paper [4]. End-user tools such as Dapper [1] scrape information from multiple HTML pages. The proliferation of structured data on the Web will hopefully eliminate the need to scrape fragile HTML, allowing tools such as Piggy Bank [3] and Tabulator [2] to collect it from multiple already-structured Semantic Web sources. Even so, the data still has to be cleaned up and aligned before it can appear coherent to the user and thus become useful. The amount of broken HTML code on the present Web forebodes

An exception to these tools is WebScripter [7], which offers casual users data alignment features to create coherent tables of data collected from several sources. However, WebScripter does not encourage users to browse and visualize the data while aligning it, and offers no features for fixing data at the syntactic level, and it has not been formally evaluated on actual users.



**Fig. 5.** Potluck's map view allows plotting and color-coding records by dropping field tags into drop target areas. Faceted browsing is also offered during map construction.

Research data-alignment tools have been built mostly for experts and research has focused primarily on data modeling theories and automated agents for ontology alignment (surveyed in [5]) rather than on user interfaces for making practical use of aggregated data. They implicitly assume that users work with the data in delineated stages, first aligning the data and cleaning it up, and then using it in some other tools. We believe that users actually work iteratively on data, switching from aligning and cleaning up the data to using the data, and back, as they get to know the data better over time. Potluck explicitly supports this approach.

## 5 Conclusion

This paper presented Potluck, a tool for casual users—those without programming skills and data modeling expertise—to “mash up” data by themselves. Potluck is novel in its use of drag and drop for merging fields, its integration and extension of faceted browsing for focusing on subsets of data to align, and its application of the simultaneous editing technique for cleaning up data syntactically. Potluck also lets the user construct rich visualizations of data in-place as the user aligns and cleans up the data. This iterative process of integrating the data while constructing useful visualizations is desirable when the user is unfamiliar with the data at the beginning—a common case—and wishes to get immediate value out of the data without having to spend the overhead of completely and perfectly integrating the data first. It thus offers instant gratification to end users who want to *do something* with their data without stopping to solve ontology alignment problems.

## Acknowledgements

This work was supported by the National Science Foundation (award number IIS-0447800), Nokia, and the Biomedical Informatics Research Network. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the funders.

## References

- [1] Dapper: The Data Mapper, <http://www.dapper.net/>
- [2] Berners-Lee, T., et al.: Tabulator: exploring and analyzing linked data on the Semantic Web. In: SWUI 2006 (2006)
- [3] Huynh, D., Mazzocchi, S., Karger, D.: Piggy Bank: Experience the Semantic Web Inside Your Web Browser. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
- [4] Huynh, D., Miller, R., Karger, D.: Potluck: Data Mash-Up for Non-programmers. In: ISWC 2007 (2007)
- [5] Kalfoglou, Y., Schorlemmer, M.: Ontology Mapping: The State of the Art. In: The Knowledge Engineering Review, 18th edn. (2003)
- [6] Miller, R., Myers, B.: Multiple Selections in Smart Text Editing. In: IUI 2002 (2002)
- [7] Yan, B., Frank, M., Szekely, P., Neches, R., Lopez, J.: WebScripter: Grass-roots Ontology Alignment via End-User Report Creation. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, Springer, Heidelberg (2003)
- [8] Yee, P., Swearingen, K., Li, K., Hearst, M.: Faceted Metadata for Image Search and Browsing. In: CHI 2003 (2003)

# Semi-automatic Ontology Engineering Using Patterns

Eva Blomqvist

Jönköping University, Jönköping, Sweden  
blev@jth.hj.se

**Abstract.** Many enterprise systems face the same kind of information processing problems that exist on the web in general, and creating semantic solutions often involve constructing an enterprise ontology. Ontology engineering in turn needs to be semi-automatic in order to reduce the effort and need for expertise. By introducing knowledge reuse in ontology construction the construction effort can be further reduced and the quality of the ontology output increased. The proposed research focuses on a hybrid approach for ontology construction based on the methodology of case-based reasoning in combination with ontology patterns.

## 1 Introduction

Semantic Web techniques are not only applicable to the "public" web, similar problems can be found also in the business world, for example on large company intranets. Development of semantic applications for enterprises most often involve constructing an enterprise ontology for the company in question. Manual ontology engineering is a tedious and complex task, therefore most recent research focus on semi-automatic approaches. One issue addressed by our research is automation throughout the construction process. Another issue is knowledge reuse; common practises of the business world should be exploited, as well as best practises in ontology engineering. To learn from past experience partial solutions from already constructed ontologies can be collected. Our intention is to develop a hybrid approach (named OntoCase) that combines the learning viewpoint from case-based reasoning (CBR) with more consensual knowledge reuse through patterns.

## 2 Background and Related Work

Our research is not restricted to a specific ontology representation formalism but assumes for some parts the possibility to represent the ontology as a (semantic net-like) graph. The research focuses on application ontologies within enterprises, here denoted enterprise application ontologies, used mainly for structuring and retrieval of information.



## 2.1 Ontology Engineering

Recent developments in ontology engineering involve semi-automatic ontology construction, or ontology learning (OL). Most OL approaches focus on techniques for text analysis in order to extract mainly concepts and relations from a text corpus. Existing approaches are for example Text2Onto [1], OntoGen [2] and Abraxas [3]. One major issue is the problem of background information not being explicitly stated in a text document. Some approaches use additional sources like WordNet (see [4]), or the web, to find and validate missing information. Patterns can also serve as such a source of additional information and structure. Recent research also tries to apply OL techniques throughout the complete ontology life-cycle and treat sets of interconnected ontologies (as suggested by [5]).

Four abstraction levels of patterns describing the internal structure of an ontology have been discussed in [6] (syntactic, semantic, design and architecture patterns). Here we focus on design and architecture patterns for semi-automatic use. An ontology design pattern is an ontology template intended to construct a part of some ontology, which is self-contained and comprised of a set of ontology primitives. Related work in ontology design patterns focus on templates mainly for manual use (e.g. [7]), but recent research efforts (like [5]) intends to also extend this into OL. The architecture of an ontology describes the components from which it is comprised, their dependencies and topology. In our method, architecture patterns will mainly be sets of constraints guiding and restricting the composition of the ontology from design patterns.

One major task involves how to assess pattern relevance. This problem is quite similar to ranking in an ontology search engine, as for example in [8], and also uses basic techniques from ontology matching (see the survey in [9]). The main differences are that in OntoCase we have a much richer input structure (the "query") as compared to ontology search, whereas compared to the general case of ontology matching we have some very specific characteristics of the input ontologies and additionally specific requirements on the matching result.

## 2.2 Case-Based Reasoning

Case-based reasoning (or CBR, see [10]) is a methodology for using previous experience to solve new problems. A case is a problem situation, and previously encountered cases are stored in a case base. The CBR process is viewed as a cycle of four phases. The first phase is retrieval, where a new case arrives, its representation is derived and used to find relevant stored cases. The second phase is reuse, where the retrieved case(s) are adapted and used to form a new solution. Next, the solution is evaluated and revised, and finally relevant parts of the new solution are retained and stored. Our approach falls within textual CBR (TCBR), where parts of cases or solutions are natural language texts (see [11]). The first and second phase of OntoCase corresponds to current research questions of TCBR as stated in [11]. The CBR idea is quite similar to the idea of using patterns, but CBR takes an even more pragmatic view by basing the solutions only on previous experience. We would like to construct a hybrid approach that exploits both best practises and new experiences.



### 3 Research Questions

In our research we have set a number of long-term goals based on the open issues in current OL research. From these long-term goals general research questions have been derived:

- How can the effort of constructing an enterprise application ontology be reduced?
  - How can CBR improve semi-automatic ontology construction?
- How can knowledge and experience be reused in ontology engineering?
  - How can ontology patterns be exploited in the semi-automatic ontology construction process?

Based on the research questions above the following hypotheses have been stated:

- Automation reduces the total construction effort.
  - CBR gives a framework for further automation of ontology construction, compared to related semi-automatic approaches that exist today.
- Domain knowledge and engineering experience can be reused through ontology patterns.
  - The CBR methodology together with patterns can improve the quality of generated ontologies, compared to existing semi-automatic approaches.

To test the hypotheses OntoCase is being developed based on CBR and the notion of ontology patterns. To verify the hypotheses the method must be evaluated and compared to related OL approaches. So far only minor parts of the approach have been evaluated. For the evaluation of the overall approach we envision the use of a "gold standard" ontology developed within one of our projects in cooperation with industry, to compare our results with achievable results of existing available OL systems.

### 4 Proposed Approach

The basis of a CBR approach is the case base, which in OntoCase corresponds to a pattern catalogue (pattern base), containing both ontology design and architecture patterns. The design patterns are represented as small ontologies (see example in [\[12\]](#)). Partly due to the lack of available enterprise ontologies the patterns have so far been constructed based on other sources, like data model patterns and textbooks on organisations. The architecture patterns are sets of constraints on the combination of design patterns. Although the CBR cycle can be illustrated as a linear process, iteration is of course present internally in the phases, also it is intended that the phases can be applied individually depending on already available input.

The retrieval phase constitutes of analysing the input text corpus and deriving its representation, then matching this to the pattern base and selecting appropriate patterns. The reuse phase concerns the adaptation of the patterns, combining them into an ontology. The revise phase includes extending the ontology, based on evaluation results. Retaining patterns includes the discovery of new patterns and improving existing patterns. So far retrieval and reuse has been the main focus of research, revise and retain are still future work.

## 4.1 Achieved Results

In [12] an industry project was used as an application case for a simple implementation of the first two phases. The evaluations described in [13] gave rise to the current research questions and hypotheses as stated previously. The main problems that were discovered could be connected to missing information in the text corpus and pattern base, and the lack of a general architecture. We envision the refinement of the approach as described below, and the additional two phases (revise and retain) to remedy many of those problems.

The initial step of the retrieval phase, extracting a representation of the text corpus, is considered mainly outside the scope of this work. We feel this is an area where already a lot of approaches exist. A small initial evaluation was conducted, comparing recent OL systems with more basic text analysis techniques available in standard components. The term output was compared to a manually constructed "gold standard". The OL system being tested almost doubled the precision compared to a combination of basic techniques, and improved recall around 30%. So far this is our motivation for trying to build on recent experimental algorithms instead of using only standard text analysis components.

The second step of the retrieval phase concerns choosing suitable patterns. For this, a pattern ranking scheme has been developed, partly inspired by ontology ranking as well as techniques in ontology matching. Compared to ontology search engines, our pattern ranking exploits a much richer input structure since the complete input representation is used. This has led to a ranking scheme of four measures; concept coverage, relation coverage, density, and proximity.

The second phase (reuse) involves first composing the initial ontology from the patterns, essentially ontology merging. This has so far been a simple process applying heuristics for resolving overlap between patterns. Furthermore, the phase contains an enrichment step where the initial ontology is extended using parts of the input representation until a suitable coverage (over the input) has been reached. External knowledge sources, like WordNet, are used for placing the input primitives in the context of the resulting ontology without loss of quality.

In our approach there is an aspect of uncertainty inherent in all the described steps. For example, each ontology primitive of the input representation has a certain degree of confidence associated. The levels of confidence are transferred onto the constructed ontology. If a standard ontology representation is required, thresholds can be set or the ontology can be validated by a user.

## 4.2 Next Steps

The two last phases, revise and retain, are still not elaborated in our approach. The revision phase is intended to contain steps for evaluating and further revising the ontology, to increase the input coverage but also for example to reduce the level of redundancy and resolve possible conflicts in the ontology. The final phase of retaining patterns is concerned with collecting feedback for applied patterns, but also to discover new pattern candidates directly from the resulting ontology. Since this is denoted a hybrid approach, in addition to the pattern discovery step, patterns can be constructed manually (but the development of a manual

pattern construction method is considered outside the scope of this work). The idea is to benefit both from the CBR view (reuse of solutions) and the pattern view (reuse of best practises).

## 5 Expected Contribution and Future Work

The main contributions of our ontology engineering approach are envisioned as both further automation of the ontology construction process, but in addition an increased quality of the produced ontology, as compared to existing OL approaches. This increased quality is mainly due to the use of patterns, representing both expert knowledge and previous experience, and the introduction of several revision steps during the four phases of the method.

Future work contains refinement of the first two phases, as well as implementation of the improvements in the prototype system. Also to treat the third and fourth phases in detail and include these in an implementation. Some open issues are how to automatically find and extract patterns, as well as how to extend the ontology in the revision phase. Probably external sources, like the web, have to be used to attach and validate missing pieces of the ontology. A major step in all research is of course evaluation of the approach, in this case to compare the result to the result of manual approaches as well as the related OL approaches stated earlier. These evaluations are so far not planned in detail.

## References

1. Cimiano, P.: *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications*. Springer, Heidelberg (2006)
2. Fortuna, B., Grobelnik, M., Mladenic, D.: *Semi-automatic Data-driven Ontology Construction System*. In: *Proc. of IS 2006*, Ljubljana, Slovenia (2006)
3. Iria, J., Brewster, C., Ciravegna, F., Wilks, Y.: *An Incremental Tri-partite Approach to Ontology Learning*. In: *Proc. of LREC 2006*, Genoa (May 2006)
4. Fellbaum, C., et al.: *WordNet - An Electronic Lexical Database*. MIT Press, Cambridge (1998)
5. NeON Website, available at <http://www.neon-project.org/>
6. Blomqvist, E., Sandkuhl, K.: *Patterns in Ontology Engineering: Classification of Ontology Patterns*. In: *Proc. of ICEIS 2005*, Miami Beach, Florida, May 24-28 2005 (2005)
7. Gangemi, A.: *Ontology Design Patterns for Semantic Web Content*. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005*. LNCS, vol. 3729, pp. 262–276. Springer, Heidelberg (2005)
8. Alani, H., Brewster, C.: *Ontology Ranking based on the Analysis of Concept Structures*. In: *Proceedings of K-CAP 2005*, Banff, Alberta, Canada (October 2005)
9. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer, Heidelberg (2007)
10. Aamodt, A., Plaza, E.: *Case-based reasoning: Foundational issues, methodological variations, and system approaches*. *AICom* 7, 39–59 (1994)
11. Weber, R.O., Ashley, K.D., Brüninghaus, S.: *Textual case-based reasoning*. *The Knowledge Engineering Review* 20(3), 255–260 (2006)
12. Blomqvist, E.: *Fully automatic construction of enterprise ontologies using design patterns: Initial method and first experiences*. In: *Proc. of ODBASE (2005)*
13. Blomqvist, E., Öhgren, A., Sandkuhl, K.: *Ontology Construction in an Enterprise Context: Comparing and Evaluating two Approaches*. In: *Proc. of ICEIS (May 2006)*

# From Texts to Structured Documents: The Case of Health Practice Guidelines

Amanda Bouffier

Laboratoire d'Informatique de Paris-Nord  
Université Paris 13 and CNRS UMR 7030  
99, av. J.-B. Clément – F-93430 Villetaneuse  
firstname.lastname@lipn.univ-paris13.fr

**Abstract.** This paper describes a system capable of semi-automatically filling an XML template from free texts in the clinical domain (practice guidelines). The XML template includes semantic information not explicitly encoded in the text (pairs of conditions and actions/recommendations). Therefore, there is a need to compute the exact scope of conditions over text sequences expressing the required actions. We present in this paper the rules developed for this task. We show that the system yields good performance when applied to the analysis of French practice guidelines.

**Keywords:** Health practice guidelines, Natural Language, GEM (Guideline Elements Model).

## 1 Introduction

During the past years, clinical practices have considerably evolved towards standardization and effectiveness. A major improvement is the development of practice guidelines [1]. However, even if widely distributed to hospitals, doctors and other medical staff, clinical practice guidelines are not routinely fully exploited. There is now a general tendency to transfer these guidelines to electronic devices (via an appropriate XML format). This transfer is justified by the assumption that electronic documents are easier to browse than paper documents.

The Guideline Elements Model (GEM) is an XML-based guideline document model that can store and organize the heterogeneous information contained in practice guidelines [2]. It is intended to facilitate translation of natural language guideline documents into a format that can be processed by computers. The main element of GEM, `knowledge component`, contains the most useful information, especially sequences of conditions and recommendations. Our aim is to format these documents, which have been written manually without any precise model, according to the GEM DTD.

One of the main problem for the task is that the scope of the conditional segments (i.e all the recommendation segments that have to be linked with a condition) may exceed the sentence boundaries and, thus, include several sentences. In other words, sequences of conditions and recommendations correspond to *discourse* structures.

Discourse processing requires the recognition of heterogeneous linguistic features (especially, the granularity of relevant features may vary according to text genre [9]). Following these observations, we made a study based on a representative corpus and automatic text mining techniques, in order to semi-automatically discover relevant linguistic features for the task and infer the rules necessary to accurately structure the practice guidelines.

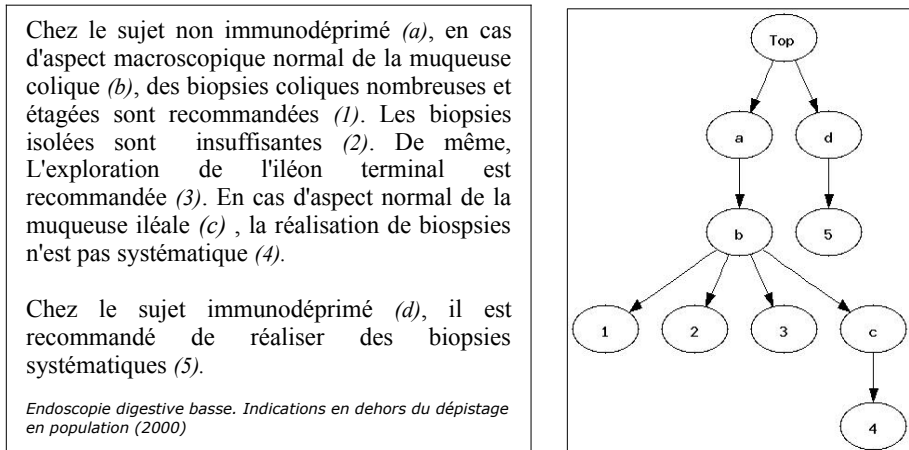
The paper is organized as follow: first, we present the task and some previous approaches (section 2). We then describe the rules for text structuring (section 3) and the method used to infer them. We finish with the presentation of some results (section 4), before the conclusion.

## 2 Document Restructuring: The Case of Practice Guidelines

As we have previously seen, practice guidelines are not routinely fully exploited. To overcome this problem, national health agencies try to promote the electronic distribution of these guidelines.

### 2.1 Previous Work

Several attempts have already been made to improve the use of practice guidelines. For example, knowledge-based diagnostic aids can be derived from them [3]. GEM is an intermediate document model, between pure text (paper practice guidelines) and knowledge-based models like GLIF [4]. GEM is thus an elegant solution, independent from any theory or formalisms, but compliant with other frameworks. Previous attempts to automate the translation process between the text and GEM are based on the analysis of isolated sentences and do not compute the exact scope of conditional segments [5].



**Fig. 1.** From text to a tree representing the scope of conditional segments

## 2.2 Our Approach

Our aim is to semi-automatically fill a GEM template from existing guidelines: the algorithm is fully automatic but the result needs to be validated by experts to yield adequate accuracy. We first focus on the most important part of the GEM DTD knowledge Component which is sequences of conditions and recommendations.

We propose a two-step strategy: 1) basic segments (conditions and recommendations) are recognized and 2) the scope of the conditional segments is computed. In this paper, we focus on the second step, which is the most difficult one and has not been solved by previous systems. What is obtained in the end is a tree where the leaves are recommendations and the branching nodes are conditional segments, as shown on Figure 1. All the children of a node are under the scope of the parent node.

## 3 Structuring Rules

We set up a representative corpus in order to infer a set of rules able to decide if a segment  $s$  is (or is not) under the scope of a conditional segment  $c$ .

### 3.1 Material and Method

The training corpus consists of 25 French Practice Guidelines (about 150 000 words, see <http://anaes.fr>). This corpus has been annotated by a domain expert, who had to manually recognize conditions and recommendations, and link them according to the tree structure described in 2.2. We have built on this basis a set of examples, each example being a couple  $(c, s)$  where  $c$  is a conditional segment and  $s$  is a segment under the scope of  $c$  (positive examples) or not (negative examples). All the examples are represented by a set of 17 potential interesting linguistic features. The list of these features has been identified from the relevant literature (e.g. [6] [7] [8]) and a manual study of the practice guidelines. More precisely, the features belong to the 4 following categories:

**1) Material text structure.** Let's take the feature: "has\_the\_same\_visual\_position( $c,s$ )". Its value for the couple  $(a,d)$  in Figure 1 is "true" because  $(a)$  and  $(d)$  are both at the beginning of a paragraph. Another attribute concerns the location of the condition: "is\_detached( $c$ )". The value of this feature for  $(c)$  in the figure 1 is "detached" because  $(c)$  is located at the sentence-initial position.

**2) Lexical relations.** Let's take the feature "have\_terms\_with\_an\_antonymic\_relation( $c,s$ )". Its value is "true" for the couple  $(a,d)$  because the terms *immunodéprimé* and *non immunodéprimé* are antonyms.

**3) Discourse connectors.** Let's take the feature "begins\_with\_a\_coordination\_marker( $s$ )". Its value is "true" for  $(a,3)$  because the proposition begins with *De même (also...)*.

**4) Co-reference relations.** This feature is useful to recognize co-references. In the medical domain, demonstrative noun phrases ("dans ce cas", "in this case...") is the most common way of marking co-reference.

We then used data mining algorithms (in particularly attributes selectors and rules learning algorithms) in order to statistically validate the potential interest of the different features, understand their relative contribution and derive the set of structuring rules.

### 3.2 The Rules Organized in Knowledge Levels

One of the main result of our study is that features related to the material structure of the text are the most discriminating ones for the task, using an “Information Gain” measure. First, the location of the condition in the sentence (detached or not) is especially important. If it is detached, it exerts in 70% of the cases an influence downward from the sentence in which it is located. Conversely, if the condition is integrated inside the sentence, its scope is limited to the sentence boundaries in 80 % of the cases. More generally, the scope of a conditional segment often complies with the material text structure. Thus, the rules which involve these features correspond to **norms**. This can be explained by the style of writing used for the “guidelines” text genre, that often makes use of visual information. The other kinds of attributes (discourse connectors, lexical relations co-coreference informations) are less discriminating simply because they are less frequent. Nevertheless, they are sometimes important since they may violate a norm and suggest a more accurate way of structuring the document. The rules which involve these features are called **exceptions**. Therefore, we have organized the structuring rules according the ability of a feature to contribute to the solution (norms vs. exceptions).

The **norms** represent the most discriminating rules and involve the most salient features which belong to the category of the material text structure.

- By default, IF  $c$  is syntactically integrated AND  $s$  is in another sentence of  $c$  THEN  $s$  is excluded from the scope of  $c$ . Conversely, IF  $c$  is syntactically integrated AND  $s$  is in the sentence of  $c$  THEN  $s$  is under the scope of  $c$ .
- By default, IF  $c$  is detached from the sentence AND  $s$  is in the same position in the material structure than  $c$  THEN  $s$  is excluded from the scope of  $c$ . Conversely, IF  $s$  is NOT in the same position in the material structure than  $c$  AND  $s$  and  $c$  are in the same paragraph THEN  $s$  is included in the scope of  $c$ .

These two rules can be violated if another set of rules called **exceptions** suggest a more accurate way of structuring the text. A first set of rules suggest an inclusion of  $s$  under the scope of  $c$ . For example, the following rule belongs to this category:

- IF  $s$  is in a co-reference relation with  $c$  THEN  $s$  is under the scope of  $c$ .

Conversely, some other rules suggest an exclusion of  $s$ . For example:

- IF  $s$  is preceded by a discourse coordination connector THEN  $s$  is excluded from the scope of  $c$ .

## 4 Evaluation

We evaluated the approach on a corpus that has not been used for training. The evaluation of basic segmentation gives the following results: .92 P&R<sup>1</sup> for conditional segments and .97 for recommendation segments. The scope of conditions is

---

<sup>1</sup> P&R is the harmonic mean of precision and recall ( $P\&R = (2 * P * R) / (P + R)$ ), corresponding to a F-measure with a  $\beta$  factor equal to 1.

recognized with accuracy above .7. This result is encouraging, especially considering the large number of parameters involved in discourse processing. In most of successful cases the scope of a condition is recognized by the default rule (default segmentation, see section 3).

## 5 Conclusion

We have presented in this paper a system capable of performing automatic segmentation of clinical practice guidelines. Our aim was to automatically fill an XML DTD from textual input. The system is able to process complex discourse structures and to compute the scope of conditional segments spanning several propositions or sentences. Moreover, our system is the first one capable of resolving the scope of conditions over several recommendations.

## References

- [1] Brownson, R.C., Baker, E.A., Leet, T.L., Gillespie, K.N.: Evidence-based public health. Oxford University Press, Oxford (2003)
- [2] Shiffman, R.N., Karras, B.T., Agrawal, A., Chen, R., Marenco, L., Nath, S.: GEM: A proposal for a more comprehensive guideline document model using XML. *Journal of the American Medical Informatics Assoc.* 7(5), 488–498 (2000)
- [3] Séroussi, B., Bouaud, J., Dréau, H., Falcoff, H., Riou, C., Joubert, M., Simon, G., Venot, A.: ASTI: A Guideline-based drug-ordering system for primary care. In: *Proceedings MedInfo*, pp. 528–532 (2001)
- [4] Peleg, M., et al.: GLIF3: The Evolution of a Guideline Representation Format. In: *Proceedings of the American Medical Informatics Association*, pp. 645–649 (2000)
- [5] Georg, G., Jaulent, M.-C.: An Environment for Document Engineering of Clinical Guidelines. In: *Proceedings of the American Medical Informatics Association*, pp. 276–280 (2005)
- [6] Charolles, M.: Framing adverbials and their role in discourse cohesion: from connexion to forward labeling. In: *Sem 2005. Papers of the Symposium on the Exploration and Modeling of Meaning*, Biarritz, France (2005)
- [7] Péry-Woodley, M-P.: Signalling in written text: a corpus-based approach. In: Stede, M., Wanner, L., Hovy, E. (eds.) *Proceeding of the Coling '98 Workshop on Discourse Relations and Discourse Markers*, pp. 79–85 (1998)
- [8] Halliday, M.A.K., Matthiessen, C.: *Introduction to functional grammar*, 3rd edn., Arnold. London, UK (2004)



# Knowledge Enhanced Searching on the Web

Afraz Jaffri

Dependable Systems and Software Engineering Group  
School of Electronics and Computer Science  
University of Southampton  
a.o.jaffri@ecs.soton.ac.uk

**Abstract.** The move towards a semantic web has been in progress for many years and more recently there have been applications that make use of semantic web technology. One of the features that made the Web so easy to use is the ability to search web pages in a matter of seconds through the use of search engines. Now that the use of OWL and RDF as a knowledge representation format is increasing, the possibility appears to improve the quality of searching by using the semantic web to enhance the ‘ordinary’ Web. This paper outlines an architecture for using distributed knowledge bases to assist and improve searching on the web.

## 1 Introduction

The semantic web promises a new generation of World Wide Web infrastructure that will make it possible for machines to ‘understand’ the data on the web instead of merely presenting it. The increased adoption of RDF and OWL as knowledge representation formats are enabling the production of semantic web systems that can manage, manipulate and display data in novel ways [1, 2]. However, there are also those who believe that the semantic web is merely a dream that will never be fulfilled [3]. In order to encourage the increase of semantic web technologies there have been suggestions that a ‘killer app’ may be needed to convince those that are still unsure about the benefits that semantic web technologies can bring.

The search engine is an example of a potential ‘killer app’ that has been responsible for increased usage of the current web. However, despite the improving quality of modern search engines, statistics show that only 17% of people find exactly the information they were looking for [4]. Furthermore, a study has shown that the recall of some search engines can be as low as 18% [5]. There is therefore a need to improve the quality of search results and user experience. The semantic web provides an opportunity to achieve such a goal. The use of RDF and OWL as knowledge representation formats can provide structured content to describe a given domain or set of domains. Using this knowledge, it should be possible to add a sense of ‘understanding’ to a search engine when searching for results whose knowledge has been partly or fully described in a knowledge representation format.

In the past, such a proposal may not have been viable due to the lack of ontologies and RDF resources available on the web. However, at the present time there are estimated to be more than 5 million RDF or OWL documents available on the web [6].

Even if most of those documents contain knowledge about a limited set of concepts, RDF data from sources such as DBpedia (<http://dbpedia.org/docs>) and Wordnet ([wordnet.princeton.edu](http://wordnet.princeton.edu)) provide a suitable base from which to begin exploring the enhancement that can be made to ordinary web searches. More importantly, there will be a number of knowledge bases that will be used to store RDF instance data and OWL ontologies that have the ability of being queried using the SPARQL query language. Therefore, there are a number of components that need to fit together in order to achieve semantically enhanced querying that will be presented in Section 4.

## 2 Related Work

There have been many projects under the general heading of ‘Semantic Search’ that work towards different goals and objectives. The Swoogle [7] search engine attempts to index all semantic web documents (SWD) on the web. The query a user makes is usually in order to find an ontology that they can use that contains descriptions about their query item. This is a purely semantic web service, i.e. it deals only with SWD and not any other type of document available on the web. There are a few ontologies that have a high rank because they are imported by other ontologies and therefore are returned frequently back to the user.

The SemSearch search engine integrates ontologies and RDF data to provide a search facility for a departmental university website [2]. Queries are semi-structured and require users to input a subject keyword as well as free text. This system is also a closed world system i.e. it does not interact with the web and it cannot make use of knowledge from other repositories.

The project that shares the same aims and objectives as those stated in this paper; to improve the quality of search on the web, is the TAP project [8]. TAP is both a semantic web application by itself and also has the ability to interact with the WWW. TAP does not try to model concepts or definitions, but instead concentrates on modelling real world entities such as movies, athletes, musicians, places, people etc. The only limitation with the system is that it does not bring in knowledge contained in other repositories. This means that when a user searches for things that are not inside the TAP knowledge base, very little useful information is returned.

The latest project that is being developed by the Linking Open Data ([linked-data.org](http://linked-data.org)) project is DBpedia. This consists of a large knowledge base containing structured information gathered from Wikipedia (<http://en.wikipedia.org>). The idea behind the project is that people will expose their data and interlink it with the RDF from DBpedia. Such a large scale repository provides an ideal starting point for implementing a semantic search engine and such a system has already been produced (<http://dbpedia.org/search>). However, DBpedia is by no means an exhaustive reference for every concept or entity. The project is also in the initial stages and not everyone can be expected to provide data that has been linked to their own. Nonetheless the SPARQL endpoint provided by DBpedia provides access to a rich set of knowledge that is being used in our system.

A gap exists in the domain of semantic searching that has not been covered by existing systems. The system proposed in this paper aims to be an open world semantic web application. The system will use heterogeneous knowledge bases that are

accessed through the SPARQL query language which means that any repository that has a SPARQL access point can be used in the system. The system will not be limited to a particular domain or a particular web site. Through the use of external links embedded in RDF, searches made on the entire web can be semantically enhanced as will be described in Section 4.

### 3 Consistent Reference Service

As described in Section 2 the DBpedia project is an effort to try and capture the information from structured data provided by Wikipedia. The RDF data they have gathered has also been linked with data from other sources through the use of owl:sameAs predicates. Whilst the community developing these projects advocates such linkage between disparate data sources, this may not always be practical. You may not be sure of which URI refers to the same entity as your own URI and how many such URI's exist on the semantic web. Our proposal is to use a CRS (Consistent Reference Service) to manage the referential integrity of semantic web resources. Each site or endpoint that provides access to RDF data maintains knowledge about 'bundles' of resources that it considers to be identical. This service has been implemented in the ReSIST project [9]. Thus, authors of papers who have different URI's from DBLP, Citeseer and their own site are bundled together with one URI chosen as the canonical representation. The CRS is being adapted for use in the system proposed in this abstract by being able to identify the similarity between concepts other than 'Person'.

### 4 Proposed Architecture

The system architecture comprises of three main components as shown in Figure 1. A user enters their keywords (KW) in a normal Google style search box without the need for using special syntax or constructs. These keywords are then fed to the knowledge manager who passes them on to the knowledge mediator.

The knowledge mediator has access to an arbitrary number of knowledge bases and a CRS that are accessed over HTTP using SPARQL. The mediator then queries the CRS for concepts that match or are similar to the given keywords. The CRS returns a 'bundle' of resources that have been found to be the same. The knowledge Mediator then issues a DESCRIBE query on each URI to find the properties and literals for each resource. The RDF returned is then passed on to the Knowledge Manager who looks for links such as 'foaf:page', 'rdfs:seeAlso' and 'dbpedia:reference' so that a list of web pages associated with the query can be returned. The knowledge manager then displays the knowledge and web links to the user according to the type of concepts returned. For example, the distributed knowledge bases contain definitions, articles, links, publications and other assorted information; these results are returned to the user as views relating to one of the concepts. The user can then select whichever view they choose to explore the results further. Formally:

Let *Concept* be the set of concepts contained in all knowledge bases.  
 Let *URI* be the set of URI's that represent each concept.  
 Let *Bundle* be the set of bundles whose elements are URI's representing a concept.  
 Let *V* be the set of property values of a URI.  
 We have a function *BundleOf* that returns one bundle for a given concept:

$$BundleOf : Concept \rightarrow Bundle$$

We have a function *Describe* that gives the property values for a given URI:

$$Describe : URI \rightarrow P(V)$$

The result of a query for a concept from the Knowledge Mediator is:

$$\forall a \in Concept, Result(a) = \bigcup_{i \in BundleOf(a)} Describe(i)$$

The results that are returned to the user are both document links and knowledge in the form of RDF statements that were returned from the knowledge mediator when the original query was sent. This enables the user to understand how the results were achieved. An investigation into whether the knowledge alone is enough to satisfy a user's query is planned as part of the research.

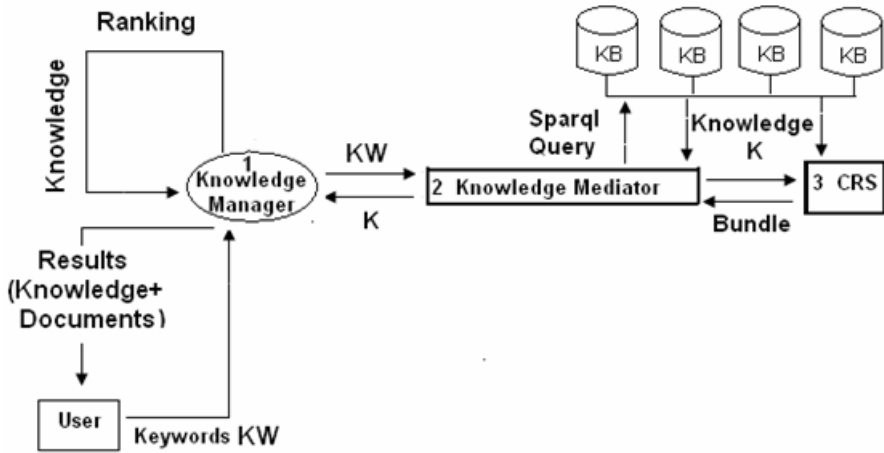


Fig. 1. The system architecture diagram shows the three main components numbered 1, (Knowledge Manager), 2, (Knowledge Mediator) and 3, (CRS)

## 5 Research Methodology and Future Work

The initial research that led to the architecture described in Section 4 being formed has been successfully completed. A proof of concept system is now being implemented that will take the form of a Google Maps mesh-up of Wikipedia(<http://labs.systemone.at/wikipedia3>), DBpedia, CIA Factbook (<https://www.cia.gov/cia/publications/factbook/index.html>) and Geonames ([www.geonames.org](http://www.geonames.org)) data. These four sources contain an extensive set of information about all the countries of the

world. The four sources each have their own SPARQL endpoint that can be queried from the web.

The interface for the system takes the form of a Google Map where the user can select any country. The system then performs queries over the RDF and returns information about the country from the different sources. The results are presented so that different types of information are separated on screen. The types of information vary from country to country and include geographical, social and political information as well as the people associated with a country and the events that have taken place in a country. Each type can be explored in more detail so that links to web sites can also be seen.

The CRS for the system identifies URI's from each knowledge base that refer to the same country. The DESCRIBE query issued to each URI then performs similarity matching on the properties to filter out duplicate entries. The Knowledge Manager then looks at each property to determine which type of information is being referred to. The results are then presented to the user under the Google Map.

Once the prototype system has been fully implemented the research will then focus on broadening out the searches for queries on any kind of entity, not just countries. The evaluation for the current system will be performed by users who will provide feedback so that any features can be included in a future version. The system will also be compared with Google and also with results obtained from each knowledge base individually to assess the improvements that can be made by distributed querying.

## References

1. Shadbolt, N., Glaser, H., Harith, A., Carr, L., Chapman, S., Ciravegna, F., Dingli, A., Gibbins, N., Harris, S., Schraefel, M.C.: CS AKTiveSpace: Building a Semantic Web Application. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 417–432. Springer, Heidelberg (2004)
2. Lei, Y., Uren, V.S., Motta, E.: SemSearch: a search engine for the semantic web. In: Staab, S., Svátek, V. (eds.) EKAW 2006. LNCS (LNAI), vol. 4248, pp. 238–245. Springer, Heidelberg (2006)
3. Shirky, C.: The Semantic Web, Syllogism, and Worldview [online] (February 15, 2007) (2001), [http://www.shirky.com/writings/semantic\\_syllogism.html](http://www.shirky.com/writings/semantic_syllogism.html)
4. Fallows, D.: Search Engine Users, PEW Internet and American Life Project [online] (February 16, 2007) (2005), [http://www.pewinternet.org/pdfs/PIP\\_Searchengine\\_users.pdf](http://www.pewinternet.org/pdfs/PIP_Searchengine_users.pdf)
5. Shafiq, S.M., Rather, R.A.: Precision and Recall of Five Search Engines for Retrieval of Scholarly Information in the Field of Biotechnology. *Webology*, 2 (2), [online] (January 20, 2006) (2005), <http://www.webology.ir/2005/v2n2/a12.html>
6. Ding, L.: Enhancing Semantic Web Data Access, PhD Thesis, University of Maryland [online] (2006), <http://ebiquity.umbc.edu/paper/html/id/317/Enhancing-Semantic-Web-Data-Access>
7. Finin, T., Ding, L., Pan, R., Joshi, A., Kolari, P., Java, A., Peng, Y.: Swoogle: Searching for knowledge on the Semantic Web. In: AAAI 2005 (intelligent systems demo) (July 2005)
8. Guha, R., McCool, R.: TAP: A Semantic Web Platform. *Computer Networks* 42(5), 557–577 (2003)
9. Millard, I., Jaffri, A., Glaser, H., Rodriguez, B.: Using a Semantic MediaWiki to Interact with a Knowledge Based Infrastructure (Poster). In: Staab, S., Svátek, V. (eds.) EKAW 2006. LNCS (LNAI), vol. 4248, Springer, Heidelberg (2006)

# Ontology-Driven Management of Semantic Spaces

Reto Krummenacher

Digital Enterprise Research Institute, University of Innsbruck, Austria  
reto.krummenacher@deri.at

**Abstract.** Recent work in the field of middleware technology proposes semantic spaces as a tool for coping with the *scalability*, *heterogeneity* and *dynamism* issues of large scale distributed environments. Reflective middleware moreover offers answers to the needs for adaptivity and self-determination of systems. Based on experiences with traditional middleware we argue that ontology-driven management is a major advancement for semantic spaces and provides the fundamental means for reflection. In this research we look at ontology-based metadata to turn semantic spaces into reflective middleware [1].

## 1 Introduction

With the transformation of today's Web to a mobile and even ubiquitous web of interactive computers and small physical devices the coordination of large numbers of autonomous nodes gets particularly challenging, and further *scalability*, *heterogeneity* and *dynamism* issues arise [2]. In consequence middleware solutions are required that can adapt to dynamic changes in application requirements and environmental conditions and customize its service to various end user devices. Reflective middleware is considered to provide essential answers in this respect [1,2,8] [3].

Self-representation – an explicit representation of the internal structure of the implementation that the middleware maintains and manipulates – is important in order to support reflection in form of inspection and adaptation [1]. A system is reflective when it is able to manipulate and reason about itself [8]. A critical concept in this respect, in particular in absence of central control, is the concept of metadata. Metadata is data about the properties, capabilities, and requirements of system elements to enable their coordination [6]. It is thus essential to develop metadata infrastructures for the self-representation of the space middleware.

Our proposed solution follows a recent trend in the field of middleware for large scale open systems: *semantic spaces*. Spaces are a powerful concept for the coordination of autonomous nodes. Instead of explicitly exchanging messages or performing remote procedure calls, communication is done by reading and writing distributed data structures in a shared space. Semantic space middleware is then particularly tailored to the Semantic Web and Semantic Web service environments. Consequently, ontology-based modeling of metadata becomes the natural choice. Moreover, ontologies provide the grounds for formal reasoning about the middleware implementation - a prerequisite for reflection.

<sup>1</sup> This work is supported by the projects TSC (tsc.deri.at) and TripCom (www.tripcom.org).

<sup>2</sup> cf. W3C Ubiquitous Web Domain (www.w3.org/UbiWeb/).

<sup>3</sup> cf. Workshops on Adaptive and Reflective Middleware (www.ics.uci.edu/~arm06/).

## 2 Research Problem

Different semantic space proposals (Section 3) brought the interaction primitives and data models to maturity. However, the issues of network dynamics and scalability (use at Web scale) are barely addressed, although they are crucial to any installation. In other words, semantic spaces lack so far solutions to the increasing complexity of middleware's non-functional properties: decentralization, availability, reliability, fault-tolerance, scalability, and security. Moreover, it is important to take the use and application context into account in order to tailor the delivered service to the current needs [6]. The application of ontologies that describe the published data, the spaces and their interrelationships and characteristics is expected to be an effective instrument in this respect. Ontology-based metadata is thus essential for the management of the middleware and provides the formal grounds for reasoning, and hence reflection. This leads us directly to the main questions of our research.

**Main Question:** *what does ontology-based metadata have to model and how in order to turn semantic spaces into reflective and self-adaptive middleware for large scale open systems?*

**Problem 1:** *how to ontologize the space middleware and the data in order to provide reflective management of core non-functional properties, in particular distribution?*

**Problem 2:** *what requirements result from ontology-driven adaptation algorithms and which additional metadata modules are needed?*

**Problem 3:** *how can metadata be acquired and provided to participating nodes for adaptation of their routines and to improve the data distribution in particular?*

Improving the data distribution does not only decrease the discovery overhead, but influences the availability and scalability of the middleware (distribution by reasoning about the network and middleware) and the efficiency and quality of retrieval routines (distribution by reasoning about the usage and needs).

## 3 State of the Art and Related Activities

Within this paper we look at related activities in two domains: (1) the emerging field of semantic space middleware and their approaches to ontology-driven management (reflection is to our knowledge not addressed so far at all), and (2) the use of ontologies and metadata for the management of middleware in general.

**Semantic spaces:** First ideas for triplespace computing (a particular approach to semantic spaces) have been proposed in [3], as middleware for Semantic Web services. This and subsequent work mostly neglected the use of ontologies for the management tasks. As example, the TSC project [4] only knows limited meta-information about the published data. It fails however to define the way the space is structured and behaves; thus the middleware remains a black box without self-representation.

Semantic Web Spaces [11] has conceived a generic lightweight coordination middleware for sharing and exchanging semantic data on the Web. This project includes a brief outline of an ontology-based metadata model, however without solid investigation of more comprehensive application areas and use cases.

As a joint successor of TSC and Semantic Web Spaces the TripCom project was established [14]. TripCom spends significant effort on ontology and rule-driven management of spaces. Rules model the adaption strategies and decisions for the optimization of triplespaces. The work presented in this paper contributes to TripCom.

**Ontologies for middleware management:** Over the past years semantic technologies and tools have gained maturity and its applicability for the management of middleware [12] and distributed information systems [6] is acknowledged.

Based on the argument that most available ontologies for middleware suffer from conceptual ambiguity, poor axiomatization, loose design, and a too narrow scope, [12] developed a management ontology as extension to DOLCE [5], a foundational ontology. While [12] targets software components and (Web) services and thus has a different domain, some of the ideas and decisions provide a fruitful basis for our space management ontologies and self-representation strategies.

There are other relevant vocabularies to describe people and agents (FOAF), online communities (SIOC) or documents (Dublin Core) [4]. The Ontology Metadata Vocabulary [7] could be used to describe the schemes and topics a space adheres to, which is relevant for the discovery and clustering of semantically adjacent nodes. Such small and simple vocabularies are expected to provide many of the ground terms of our management ontologies, also because we emphasize on the importance of ontology reuse.

Furthermore, ontologies have gained momentum for modeling context [10]. Information about devices, network status and user intentions are important for adaptation. One early example of ours in this respect is the modeling of context rules with ASC [9,15]. ASC describes contextual data by taking into account quality of information, provenance, validity and satisfiability of information.

## 4 Research Methodology

The evaluation of the needs for metadata and the development of the formal models is an important part of this work. Requirements are deduced from our experience with semantic space middleware. First ideas for a semantic space ontology were outlined as part of TripCom [14]. Additionally we will consider the Semantic Web specific requirements outlined in [12], amongst others: awareness for Semantic Web languages, ontology modularization, interoperability, and verification. In [10] we compiled a further set of evaluation criteria dedicated to context modeling ontologies: traceability, comparability, logging and quality of data, and scalability of the ontology. Similar criteria form the basis for the development and theoretical evaluation of our models.

The metadata is first of all used to cluster data and to formally describe the decision rules that lead to the choice of the appropriate clustering algorithm. Distribution through semantic clustering provides the chosen 'proof-of-concept'. By applying the developed ontologies to a semantic space middleware implementation, we can evaluate our solutions against an existing infrastructure [5].

---

<sup>4</sup> FOAF: [www.foaf-project.org](http://www.foaf-project.org), SIOC: [www.sioc-project.org](http://www.sioc-project.org), and DC: [www.dublincore.org](http://www.dublincore.org)

<sup>5</sup> A demonstrator is available from TSC; a first prototype from TripCom.



The distribution algorithms for structuring the virtual spaces and for clustering the published data will be evaluated qualitatively. First, we compare the management effort and retrieval quality with a non-reflective implementation, by help of use cases. Moreover, we measure the quality of the clusters (e.g. divergence of members, size of clusters and difference of content), the management overhead of the algorithms, and the number of hops needed to resolve user requests - a more P2P-based criterion.

## 5 Expected Contribution

As outcome of this research we expect a metadata infrastructure consisting of an integrated set of ontologies. The ontologies will be developed alongside the criteria shortly mentioned in the previous section and tailored to the management processes present in semantic space middleware; in particular to the ones that profit from reflection and self-adaptation with a primary focus on distribution. As [13] pointed out it is advisable to approach the Semantic Web by small and simple ontologies that are easier adopted and reused in the large. Consequently we will develop a collection of well-integrated, but distinct ontologies for different aspects of our system: the representation of spaces and data, users and their context, descriptions of kernels, their functionality and also their connection to other kernels. We like to highlight one particular issue that is mostly neglected in distributed information systems [6] and that is, as pointed out before, essential for the installation of reflection:

*In many scenarios the relevance of information depends on the usage purpose and its interpretation on the context at hand. This becomes especially eminent in distributed settings where data producers and consumers have different backgrounds. Context-awareness and personalization are indispensable instruments to increase the sensitivity of middleware.*

In summary we develop ontologies for the management of semantic space middleware implementations for large-scale, open and dynamic systems with the goal to enable reflection. Therewith we expect to significantly contribute to the success of semantic space technology: our ontology-driven management procedures go beyond the ones of Trip-Com and other related projects by being adaptive with respect to changing user needs and environmental conditions. The potential adaptations to the structure and behavior of the middleware are encoded in decision rules and the management processes are thus at least partly automated. Improvement is in particular expected in settings where semantic spaces are applied to applications in large mobile or ubiquitous web settings.

## 6 Conclusion

While the conceptual work for semantic spaces becomes mature and the data models and interaction primitives are well defined, the given approaches to ontology-driven management are very much in its infancy. There are ontological vocabularies released for the modeling of spaces and the description of data. However, these ontologies do clearly neither address the requirements of distributed and dynamic systems nor the

additional needs of reflection and adaptation algorithms: "an explicit representation of the internal structure of the middleware implementation that the middleware maintains and manipulates" [8].

Our next steps include further analysis of the management processes and metadata vocabularies of existing distributed information systems and reflective middleware solutions. A number of reflective middleware implementations are discussed in [2]. Based thereon we will develop the set of ontologies necessary for our purposes and apply it to an existing semantic space implementation. This provides a solid proof of concept and allows showcasing our approach with respect to the non-functional property distribution: considering in particular the management overhead, availability of nodes and scalability. The findings with respect to data distribution are then easily generalized, as the use of reflection is known to improve other non-functional aspects too.

## References

1. Coulson, G.: What is Reflective Middleware? *IEEE Distr. System Online* 2(8) (2001)
2. Curry, E.: Adaptive and Reflective Middleware. In: Mahmoud, Q. (ed.) *Middleware for Communications*, Wiley, Chichester (2004)
3. Fensel, D.: Triple-Space Computing: Semantic Web Services Based on Persistent Publication of Information. In: *IFIP Int'l. Conf. on Intelligence in Communication Systems*, pp. 43–53 (November 2004)
4. Fensel, D., Krummenacher, R., Shafiq, O., Kuehn, E., Riemer, J., Ding, Y., Draxler, B.: TSC - Triple Space Computing. *e&i Elektrotechnik und Informationstechnik* 124(1/2) (2007)
5. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening Ontologies with DOLCE. In: *13th Int'l. Conf. on Knowledge Engineering and Management*, pp. 166–181 (October 2002)
6. Haase, P.: *Semantic Technologies for Distributed Information Systems*. PhD thesis, Universität Karlsruhe (2006)
7. Hartmann, J., Palma, R., Sure, Y., Suarez-Figueroa, M.C., Haase, P.: OMV - Ontology Metadata Vocabulary. In: *Workshop on Ontology Patterns for the Semantic Web* (November 2005)
8. Kon, F., Costa, F., Blair, G., Campbell, R.H.: The Case for Reflective Middleware. *Communications of the ACM* 45(6), 33–38 (2002)
9. Krummenacher, R., Lausen, H., Strang, T.: On the Modeling of Context-Rules with WSML. In: *Workshop Contexts&Ontologies: Theory, Practice and Applications* (August 2006)
10. Krummenacher, R., Lausen, H., Strang, T.: Analyzing the Modeling of Context with Ontologies. In: *Int'l. Workshop on Context-Awareness for Self-Managing Systems* (May 2007)
11. Nixon, L., Simperl, E., Antonenko, O., Tolksdorf, R.: Towards Semantic Tuplespace Computing: The Semantic Web Spaces System. In: *22nd ACM Symp. on Applied Computing*, ACM Press, New York (March 2007)
12. Oberle, D.: *Semantic Management of Middleware*. Springer, Heidelberg (2006)
13. Rousset, M.-C.: Small Can Be Beautiful in the Semantic Web. In: *3rd Int'l. Semantic Web Conf.* (November 2004)
14. Simperl, E., Krummenacher, R., Nixon, L.: A Coordination Model for Triplespace Computing. In: *9th Int'l. Conference on Coordination Models and Languages* (June 2007)
15. Strang, T., Linnhoff-Popien, C., Frank, K.: CoOL: A Context Ontology Language to Enable Contextual Interoperability. In: *4th Int'l. Conf. on Distributed Applications and Interoperable Systems* (November 2003)

# Ontology Mapping: An Information Retrieval and Interactive Activation Network Based Approach

Ming Mao

School of Information Sciences  
University of Pittsburgh  
mingmao@mail.sis.pitt.edu

**Abstract.** Ontology mapping is to find semantic correspondences between similar elements of different ontologies. It is critical to achieve semantic interoperability in the WWW. This paper proposes a new generic and scalable ontology mapping approach based on propagation theory, information retrieval technique and artificial intelligence model. The approach utilizes both linguistic and structural information, measures the similarity of different elements of ontologies in a vector space model, and deals with constraints using the interactive activation network. The results of pilot study, the PRIOR, are promising and scalable.

**Keywords:** ontology mapping, profile propagation, information retrieval, interactive activation network, PRIOR.

## 1 Introduction

The World Wide Web (WWW) now is widely used as a universal medium for information exchange. Semantic interoperability among different information systems in the WWW is limited due to information heterogeneity, and the non semantic nature of HTML and URLs. Ontologies have been suggested as a way to solve the problem of information heterogeneity by providing formal and explicit definitions of data. They may also allow for reasoning over related concepts. Given that no universal ontology exists for the WWW, work has focused on finding semantic correspondences between similar elements of different ontologies, i.e., *ontology mapping*. Automatic ontology mapping is important to various practical applications such as the emerging Semantic Web [3], information transformation and data integration [2], query processing across disparate sources [7], and many others [4].

Ontology mapping can be done either by hand or using automated tools. Manual mapping becomes impractical as the size and complexity of ontologies increases. Fully or semi-automated mapping approaches have been examined by several research studies, e.g., analyzing linguistic information of elements in ontologies [15], treating ontologies as structural graphs [12], applying heuristic rules to look for specific mapping patterns [8] and machine learning techniques [1]. More comprehensive surveys of ontology mapping approaches can be found in [9][14].

This paper proposes a new generic and scalable ontology mapping approach, shown in Fig. 1. The approach takes advantage of propagation theory, information

retrieval technique and artificial intelligence model to solve ontology mapping problem. It utilizes both linguistic and structural information of ontologies, measures the similarity of different elements of ontologies in a vector space model, and integrates interactive activation network to deal with constraints.

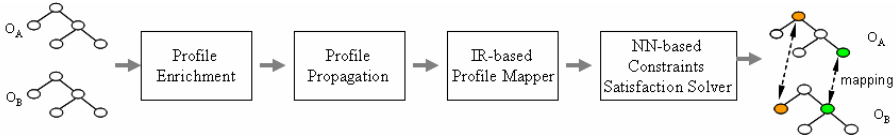


Fig. 1. The architecture of the proposed approach

## 2 The Proposed Approach

### 2.1 Profile Enrichment

Similar as the virtual document used in Falcon-AO system [15], the *profile* of a concept is a combination of linguistic information of a concept, i.e. the profile of a concept = its name + label + comment + property restriction + other descriptive information. The Profile Enrichment is a process that uses a profile to represent a concept in the ontology, and thus enrich its information. It is based on the observation that sometimes the information carried in the name of a concept is restricted, but other descriptive information like labels and comments may contain words that better convey the meaning of concepts. A sample profile of a concept “book” looks like  $Profile(book) = (book, book, book, monograph, collection, write, text)$ . Afterwards the *tf-idf* (term frequency–inverse document frequency) weight will be used to assign larger weight to the terms that have a high frequency in given document and a low frequency in the whole collection of documents. Here each profile is treated as a document and all profiles in two ontologies are treated as the collection of documents.

### 2.2 Profile Propagation

The Profile Propagation exploits the neighboring information of each concept. That is, the profile of ancestors and descendants are passed to that of the concept itself based on propagation theory [6]. The process of profile propagation can be represented using Equation 1, where  $C$  and  $C'$  denote two concepts in the ontologies,  $S$  denotes the set of all concepts in the ontologies,  $V_{C_{new}}$  denotes the new profile vector of the concept  $C$ ,  $V_{C'}$  denotes the profile vector of the concept  $C'$ , and  $w(C, C')$  is a function that assigns different weights to the neighbors of the concept following two principles: 1. The closer two concepts locate, the higher impact they have. 2. The impact from ancestors to descendants is higher than the impact from descendants to ancestors.

$$V_{C_{new}} = \sum_{C' \in S} w(C, C')V_{C'} \tag{1}$$

### 2.3 IR-Based Profile Mapper

The insight of the proposed approach is to treat ontology mapping problem as an information retrieval task. That is, if concepts in an ontology are seen as documents in a collection, finding correspondence between similar concepts in ontologies is just like to search the most relevant document in one collection given a document in another collection. Given a query and a set of documents, classical IR methods usually measure the similarity of a query and different documents, and then return the documents with top-ranked similarities as result. In the context of ontology mapping, such IR method can be applied as following: Given two to-be-mapped ontologies,  $O_A$  and  $O_B$ , all profiles of concepts in  $O_A$  are indexed first. Simultaneously queries based on the profile of each concept in  $O_B$  are generated. Then searches are executed in  $O_A$  using queries generated from  $O_B$  one by one. Afterwards the concepts in  $O_A$  with top-ranked similarities or above a predefined threshold are returned and stored. Now two ontologies are switched and the whole process is repeated. Finally two result sets are compared and the overlapped ones indicate possible mappings.

Cosine angle between two vectors of the documents is commonly used to measure their similarity. In the context of ontology mapping, the cosine similarity between two concepts  $C$  and  $C'$  can be measured using Equation 2, where  $V_C$  and  $V_{C'}$  are two vectors of the profile of concept  $C$  and  $C'$  respectively,  $n$  is the dimension of the profile vectors,  $V_i^C$  and  $V_i^{C'}$  are  $i$ th element in the profile vector of concept  $C$  and  $C'$  respectively,  $|V_C|$  and  $|V_{C'}|$  are the lengths of the two vectors respectively. The output of Profile Mapper is a concept-to-concept similarity matrix, where each element represents a similarity between two concepts. Note that such a similarity matrix might be very sparse due to the large size of ontologies and the low overlap between them.

$$Sim_{c,c'} = Sim(V_C, V_{C'}) = \frac{\overline{V_C} \cdot \overline{V_{C'}}}{|V_C| |V_{C'}|} = \frac{\sum_{i=1}^n (V_i^C * V_i^{C'})}{\sqrt{\sum_{i=1}^n (V_i^C)^2} \sqrt{\sum_{i=1}^n (V_i^{C'})^2}} \tag{2}$$

### 2.4 Interactive Activation Network Based Constraints Satisfaction Solver

Constraints satisfaction problem (CSP) [16] arises as an intriguing research problem in ontology mapping due to the characteristics of ontology itself and its representations. The hierarchical relations in RDFS, the axioms in OWL and the rules in SWRL result in different kinds of constraints. For example, "if concept A matches concept B, then the ancestor of A can not match the child of B in the taxonomy" and "two classes match if they have owl:sameAs or owl:equivalentClass relations". To improve the quality of ontology mapping, it is critical to find the best configuration that can satisfy such constraints as much as possible.

CSPs are typically solved by a form of search, e.g. backtracking, constraint propagation, and local search [16]. The interactive activation network is first proposed to solve CSPs in [13]. The network usually consists of a number of competitive nodes connected to each other. Each node represents a hypothesis. The connection between two nodes represents constraint between their hypotheses. Each connection is associated with a weight. For example, we have two hypotheses,  $H_A$  and  $H_B$ . If whenever  $H_A$  is true,  $H_B$  is usually true, then there is a positive connection from node

A to node B. Oppositely if  $H_A$  provides evidence against  $H_B$ , then there is a negative connection from node A to node B. The importance of the constraint is proportional to the strength (i.e. *weight*) of the connection representing that constraint. The state of a node is determined locally by the nodes adjacent to it and the weights connecting to it. The state of the network is the collection of states of all nodes. Entirely local computation can lead the network to converge to a global optimal state.

In the context of ontology mapping, a node in an interactive activation network represents a hypothesis that concept  $C_{1i}$  in ontology  $O_1$  can be mapped to concept  $C_{2j}$  in ontology  $O_2$ . The initial activation of the node is the similarity of  $(C_{1i}, C_{2j})$ . The activation of the node can be updated using the following simple rule, where  $a_i$  denotes the activation of *node i*, written as  $n_i$ ,  $net_i$  denotes the net input of the node.

$$a_i(t+1) = \begin{cases} a_i(t) + net_i(1 - a_i(t)), & net_i > 0 \\ a_i(t) + net_i a_i(t), & net_i < 0 \end{cases} \quad (3)$$

The  $net_i$  comes from three sources, i.e. its neighbors, its bias, and its external inputs, as defined in Equation 4, where  $w_{ij}$  denotes the connection weight between  $n_i$  and  $n_j$ ,  $a_j$  denotes the activation of node  $n_j$ ,  $bias_i$  denotes the bias of  $n_i$ , the  $istr$  and  $estr$  are constants that allow the relative contributions of the input from internal sources and external sources to be readily manipulated. Note that the connection matrix is not symmetric and the nodes may not connect to themselves, i.e.,  $w_{ij} \neq w_{ji}$ ,  $w_{ii} = 0$ .

$$net_i = istr \times \left( \sum_j w_{ij} a_j + bias_i \right) + estr \times (input_i) \quad (4)$$

Furthermore, the connections between nodes in the network represent constraints between hypotheses. For example, the constraint that “only 1-to-1 mapping is allowed” results in a negative connection between nodes  $(C_{1i}, C_{2j})$  and  $(C_{1i}, C_{2k})$ , where  $k \neq j$ . Moreover, “two concepts match if all their children match”, results in a positive connection between nodes  $(C_{1i}, C_{2j})$  and  $(C_{1k}, C_{2i})$ , where  $C_{1k}$  and  $C_{2i}$  are the children of  $C_{1i}$  and  $C_{2j}$  respectively. Finally, the complexity of the connections may be very large because of complex constraints.

### 3 Pilot Study

The proposed approach has been partially implemented in the PRIOR [10][11], an ontology mapping tool based on propagation theory and information retrieval techniques. The results from OAEI ontology matching campaign 2006<sup>1</sup> show the PRIOR is promising and competitive to all other approaches in different tracks, namely benchmark, web directory, food, and anatomy [5].

### 4 Future Work

The implementation of the interactive activation network to satisfy constraints in ontology mapping is our major future work. Other work includes integrating auxiliary information such as WordNet to distinguish synonyms.

<sup>1</sup> <http://oaei.ontologymatching.org/2006/results/>

## References

1. Doan, A., Madhavan, J., et al.: Learning to Match Ontologies on the Semantic Web. *VLDB Journal* 12(4), 303–319 (2003)
2. Dou, D., McDermott, D., et al.: Ontology Translation on the Semantic Web. *Journal on Data Semantics (JoDS)* II, 35–57
3. Ehrig, M.: *Ontology Alignment: Bridging the Semantic Gap (Semantic Web and Beyond)*. Springer (2006) ISBN-038732805X
4. Euzenat, J., Bach, T., et al.: State of the art on ontology alignment, *Knowledge web NoE* (2004)
5. Euzenat, J., et al.: Results of the Ontology Alignment Evaluation Initiative 2006. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006. LNCS*, vol. 4273, Springer, Heidelberg (2006)
6. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient belief propagation for early vision. *International Journal of Computer Vision* 70(1) (2006)
7. Gasevic, D., Hatala, M.: Ontology mappings to improve learning resource search. *British Journal of Educational Technology* (2005)
8. Hovy, E.: Combining and standardizing large-scale, practical ontologies for machine translation and other uses. In: *Proceedings of the 1st International Conference on Language Resources and Evaluation (LREC)*, Granada, Spain (1998)
9. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. *The Knowledge Engineering Review* 18(1), 1–31 (2003)
10. Mao, M., Peng, Y.: PRIOR System: Results for OAEI 2006. In: *Proceedings of ISWC 2006 Ontology Matching Workshop*, Atlanta, GA (2006)
11. Mao, M., Peng, Y., Spring, M.: A Profile Propagation and Information Retrieval Based Ontology Mapping Approach. In: *Proceedings of SKG 2007* (2007)
12. Melnik, S., Garcia-Molina, H., et al.: Similarity flooding: a versatile graph matching algorithm and its application to schema matching. In: *ICDE 2002. Proc. 18th International Conference on Data Engineering* (2002)
13. McClelland, J.L., Rumelhart, D.E.: *Explorations in Parallel Distributed Processing: A Handbook of Models, Programs, and Exercises*. MIT Press, Cambridge (1988)
14. Noy, N.: Semantic Integration: A Survey of Ontology-Based Approaches. *SIGMOD Record* 33(4), 65–70 (2004)
15. Qu, Y., Hu, W., Cheng, G.: Constructing virtual documents for ontology matching. In: *Proceedings of the 15th International Conference on World Wide Web* (2006)
16. Tsang, E.: *Foundations of Constraint Satisfaction*: Academic Press (1993)

# Probabilistic Information Integration and Retrieval in the Semantic Web

Livia Predoiu

Institute of Computer Science, University of Mannheim,  
A5,6, 68159 Mannheim, Germany  
livia@informatik.uni-mannheim.de

## 1 Research Context

The Semantic Web (SW) has been envisioned to enable software tools or Web Services, respectively, to process information provided on the Web automatically. For this purpose, languages for representing the semantics of data by means of ontologies have been proposed such as RDF(S) and OWL. While the semantics of RDF(S) requires a non-standard model-theory that goes beyond first order logics, OWL is intended to model subsets of first order logics. OWL consists of three variants that are layered on each other. The less expressive variants OWL-Light and OWL-DL correspond to the Description Logics *SHIF*(D) and *SHOIN*(D) [1], respectively, and thus to subsets of First Order Logics [2].

While RDF and OWL are W3C recommendations and hence a kind of standard, a lot of proposals emerged recently for representing Logic Programming (LP) variants on the Web. Such proposals are e.g. SWRL [1] and WRL [2]. Furthermore, a working group exists at the W3C for defining a rule interchange format [3]. Therefore, it can be expected that rule languages will play an important role in the SW. The Description Logics (DL) and the LP paradigm are orthogonal having just a small subset in common [3] and a comparison reveals a balanced amount of advantages and disadvantages of one compared to the other e.g. concerning the efficiency of certain reasoning tasks [4].

The SW will consist of independent peers each providing information that describes overlapping domains by different ontologies or logic programs specified in different knowledge representation languages of the DL and the LP paradigm. In order to enable intelligent software tools to utilize the information represented by these peers in a coherent manner, the ontologies and logic programs need to be aligned by means of mappings. An number of approaches for learning mappings between ontologies exist already [5]. Most of them detect very simple matchings and can be used for learning mappings between ontologies and logic programs as well.

---

<sup>1</sup> <http://www.w3.org/Submission/SWRL/>

<sup>2</sup> <http://www.w3.org/Submission/WRL/>

<sup>3</sup> <http://www.w3.org/2005/rules/wg.html>



## 2 Problem Definition

Currently, mappings are mainly used deterministically. I.e. although in general, automatically learned mappings are closely connected to a confidence which expresses some kind of belief of the matcher that each mapping holds, the mappings are considered to be either true or false depending on some threshold level.

If the mappings that are found by a probabilistic matching approach, there is evidence that keeping the probabilities and using probabilistic inference for answering queries is likely to change and improve the outcome compared to a deterministic usage of mappings. This holds especially if we consider real world SW mapping scenarios where several ontologies are connected by mappings in a catenarian way (with mapping composition). Using mappings that have a probability lower than a threshold is likely to influence the results in such a way that previously ruled out results get a high probability. Also mappings that are found by non-probabilistic approaches are in general found with a number that expresses the confidence of its validity. This number can be interpreted probabilistically, e.g. by means of stating error probabilities. Thus, these findings are not limited to probabilistic mapping approaches.

This thesis aims for the development of a framework that enables a Semantic Web consisting of DL and LP knowledge bases being connected by mappings that are attached each by a probability that expresses the certainty of the validity of it. By means of such a framework and a probabilistic extension of a language that integrates LP and DL variants probabilistic information retrieval for the Web can be implemented.

## 3 Expected Contribution

For the development of a probabilistic information integration framework for the SW that integrates probabilistic mappings with probabilistic and deterministic ontologies and logic programs being mapped one to another by the mappings, a probabilistic SW Language and reasoning algorithms are required.

Hence, the expected contribution is

1. a probabilistic extension of a language that is capable of integrating the DL variants underlying OWL (or OWL 1.1 or one or more of its tractable fragments [6]) and variants of the LP paradigm.
2. distributed reasoning algorithms for this language that consider the inherently distributed nature of the information sources in the SW.
3. tools that implement the language and the algorithms.
4. a framework that
  - integrates probabilistic and deterministic knowledge bases provided by peers in the SW by utilizing the language and reasoning algorithms mentioned above
  - provides facilities for (distributed) probabilistic information retrieval in order to enable efficient retrieval of the probabilistically integrated information

The advantages of such a framework are that it will be possible to

- express probabilistic knowledge in the SW
- integrate probabilistic and deterministic knowledge in the SW
- integrate DL and LP knowledge bases
- use the confidence of mappings (and thus the heuristics that matchers are using for discovering mappings) and improve the preciseness of information integration especially in settings that involve mapping composition.
- incorporate means to integrate preference between and trust in data sources and/or matchers
- use conflicting mappings to some extent
- perform information retrieval over distributed DL and LP knowledge bases

## 4 Related Work

A probabilistic framework for Information Integration and Retrieval on the SW does not exist yet. However, in [7] suggestions are made for such a framework. But the only substantial contribution to such a framework is a tool for learning mappings consisting of simple probabilistic Datalog (pDatalog) rules [8] between OWL ontologies. Ideas on how to reason with the ontologies and rules are missing.

There exist a couple of probabilistic extensions of SW languages that provide a tight integration on the formal level between a SW Language or a subset of it and a probabilistic model. Such a tight integration is needed for the framework that is intended to be developed in this thesis. Besides probabilistic extensions that just consider RDF or OWL, the following extensions are related to integrating DL and LP. pOWL Lite<sup>-</sup> and its extension with equality, pOWL Lite<sup>EQ</sup>, [9] are probabilistic extensions of a subset DLPs [10] basing on pDatalog. The resulting formalism is a subset of pDatalog. Information integration is not considered in the context of pOWL Lite<sup>-</sup>/<sup>EQ</sup>. The languages have been proposed solely for the purpose of expressing probabilistic OWL statements. However, as these languages are basing on DLPs which is a KR formalism lying in the common subset of DL and LP, probabilistic Information Integration can be realized with them. As oMap [7] discovers mappings consisting of simple pDatalog rules, an information integration setting is conceivable that combines the pOWL Lite<sup>-</sup>/<sup>EQ</sup> languages with oMap.

Probabilistic Description Logic Programs is a KR formalism that integrates the DLs underlying OWL-Lite and OWL-DL with stratified Datalog [11] and disjunctive Datalog with Negation [12]. Its probabilistic model is based on Independent Choice Logic [13]. However, the interaction between the DL part and the LP part is limited. A less restricted probabilistic integration of the DLs underlying OWL-Lite and OWL-DL with disjunctive Datalog with Negation is expressed by tightly integrated probabilistic description Logic Programs (tiPDL) [14]. Currently, there are no reasoning tools available for these formalisms and reasoning in the general formalisms is very inefficient.

## 5 Approach and Methodology

**Problem Definition:** A framework for probabilistic information integration and retrieval for the SW which can be expected to consist of DL and LP knowledge bases does not exist yet, but is needed in order to make use of the uncertainty that is inherently present in each mapping.

### *Identification of Requirements*

The **language** for the framework has the following requirements on its *expressivity*: it needs to be capable of integrating the DL and LP variants that are important in the SW. It also requires a *tight integration with a probabilistic model*. In the scope of this thesis, DLPs have been extended with probabilities obeying the probabilistic model of Bayesian Logic Programs (BLPs) [15], yielding Bayesian DLPs. The resulting formalism is called Bayesian DLPs (BDLPs). For BDLPs, also a way to integrate probabilistic and deterministic ontologies and logic programs lying in the DLP fragment with probabilistic mapping rules has been proposed in [4]. For reasoning, usage of the existing BLP reasoner Balios has been proposed as BLPs are a superset of BDLPs. While the integration of DLPs and BLPs is very tight and thus sufficient for our purposes, the expressivity of the DLP fragment is too limited. Currently, I am investigating the tiPDL [14] KR formalism mentioned above and subsets of it. Subsets that use the subset of ICL that lies in Bayesian Networks seem to be very promising for the purpose of the framework.

The requirements for **reasoning algorithms** in this framework are the *consideration of the inherent distribution* of the data over several peers. I.e. the reasoning algorithms to be developed should be able to select peers that are relevant to a specific query, merge the results of distributed reasoning resources and thus take advantage of parallel reasoning. Due to the high expressivity of the language, reasoning in general can be expected to be very inefficient. Therefore, the reasoning algorithms to be developed in this thesis will be approximate reasoning algorithms due to the natural requirement of *efficiency*.

The requirements for an **implementation of the framework** is the *creation of an infrastructure* for the framework. For this purpose, existing tools are intended to be reused. Thus, appropriate tools need to be evaluated in order to enable the choice of the ones that are best suited for the framework. Clearly, another requirement for the implementation is efficiency. Furthermore, methods that assess preference between and trust into data sources and matchers are needed.

**Design:** The design of the framework needs to enable fast and efficient access of the data sources to be integrated. It will be modular. However, its' specification depends on the results of the tool analysis for the implementation of the framework.

**Evaluation:** For the evaluation of the framework, ontologies (and mappings) from the Ontology Alignment Evaluation Initiative<sup>4</sup> can be used. A set of logic programs needs to be collected as well and a couple of mapping tools can be used for discovering a set of mappings between the logic programs and the logic

<sup>4</sup> <http://oaei.ontologymatching.org/>

programs and the ontologies. In order to show that the probabilistic approach of this thesis is appropriate for resolving conflicting mappings, it can be evaluated against approaches that resolve conflicting mappings by repairing, e.g. [16]. It has also to be shown whether the usage of the confidence and probability values of the matchers improves the results of information integration in a setting that involves several ontologies and mapping composition. The information retrieval facilities will be compared with other current information retrieval tools.

## References

1. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From *SHIQ* and RDF to OWL: the making of a Web Ontology Language. *Journal of Web Semantics* (2003)
2. Borgida, A.: On the relationship between description logic and predicate logic. *Artificial Intelligence* 82(1-2) (2006)
3. Krötzsch, M., Hitzler, P., Vrandečić, D., Sintek, M.: How to reason with OWL in a logic programming system. In: *Proc. of the Conference on Rules and Rule Markup Languages for the Semantic Web* (2006)
4. Predoiu, L.: Information integration with bayesian description logic programs. In: *Proc. of 3rd IIWeb Workshop for Information Integration on the Web* (2006)
5. Euzenat, J., Shvaiko, P.: *Ontology matching*. Springer, Heidelberg (2007)
6. Grau, B.C., Calvanese, D., Giacomo, G.D., Horrocks, I., Lutz, C., Motik, B., Parsia, B., Patel-Schneider, P.F.: *OWL 1.1 Web Ontology Language Tractable Fragments* (2006), URL: <http://www.w3.org/Submission/owl11-tractable/>
7. Straccia, U., Troncy, R.: Towards Distributed Information Retrieval in the Semantic Web: Query Reformulation Using the oMAP Framework. In: *Proc. of the 3rd European Semantic Web Conference* (2006)
8. Fuhr, N.: Probabilistic Datalog: Implementing Logical Information Retrieval for Advanced Applications. *Journal of the American Society for Information Science* 51(2) (2000)
9. Nottelmann, H., Fuhr, N.: Adding Probabilities and Rules to OWL Lite Subsets based on Probabilistic Datalog. *Uncertainty, Fuzziness and Knowledge-Based Systems* 14(1) (2006)
10. Grosz, B.N., Horrocks, I., Volz, R., Decker, S.: Description Logic Programs: combining logic programs with description logic. In: *Proc. of the 12th international conference on World Wide Web* (2003)
11. Lukasiewicz, T.: Stratified Probabilistic Description Logic Programs. In: *Proc. of the ISWC Workshop on Uncertainty Reasoning for the Semantic Web* (2005)
12. Lukasiewicz, T.: Probabilistic Description Logic Programs. In: *Proc. of the conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty* (2005)
13. Poole, D.: The independent choice logic for modelling multiple agents under uncertainty. *Artificial Intelligence* 94(1-2) (1997)
14. Cali, A., Lukasiewicz, T.: *Tightly Integrated Probabilistic Description Logic Programs*. Technical report, Institut für Informationssysteme. TU Wien (2007)
15. Kersting, K., Raedt, L.D.: *Bayesian Logic Programs*. Technical report, Albert-Ludwigs University, Freiburg (2001)
16. Meilicke, C., Stuckenschmidt, H., Tamilin, A.: Repairing Ontology Mappings. In: *Proc. of AAAI* (2007)

# OWL-DL as a Power Tool to Model Negotiation Mechanisms with Incomplete Information

Azzurra Ragone

SisInfLab, Politecnico di Bari, I-70125, Bari, Italy  
a.ragone@poliba.it

**Abstract.** We propose a logic-based framework to model negotiations among agents in scenarios with fully incomplete information. In particular, we address multi-issue bilateral negotiations, with issues are expressed and related to each other via an OWL ontology. In particular we use OWL-DL sub-language and exploit its formal semantics, based on  $\mathcal{SHOIN}(D)$ . Agents goals are expressed through OWL-DL as (complex) concept expressions, and the worth of goals as weights over concepts. Noteworthy, we adopt a very general setting with incomplete information by letting agents keep both goals and worths of goals as private information. Moreover we propose a protocol for agents to negotiate on and reach an agreement, exploiting inference services. Agents, according to the protocol, can pursue different sets of strategies —depending on their type— and we illustrate and motivate two possible strategies.

## 1 Introduction

Several recent research efforts have been focused on automated negotiation in various contexts, including e-marketplaces, resource allocation settings, online auctions, supply chain management and, in general terms, e-business processes. Negotiation mechanisms usually model resource and task allocation problems where issues to negotiate on are well established and defined in advance, *e.g.*, online auctions. Many other negotiation mechanisms instead model e-marketplaces of undifferentiated products (commodities) where the only issues to negotiate on are price or quantity. Nevertheless there a number of frameworks where agents have to reach an agreement on a product (car, house, etc.) or service (travel booking, wedding service, etc.) that can be described by many issues amenable to negotiation, and such issues may well be not necessarily all established in advance. Moreover Buyer (Requester) and Seller (Provider) may be not necessarily interested in the same set of issues and may have different preferences on bundles of interrelated issues. Obviously, if issues are not fixed there is the problem to express what agents “want” or “prefer”. For instance, considering an housing scenario, how to express a request for *a double room within an apartment provided with a washing machine for a period of six month* or, conversely an offer for *single or double room in an apartment where pets are not allowed and the bills are included in the lease*? Is there any negotiation space? Can an agreement —in an automated way— be reached? Trying to answer to this and other questions, we present a framework for multi-issue bilateral negotiation, with issues expressed and related to each other exploiting OWL-DL formalism. Agents goals are expressed through (complex) concepts, and the worth of goals as

weights over concepts. We consider a very general setting with incomplete information, *i.e.*, agents keep both goals and worths of goals as private information. We introduce a negotiation protocol and possible agents' strategies for such a setting, exploiting inference services—satisfiability and subsumption—available as services in DL reasoners. In a nutshell, satisfiability can be useful to catch inconsistency between agent's goals w.r.t. the ontology  $\mathcal{T}$ , *i.e.*, inconsistent goals cannot be in the same agreement, (*e.g.*, agents cannot agree on  $A$  and  $B$  at the same time if in  $\mathcal{T}$   $A$  is defined as disjoint from  $B$ ). Through subsumption one can discover if an agent's goal is implied by a goal of its opponent, even if this fact does not immediately appear at the syntactic level.

## 2 Scenario

We set the stage of our approach in a generic framework with incomplete information: we assume agents do not reveal their goals either to the opponent or to a mediator, but they keep as private information both goals and their worths. Actually, the difficulty to model scenarios with incomplete information is due to the fact that the agent cannot be sure how the opponent will evaluate its offers, and therefore it may be unable to negotiate to the best of its capacity [4]. Usually, to overcome such drawback, a preliminary step is added to strategic negotiations where agents reveal some (or all) private information. Obviously, the revelation mechanism has to ensure agents truthfully report their private information and punish liars [12]. Yet it is not always possible to design truthful revelation mechanisms, since they depend on the particular scenario taken into account (see [4, p.64] for an extensive discussion). When negotiation involves organizations, *e.g.*, companies, revealing information may conflict with company's interests and assessing the truthfulness of the agents' declarations can be very hard or impossible. An advantage of our proposal is that having considered a "worst case" scenario, we can then—in a principled way—study simplified negotiation scenarios.

The protocol we propose here is able to deal with such incomplete information without forcing agents to reveal neither their goals nor their utility functions, so it suits all scenarios where agents are not willing to reveal private information or when it is hard to design a truthful revelation mechanism. In this paper we refer, as an example, to a scenario where agents negotiate for a lease. Let us suppose that Amy is a student having to spend a period of about six months abroad as visiting student, so she needs to find a room in an apartment and she has to negotiate with the landlord about the lease terms. She is searching for a *bedroom*, which is not a *multiple room*, in an apartment provided with *washing machine*; actually she would prefer a *single room*, but this is not a mandatory constraint. She is looking for an apartment where *cats are not allowed*, because of her allergy, and where *no security deposit* is required and *bills are included*; finally she would prefer to have a *parking* (street parking or own parking) near the apartment. The following goals can be formalized as in Fig. 1. On the other hand also the landlord—Oscar—has some goals, that can be formalized as in Fig. 2. Relations among issues are expressed in the Ontology reported in Fig. 3. Obviously, the one described above is not the only feasible scenario to apply the approach proposed, since the negotiation framework we propose is very general and can be applied to several negotiation scenarios where resource descriptions can be modeled through KR languages.

$A_1 = \exists \text{hasBedroom.}(\geq 1 \text{ hasPlaces}) \sqcap \forall \text{hasBedroom.} \neg \text{MultipleRoom}$   
 $A_2 = \exists \text{appliances\_included.} \text{Washing\_machine}$   
 $A_3 = \exists \text{hasBedroom.}(= 1 \text{ hasPlaces})$   
 $A_4 = \forall \text{pets\_allowed.} \neg \text{Cats}$   
 $A_5 = \text{lease\_period} = 6$   
 $A_6 = \text{bills\_included.} \text{Gas}$   
 $A_7 = \forall \text{lease\_requirements.} \neg \text{SecurityDeposit}$   
 $A_8 = \exists \text{hasParking}$

**Fig. 1.** Amy's goals formalized using the logic  $SHOIN(D)$

$O_1 = \exists \text{hasBedroom.} \text{DoubleRoom}$   
 $O_2 = \exists \text{hasSharedRoom.} \text{Laundry}$   
 $O_3 = \neg \exists \text{pets\_allowed}$   
 $O_4 = \text{lease\_period\_negotiable}$   
 $O_5 = \forall \text{bills\_included.} \perp$   
 $O_6 = \exists \text{lease\_requirements.} \text{SecurityDeposit}$   
 $O_7 = \forall \text{hasParking.} \text{StreetParking} \sqcap \exists \text{hasParking}$

**Fig. 2.** Oscar's goals formalized using the logic  $SHOIN(D)$

$\mathcal{T}$

$\text{SingleRoom} \equiv \text{Bedroom} \sqcap (= 1 \text{ hasPlaces})$   
 $\text{DoubleRoom} \equiv \text{Bedroom} \sqcap (= 2 \text{ hasPlaces})$   
 $\text{MultipleRoom} \equiv \text{Bedroom} \sqcap (\geq 3 \text{ hasPlaces})$   
 $\text{Laundry} \sqsubseteq \text{Room} \sqcap \exists \text{appliances\_included.}(\text{Washing\_machine} \sqcap \text{Dryer})$   
 $\exists \text{hasRoom.} \text{Laundry} \sqsubseteq \exists \text{appliances\_included.}(\text{Washing\_machine} \sqcap \text{Dryer})$   
 $\text{hasSharedRoom} \sqsubseteq \text{hasRoom}$   
 $\text{hasBedroom} \sqsubseteq \text{hasRoom}$   
 $\text{lease\_period\_negotiable} \equiv \text{lease\_period} \leq 12$   
 $\text{Bills} \equiv \text{ONE-OF}\{\text{Gas}, \text{Electricity}, \text{Water}\}$   
 $\text{House} \equiv \exists \text{hasBedroom} \sqcap \exists \text{hasRoom.} \text{Bathroom}$

**Fig. 3.** The Ontology used in the example

### 3 The Negotiation Protocol

Following the idea in [5] agents negotiate making proposals and counter-proposals. At each round the agent can decide to *accept* the proposal made by its opponent, to *reject* such a proposal or to *refine* the proposal itself, until an agreement is reached or an agent quits the negotiation. In order to define the protocol we define the *utility of a proposal* and a motivated proposal. A *motivated proposal* is a proposal such that 1. it is not in conflict with what has been agreed in the previous rounds of the protocol: 2. it increases the utility of the agent making it. We call such a protocol *Cumulative protocol* because the agreement is reached in an incremental way, adding at each round a goal or a set of goals, which can be satisfiable w.r.t. the ontology. It is possible to demonstrate that such a protocol converges. Given the rules of the protocol it is possible to pursue different strategies. We analyzed the properties of two strategies, the IWIN strategy and



CoCo strategy. The former aims at maximizing in the first step the utility of the agent making the first proposal; although we demonstrated that this strategy allows to reach Pareto-efficient agreements, it may yield to highly *unfair* distributions of utilities in a proposal, hence, a negotiation where both agents immediately saturate their utility is likely to be a very long sequence of proposals and rejections. Therefore we introduced another strategy called CoCo, where agents try to compensate the loss in utility when they accept a proposal made by the opponent. Since the CoCo strategy uses a greedy approach the final agreement might not be Pareto-efficient, yet a computational analysis has clearly shown that CoCo has a low computational effort compared to IWIN<sup>1</sup>.

## 4 Relations with Related Work and Conclusion

Automated bilateral negotiation between agents has been widely investigated, both in artificial intelligence and in microeconomics research communities. AI-oriented research has usually focused on automated negotiation between agents and on designing high-level protocols for agent interaction. Agents can play different roles: act on behalf of buyer or seller, but also play the role of a mediator or facilitator. In the following we give a brief overview of logic-based approaches to automated negotiation, comparing our approach with existing ones and highlighting differences. In [1] the use of propositional logic in multi-issue negotiation was investigated, while in [2] weighted propositional formulas in preference modeling were considered. However, in such works, no semantic relation between issues is taken into account. In our approach we adopt a logical theory, *i.e.*, an ontology, which allows one *e.g.*, to catch inconsistencies between demand and supply, model implication, find out a feasible agreement in a bundle, which are fundamental issues to model in a negotiation setting. In [13] an agreement is defined as a model for a set of formulas from both agents. The approach does not take preferences into account, so that it is not possible to guarantee that the reached agreement is Pareto-efficient. With reference to the work presented in [14], adopting a propositional logic setting, *common knowledge* is considered as just more entrenched preferences, that could be even dropped in some deals. We adopt a *knowledge base*, or ontology  $\mathcal{T}$ , of formulas which are common knowledge for both agents, whose constraints must always be enforced in the negotiation outcomes. Moreover we use *additive utilities* over formulas: this allows an agent to make compensations between its requests and its concessions, while in [14] the concession of a more entrenched formula can never be compensated by less entrenched ones, no matter how many they are. Finally we devised a *protocol* which the agents should adhere to while negotiating; in contrast in [14] a game-theoretic approach is taken, presenting no protocol at all, since communication between agents is not considered. Prior to considering description logics as logical language for bilateral negotiation, we have studied different negotiation mechanism with the presence of a mediator [7,6,11] and without a mediator with partial incomplete information [10], in the framework of theory-endowed propositional logic. To the best of our knowledge, our approach is the first one using DLs to design a logic-based negotiation mechanism, ensuring a greater expressiveness w.r.t. propositional logic. Moreover,

<sup>1</sup> For the sake of conciseness we omit a lot of details about the protocol and the evaluation of strategies, which can be found in [9].



w.r.t. to non-logic-based approaches, the use of an ontology  $\mathcal{T}$  allows exploiting inference services that are used in the actual negotiation mechanisms. In future work we are planning to validate our approach with agent-based simulations and we are also setting up an analysis of the game theoretic properties, as related properties of the negotiation protocols ( *e.g.*, Pareto-efficiency), equilibrium strategies or properties of the agents (*e.g.*, individual rationality). Research is also ongoing exploiting, in the negotiation mechanism, non-monotonic inferences in DLs [3]—namely Concept Contraction—to identify and manage conflicting information [8].

## References

1. Bouveret, S., Lemaitre, M., Fargier, H., Lang, J.: Allocation of indivisible goods: a general model and some complexity results. In: Proc. of AAMAS 2005, pp. 1309–1310 (2005)
2. Chevaleyre, Y., Endriss, U., Lang, J.: Expressive power of weighted propositional formulas for cardinal preference modeling. In: Proc. of KR 2006, pp. 145–152 (2006)
3. Di Noia, T., Di Sciascio, E., Donini, F.M.: Semantic Matchmaking as Non-Monotonic Reasoning: A Description Logic Approach. JAIR 29, 269–307 (2007)
4. Kraus, S.: Strategic Negotiation in Multiagent Environments. MIT Press, Cambridge (2001)
5. Parsons, S., Sierra, C., Jennings, N.: Agents that reason and negotiate by arguing. J. of Log. and Comp. 8(3), 261–292 (1998)
6. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: A logic-based framework to compute pareto agreements in one-shot bilateral negotiation. In: Proc. of ECAI 2006, pp. 230–234 (2006)
7. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: Propositional-logic approach to one-shot multi issue bilateral negotiation. ACM SIGecom Exchanges 5(5), 11–21 (2006)
8. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: Alternating-offers protocol for multi-issue bilateral negotiation in semantic-enabled marketplaces. In: Proc. of ISWC 2007 (to appear, 2007)
9. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: Description logics for multi-issue bilateral negotiation with incomplete information. In: proc. of AAI 2007, pp. 477–482. AAAI, Stanford, California, USA (2007)
10. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: Logic-based alternating-offers protocol for automated multi-issue bilateral negotiation in P2P e-marketplaces. In: Proc. of DL 2007 (2007)
11. Ragone, A., Di Noia, T., Di Sciascio, E., Donini, F.M.: When price is not enough: Combining logical and numerical issues in bilateral negotiation. In: proc. of AAMAS 2007, pp. 97–99. ACM Press, New York (2007)
12. Rosenschein, J.S., Zlotkin, G.: Rules of Encounter. MIT Press, Cambridge (1994)
13. Wooldridge, M., Parsons, S.: Languages for negotiation. In: Proc. of ECAI 2000, pp. 393–400 (2000)
14. Zhang, D., Zhang, Y.: A computational model of logic-based negotiation. In: Proc. of AAI 2006, pp. 728–733. AAAI, Stanford, California, USA (2006)

# Cultural Adaptivity for the Semantic Web

Katharina Reinecke

Department of Informatics, University of Zurich,  
Binzmühlestr. 14, 8050 Zurich, Switzerland  
reinecke@ifi.uzh.ch

**Abstract.** Our research explores the possibilities for factoring culture into user models, working towards cultural adaptivity in the Semantic Web. The aim is to automate the localization process by acquiring details about an individual's cultural background. The paper shows how we have based our approach on research findings from related fields, outlining the connection between the cultural user and domain model ontologies, and a test environment. Furthermore, we describe a preliminary plan of how to validate our cultural user model by comparison to our previous experience in the manual adaptation of *imbuto*.

## 1 Motivation

Culturally adapted user interfaces have been demonstrated to outperform ordinary applications for culturally diverse user groups with regard to user acceptance and usability in general. Our experience with *imbuto*, a learning software developed for Rwandan agricultural advisers, underlined the assumption that work efficiency and user satisfaction drastically increased with the manual localization of the UI [1]. On the other hand, extensive circular adaptations over five months were required to be implemented on-site in Rwanda. An extension of the target group to encompass other cultures would require even greater time. Cultural software localization can, therefore, be seen as *extremely time-consuming and prohibitively expensive*. However, this is not the only problem with manual localization: Culture, by its elusive and intangible nature, makes it hard to recognize one's own preferences. Therefore, *users cannot be expected to know themselves where the system should be adapted* [2].

We propose to tackle these problems with cultural adaptivity, and thus, to automate the personalization process with the help of a cultural user model. After exploring our requirements and aligning them with several techniques for building adaptive systems (see [3]), our requirements for a distributed and reusable user profile – enabling an extension to numerous application and devices for cross-system personalization [4] – led to the decision to exploit Semantic Web services for our purposes.

In the following, we discuss the current state of related research (Section 2), present our own advances towards a semantically represented cultural user model and provide a research plan for advancing and evaluating this system (Section 3). The paper closes with a discussion of our contribution to current Semantic Web research and Human-Computer Interaction (HCI) in general.

## 2 Related Work

The idea of semantic user modeling for culturally adaptive systems is based on work from the related fields of international usability and culture, semantic user modeling and adaptive hypermedia. However, these isolated research efforts have not yet brought forward an approach comparable to ours. To our knowledge, Kamenz [5] is alone in her attempt to integrate culture in an approach to modeling learner characteristics of an adaptive e-learning application. The resulting system, however, does not meet our requirement for cross-system personalization, as introduced in [4].

*Culture and Usability.* Considering culture in the software development process has been suggested by many researchers in the field of HCI. The motivating factor behind it is the enormous improvements in working efficiency seen as a result of (manually) localized UIs [6,7]. Researchers in this area have mainly concentrated on applying measurable classifications of culture developed by cultural theorists Hofstede and Trompenaar [8,9] to the field of UI design [10,7,5]. In our approach, the cultural dimensions developed by Hofstede serve as a basis for the initial classification of the user model (please refer to [3] for more details).

*Semantic User and Domain Modeling.* With the rise of the Semantic Web, application specific user models have been countered with approaches to distributed user modeling [11], sharing user models through ontologies [12]. The Portal Adaptation Ontology [13], for example, includes a user profile ontology as well as an adaptation ontology that holds knowledge about the adaptation process. Further efforts include the General User Model Ontology (GUMO), which supplies a method “for the uniform interpretation of distributed user models in intelligent Semantic Web enriched environments” [14]. Parts of GUMO’s user modeling functionalities have been used by the Personal Reader project, which provides an environment for the construction of personalized Web content readers [15]. GUMO also serves as a supplement for our cultural user model.

Apart from the representation of a user model, user modeling requires the *acquisition of data* throughout the user’s interaction with the system. So far, most personalization mechanisms have constructed their user model by analyzing navigational behavior [16]. A more detailed approach to tracking user behavior is described in [17]: Here, a semantically rich user model is built by combining the web development technique AJAX with the Semantic Web. Advantages of this approach are the on-the-fly adaptation that removes the need for reloading a page and the ability to record scrolling, mouse-over and keystroke events [17]. As we assume that inferring the user’s culture does require such a sophisticated navigation analysis, we have adopted this approach for our test application.

Furthermore, we need to perform some kind of *inferencing on the obtained data* to access implicit information. Efforts in this direction include the Web Usage Ontology Generation [12] that applies fuzzy logic techniques to derive knowledge about the user’s behavior. Only recently, Roberto and da Silva [18] presented an algorithm to classify the user’s intentions recorded in a semantic

log. Again, this approach presumes that the analyzed web site has an underlying adaptation ontology. In the past, many projects also made use of the strong reasoning support provided by the Ontobroker inference engine [19].

### 3 Research Plan

The aim of our work is to extend existing approaches to user modeling in the Semantic Web by adding cultural dimensions to the user's profile. The information about the user's position in the cultural dimensions will provide applications in different domains with the ability to adequately localize the UI and its content.

We will integrate our cultural user model in an AJAX portal, which we are currently developing to support the online storage of to-do lists and notes. The application will be accessible to different devices such as mobile phones or computers. The target group consists of people from around the world, however, we will initially test the portal with students from Switzerland (thus being influenced by the French, German, Italian and Swiss cultures) as well as with Rwandan students.

#### 3.1 Current State of Our Research

*User and Domain Model.* So far, we have developed an OWL ontology that is composed of essential classes for outlining the user's culture. This *cultural user modeling ontology* covers general information, such as the user's birthplace, the religion or the parents' nationality. Each of these classes have been allocated an influence factor. According to the user's age, the duration of stays abroad is also included in the determination of his culture.

Furthermore, the cultural user modeling ontology contains Hofstede's five dimensions (see [8]) which can be extracted to represent the user's cultural position in a five-dimensional space. This information is then linked with the domain model ontology which we are currently working on. Our preliminary work in this respect includes the integration of results from existing evaluations on the effects of culture on UIs and content adaptation. The outcome were guidelines for a *cultural domain model* that explain the necessary localization strategies for each of the five dimensions. The classification of the user into the dimensions is carried out by taking the information gained from an initial interview about his origin and other cultural influences. Taking the percentages of each influencing factor, we can calculate the user's exact score for each dimension. Our cultural domain model then gives information about the effect on the UI localization: A high score in the dimension *Uncertainty Avoidance*, for example, triggers the adaptation to a linear navigation and a complex but clearly arranged spatial organisation of elements, among others.

*Reasoning.* Making inferences about culture works nearly the same way as in conventional personalization systems: The cultural dimensions are represented by integer values in the user modeling ontology and are assigned a default value that describes the average rating. Additionally, we are going to allocate each

cultural dimension to a certain user behavior. In our approach, this behavior is tracked by an annotated AJAX interface and is stored in the user model. Furthermore, we plan to align the resulting patterns with the underlying adaptation ontology which triggers the adaptation of UI elements.

*Application Domain.* At present, we are implementing a portal combining Semantic Web and AJAX technologies. From our experience with *imbuto* we have gained substantial information about the manual localization of a UI and the content to the Rwandan culture. This provides the means for the verification of our cultural user and domain model: If evaluations with Rwandans using the portal produce comparable results regarding user satisfaction and work efficiency, our approach can be assumed to have successfully automatized the UI localization.

### 3.2 Plan for Further Research

Although we have laid the foundations for cultural adaptivity, there are still several steps to take: Firstly, we have to revise the user and adaptation ontologies before connecting them to the portal. The adaptation mechanisms have to be extended with adequate inference rules that map certain user behaviour to interface adaptations. Secondly, we plan to validate the ontologies. Evaluations will be carried out in 2008 with Bachelor and Master students from the University of Zurich and at the National University of Rwanda. Each user group has to work with a non-localized version first in order to be able to classify the usability tested after the automatic localization. The evaluations with Rwandan students are especially useful to verify our results: We plan to compare the automatic adaptations resulting from the use by the Rwandan students with our results of the manual adaptation in *imbuto*. More precisely, the evaluations will include the same usability test procedures as used for *imbuto*, as well as an analysis of the interface characteristics after the adaptation. These evaluations will then help us to refine our ontologies.

## 4 Conclusion

We presented our approach to cultural adaptivity using the Semantic Web for both user modeling and adaptation. We extend upon research cited throughout this paper in various ways. Firstly, we combine knowledge about cultural localization with personalization techniques. Secondly, we work towards a reusable, distributed user model that allows for the mapping of culture to various application domains. Thirdly, we are implementing a test environment that allows for extensive usability tests in regards to culturally extended ontologies and personalization in general. Therefore, our scientific contribution in the field of personalization in the Semantic Web is an extended user model ontology that we will verify by comparing manual and automatic adaptation to a target culture. The practical impact of our approach is an automatic cultural localization possibility that reduces both development time and expenses.

## References

1. Reinecke, K.: A Learning Platform for the Education of Agricultural Advisers in Rwanda (2005), <http://www.ifi.uzh.ch/ddis/people/reinecke/>
2. Kobsa, A.: User Modeling: Recent Work, Prospects and Hazards. In: Adaptive user interfaces: Principles and practice., North-Holland, pp. 111–128 (1993)
3. Reinecke, K., Bernstein, A.: Culturally Adaptive Software: Moving Beyond Internationalization. In: Proceedings of the 12th International Conference on Human-Computer Interaction, Beijing, China, Springer, Heidelberg (2007)
4. Mehta, B., Nejdl, W.: Intelligent Distributed User Modelling: from Semantics to Learning. In: International Workshop on Ubiquitous and Decentralized User Modeling, Corfu, Greece (2007)
5. Kamentz, E.: Adaptivität von hypermedialen Lernsystemen. Phd thesis, University of Hildesheim (2006)
6. Barber, W., Badre, A.: Culturability: The Merging of Culture and Usability. In: Proceedings of the 4th Conference on Human Factors and the Web (1998)
7. Ford, G., Gelderblom, H.: The Effects of Culture on Performance Achieved through the use of Human Computer Interaction. In: Proceedings of the 2003 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologies on Enablement through Technology (SAICSIT) (2003)
8. Hofstede, G.: Culture's Consequences: Comparing values, behaviours and organisations across nations. Sage Publications Inc., Thousand Oaks (2003)
9. Trompenaars, F., Hampden-Turner, C.: Riding the Waves of Culture: Understanding Cultural Diversity in Business. Nicholas Brealey Publishing Ltd (1997)
10. Baumgartner, V., Marcus, A.: A Practical Set of Culture Dimensions for Global User-Interface Development. In: Proceedings of the 6th Asia-Pacific Conference on Computer-Human Interaction (2004)
11. Dolog, P., Nejdl, W.: Challenges and Benefits of the Semantic Web for User Modelling. In: Adaptive Hypermedia 2003 (2003)
12. Zhou, B., et al.: Web Usage Mining for Semantic Web Personalization. In: Workshop on Personalization on the Semantic Web, Edinburgh, UK, pp. 66–72 (2005)
13. Stojanovic, L., Thomas, S.: Fostering Self-Adaptive e-Government Service Improvement Using Semantic Technologies (2006)
14. Heckmann, D., et al.: GUMO - the General User Model Ontology. In: The 10th International Conference on User Modeling, Edinburgh, UK, pp. 428–432. Springer, Heidelberg (2005)
15. Henze, N.: Personalization Services for e-Learning in the Semantic Web. In: Proceedings of the 2nd International Workshop on Adaptive Systems for Web-Based Education: Tools and reusability, Amsterdam, The Netherlands (2005)
16. Eirinaki, M., Vazirgiannis, M.: Web Mining for Web Personalization. ACM Transactions on Internet Technology (TOIT) 3(1), 1–27 (2003)
17. Schmidt, K., et al.: On Enriching Ajax with Semantics: The Web Personalization Use Case. In: Proceedings of the 4th European Semantic Web Conference, Innsbruck, Austria (2007)
18. Roberto, R.L., da Silva, S.: An Approach for Identification of User's Intentions during the Navigation in Semantic Websites. In: Proceedings of the 4th European Semantic Web Conference, Innsbruck, Austria (2007)
19. Oberle, D., Wenke, D., Volz, R., Staab, S.: OntoBroker and OntoEdit Adaptation. Technical report (2003)

# Lightweight Community-Driven Ontology Evolution

Katharina Siorpaes

Digital Enterprise Research Institute (DERI), University of Innsbruck, Austria  
katharina.siorpaes@deri.at

**Abstract.** Only few well-maintained domain ontologies can be found on the Web. The likely reasons for the lack of useful domain ontologies include that (1) informal means to convey intended meaning more efficiently are used for ontology specification only to a very limited extent, (2) many relevant domains of discourse show a substantial degree of conceptual dynamics, (3) ontology representation languages are hard to understand for the majority of (potential) ontology users and domain experts, and (4) the community does not have control over the ontology evolution. In this thesis, we propose to (1) ground a methodology for community-grounded ontology building on the culture and philosophy of wikis by giving users who have no or little expertise in ontology engineering the opportunity to contribute in all stages of the ontology lifecycle and (2) exploit the combination of human and computational intelligence to discover and resolve inconsistencies and align lightweight domain ontologies. The contribution of this thesis is a methodology and prototype for community-grounded building and evolution of lightweight domain ontologies.

## 1 Problem Statement

Even though one can observe highly active research in the areas of ontologies and the Semantic Web, there is a lack of up-to-date domain ontologies for many fields. The complexity of reality makes building ontologies difficult in several ways: (1) Ontology engineering methodologies tend to focus on formal means for specifying ontologies. However, in order to describe the *intended* meaning of ontology elements, informal means are also needed. (2) Many relevant domains of discourse, such as e-commerce, show a high degree of conceptual dynamics, i.e. it is hard to keep up with the pace of change in reality [1]. (3) A prerequisite for using an ontology and thus committing to its view of the world is to be able to understand the meaning of concepts and relations [1]. This is problematic for many users, since they cannot easily figure out what they would be committing to when using a particular ontology file from the Web, as the discourse that lead to a certain specification is not available. (4) Most ontologies are built by a group of engineers and the user community does not have direct control over the evolution of the ontology.

## 2 Motivation

A community-oriented approach has several advantages as compared to an isolated engineering-oriented approach, where a small group of engineers carefully constructs

the ontology and releases it to the user community at a later point of time: (1) We believe that ontology building is inherently a collaborative task, for two reasons: first, ontologies are supposed to be community contracts [2, 3]; second, the combination of required domain expertise and modeling skills is more likely to be found in a group than in a single individual. (2) Timeliness of a domain ontology is important in rapidly changing domains. A community can keep up with the pace reality is changing more easily. (3) The user community is more likely to agree on a view of the world that is represented by the ontology. Therefore, it is likely that this community will also actually use and further develop the ontology as it is not a subjective conceptualization based on an outdated state of the world. (4) The burden of creating, maintaining the ontology can be shared by those benefiting from the ontology. Similarly, inconsistencies can be discovered and resolved faster. In [4], we provide evidence for the assumption that a community can be trusted with the task of building lightweight ontologies by showing that the conceptual reliability of Wikipedia articles is very high (93% of URIs convey a stable ontological meaning, even though Wikipedia is maintained by a highly diverse group of actors).

### 3 Approach and Design Principles

In the following we outline the design principles of this approach: **Tapping the “wisdom of crowds”**: We believe that a diverse community consisting of domain experts as well as knowledge engineers is smarter and more agile than only a small ontology development team. Instead of forcing one view of the world onto the user communities, we envision co-existence and interoperability of conflicting views and to support the community in achieving consensus. **Openness and multimedia richness**: Traditional ontology building environments usually impose quite high entrance barriers on a user while wikis allow many users to contribute easily with only basic Web-editing skills. The culture of Wikis is the underlying paradigm of this work. Additionally, we propose the use of multimedia elements to better convey the informal part of the intended meaning of a concept. **Integration of external data**: with the increasing popularity of “Web 2.0”, a wealth of data is available, which can be exploited for improving the process of ontology construction. **Combination of human and computational intelligence**: Building ontologies is a task that depends on human intelligence, both as a source of domain expertise and for producing a consensual conceptualization as well as aligning ontologies and resolving inconsistencies. We aim at developing functionality that combines human and computational power and thus supports users in achieving several ontology building tasks.

### 4 Methodology and Contribution

The envisioned research methodology is as follows: (1) Analysis of a trade-off between expressivity and agility of an ontology meta-model suitable for a broad audience. (2) Deployment of a suitable model for ontology evolution. (3) Integration of external resources to support lexical and multimedia-based enrichment: tagging,



i.e. describing objects with keywords to enhance content retrieval, avoids the limitations known in ontology building as they allow the instant maintenance of tags. Statistical analysis of the large amount of data produced in tagging application can be used for ontology construction [5, 6]. Additionally, Web resources, such as Google, Wikipedia, or Freebase, and terminological resources, such as Wordnet, will be exploited for ontology construction and background enrichment. (4) Specification of user roles and processes supporting the community in achieving consensus. (5) Combination of human and computational intelligence: this includes discovering and resolving inconsistencies and aligning ontologies by combining matching algorithms with human intelligence, mapping Wikipedia special pages to an ontology building environment, and achieving stable releases of ontologies. (6) Application of various techniques for visualization of ontologies and user interfaces to foster comprehensibility. (7) Investigating user incentives for contributing to ontology building in a community-driven environment. (8) Implementation of a community portal based on the methodology and evaluation: the portal will be released to a community who will create and maintain several domain ontologies. The data will be logged and analyzed, in order to find out about the domain coverage, stability of concept definitions, cost, user participation, and precision of usage.

## 5 Related Work

The work related to this thesis can be divided into the following areas: **Traditional ontology engineering** methodologies are described and compared in [3]. **Collaborative and community-driven ontology engineering:** OntoSaurus [7] is a web-based tool for editing and browsing ontologies; however, it has no explicit support for collaborative or community-based work. Domginue [8] describes Tadzebao and WebOnto: Tadzebao supports asynchronous and synchronous discussions on ontologies. WebOnto complements Tadzebao by supporting collaborative browsing, creation and editing of ontologies. Vrandecic and colleagues [9] describe the DILIGENT knowledge process where ontology evolution and collaborative concept mapping are applied to deal with conceptual dynamics of domains. OntoEdit [10] is a collaborative ontology editing environment. Kotis and Vouros describe the HCOME methodology in [11], which is a “human-centered” approach: ontology development is defined as a dynamic process with a focus on ontology evolution. HCOME is a decentralized engineering model where everyone first formalizes her own ontology and shares it in a further step within the community. This is different to our approach as the ontology is not open to the community at all times. Braun and colleagues [12] present an ontology maturing process consisting of four steps: emergence of ideas, consolidation in communities, formalization, axiomatization. They regard the evolution of an ontology as maturing from tags to formal ontologies via the community. Gibson and colleagues [13] distinguish between ontological and meta-ontological data and propose that more attention should be dedicated to the meta-ontological data and to allow the community to discuss ontologies at an early stage of development with a Web 2.0 user interface, where no formal model is required. They emphasize the integration of natural language. **Semantic Wikis:** [14] describe Makna, a Wiki engine that was extended with generic

ontology-driven components that allow collaborative authoring, querying, and browsing Semantic Web information. IkeWiki [15] allows annotating links, typing of pages, and context dependent content adaptation. [16] have the objective to make the knowledge within Wikipedia, the online encyclopedia, machine-accessible by adding semantic information. Platypus Wiki [17] focus on the creation of RDF (instance) data, Platypus Wiki aims at augmenting a wiki with semantics. WikiFactory [18] is a framework that allows the automatic generation of domain specific wikis. The main difference to our work is that existing approaches aim at augmenting existing wiki content with semantics instead of using a Wiki-like infrastructure as an environment for collaboratively building ontologies.

## 6 Expected Impact

The final goal of the thesis is to develop a deployable prototype for wiki-based ontology construction by large, unsupervised audiences. The process of community-driven ontology construction and maintenance will be supported by background intelligence in the tool that helps the community by suggestions based on statistically significant patterns or external lexical or structural resources. Eventually, thus should hand back the control over the evolution of ontologies to the community in order to overcome the current shortage of domain ontologies.

**Acknowledgments.** I would like to thank Martin Hepp for supervising my thesis and his continuous support. The work presented in this paper has been funded by the Austrian BMVIT/FFG under the FIT-IT project myOntology (grant no. 812515/9284).

## References

1. Hepp, M.: Possible Ontologies: How Reality Constrains the Development of Relevant Ontologies. *IEEE Internet Computing* 11(7), 96–102 (2007)
2. Hepp, M., Bachlechner, D., Siorpaes, K.: OntoWiki: Community-driven Ontology Engineering and Ontology Usage based on Wikis. In: *Proceedings of the 2005 International Symposium on Wikis (WikiSym 2005)*, San Diego, California, USA (2005)
3. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: *Ontological Engineering*. Springer, Heidelberg (2004)
4. Hepp, M., Siorpaes, K., Bachlechner, D.: Harvesting Wiki Consensus: Using Wikipedia Entries as Vocabulary for Knowledge Management. *IEEE Internet Computing* 11(5), 54–65 (2007)
5. Van Damme, C., Hepp, M., Siorpaes, K.: FolksOntology: An Integrated Approach for Turning Folksonomies into Ontologies. In: *ESWC 2007. Proceedings of the Workshop Bridging the Gap between Semantic Web and Web 2.0*, Innsbruck, Austria (2007)
6. Specia, L., Motta, E.: Integrating Folksonomies with the Semantic Web. In: *ESWC 2007. Proceedings of the European Semantic Web Conference*, Innsbruck, Austria. LNCS, vol. 2007, Springer, Heidelberg (2007)
7. Swartout, B., et al.: Ontosaurus: a tool for browsing and editing ontologies. In: *Proceedings of the 9th Banff Knowledge Acquisition for Knowledge-based systems Workshop*, Banff, Canada (1996)

8. Domingue, J.: Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web. In: Proceedings of the 11th Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada (1998)
9. Vrandečić, D., et al.: The DILIGENT knowledge process. *Journal of Knowledge Management* 9(5), 85–96 (2005)
10. Sure, Y., et al.: OntoEdit: Collaborative Ontology Engineering for the Semantic Web. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, Springer, Heidelberg (2002)
11. Kotis, K., Vouros, G.A.: Human-centered ontology engineering: The HCOME Methodology. *Knowledge and Information Systems* 10(1), 109–131 (2005)
12. Braun, S., et al.: Ontology Maturing: a Collaborative Web 2.0 Approach to Ontology Engineering. In: WWW 2007. Proceedings of the Workshop on Social and Collaborative Construction of Structured Knowledge at the 16th International World Wide Web Conference, Banff, Canada (2007)
13. Gibson, A., Wolstencroft, K., Stevens, R.: Promotion of Ontological Comprehension: Exposing Terms and Metadata with Web 2.0. In: WWW 2007. Proceedings of the Workshop on Social and Collaborative Construction of Structured Knowledge at 16th International World Wide Web Conference, Banff, Canada (2007)
14. Dello, C., Simperl, E.P.B., Tolksdorf, R.: Creating and using semantic content with Makna. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)
15. Schaffert, S., IkeWiki, A.: Semantic Wiki for Collaborative Knowledge Management. In: STICA 2006. 1st international workshop on Semantic Technologies in Collaborative Applications (2006)
16. Völkel, M., et al.: Semantic Wikipedia. In: WWW 2006. Proceedings of the 15th International Conference on World Wide Web, Edinburgh, Scotland (2006)
17. Campanini, S.E., Castagna, P., Tazzoli, R.: Platypus Wiki: a Semantic Wiki Wiki Web. In: Campanini, S.E. (ed.) SWAP 2004. Proceedings of the 1st Italian Semantic Web Workshop Semantic Web Applications and Perspectives, Ancona, Italy (2004)
18. Di Iorio, A., Presutti, V., Vitali, F.: WikiFactory: An Ontology-Based Application for Creating Domain-Oriented Wikis. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)

# Exploiting Patterns in Ontology Mapping

Ondřej Šváb

Department of Information and Knowledge Engineering,  
University of Economics, Prague, W. Churchill Sq. 4, 130 67 Praha 3, Czech Republic  
svabo@vse.cz

**Abstract.** Unsatisfactory performance of ontology mapping methods on many tasks seems to indicate the need for employing some background knowledge. My PhD work focuses on exploiting pattern-like structures for this purpose. Design patterns are related to different styles of modelling ontology structure and/or naming of concepts. On the other hand, mapping patterns emerge as result of the mapping process. Preliminary results and schedule of future work are presented.

## 1 Introduction

Current ontology mapping methods are typically based on generic methods and/or ad hoc heuristics. However, ontologies themselves are not randomly assembled graph structures but conceptual models of reality, and thus reflect some explicit or implicit patterns that have already been identified in ontological engineering research.

We examine two different though interrelated notions of patterns that are relevant for ontology mapping. One is that of *mapping pattern*: a frequently appearing structure involving semantic links both across the mapped ontologies (i.e. alignments resulting from the mapping process) and within each of them. The other is that of *design pattern* in the general sense: modeller's choices such as *naming conventions* or *structural patterns*. While the analysis of mapping patterns can provide insight into the functionality of mapping methods a posteriori, the prior analysis of design patterns can help select or tune the mapping methods a priori.

Section 2 of the paper briefly surveys the experiments with mapping patterns. Section 3 discusses the role of design patterns in general, and reports on preliminary evaluation of sensitivity of a popular graph-based mapping method (similarity flooding) to the presence of some simple structural design patterns. Finally, the paper includes a survey of related work (section 4) and a schedule of future work (section 5).

## 2 Mapping Patterns

Mapping patterns reflect the *internal structure* of ontologies as well as *mappings* between elements of (typically, two) ontologies. A mapping pattern is a graph structure, where nodes are classes, properties or instances. Edges represent mappings, relations between elements (eg. domain and range of properties) or structural relations between classes (eg. subclasses or siblings). In our experiments we employed three simple patterns. The *first* one is depicted in Figure 1. The left-hand side (class A) is from ontology

O1 and the right-hand side (class B and its subclass C) is from ontology O2. There is a mapping between A and B and at the same time between A and C. The *second* pattern is depicted in Figure 2. It is quite similar to the previous one, but now we consider a child and a parent from each ontology and simultaneous mappings between parents and between children. The *third* pattern is depicted in Figure 3. It consists of simultaneous mappings between class A from O1 and two sibling classes C and D from O2.

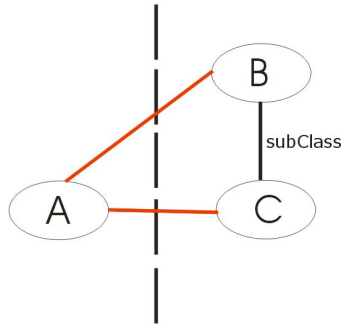


Fig. 1. Pattern 1 – ‘Parent-child triangle’

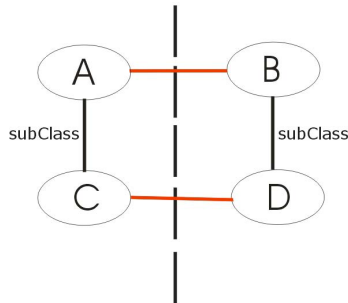


Fig. 2. Pattern 2 – ‘Mapping along taxonomy’

First experiments have been done over mapping results from participants of ‘Conference track’<sup>1</sup> within OAEI-2006 [2] where I was co-organiser. The goal of these experiments was to find out about features of different OM systems<sup>2</sup> via a Data Mining (DM) approach using mapping patterns. An example of a hypothesis discovered is: “Mappings output by the HMatch tool with medium validity (between 0,5 and 0,8) are more likely to connect a child with a class that is also connected (with high validity) with a parent (Pattern 1) than such correspondences with all validity values (on average).” Detailed description of the approach used and more results are in [8].

<sup>1</sup> <http://nb.vse.cz/~svabo/oeai2006/>

<sup>2</sup> This work also follows with my previous research in exploring interdependencies between mapping methods using Bayesian Networks, see [7].

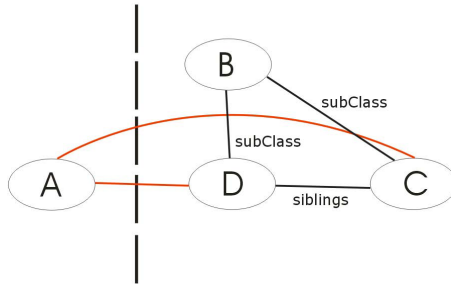


Fig. 3. Pattern 3 – ‘Sibling-sibling triangle’

## 3 Ontology Design Patterns

### 3.1 Design Patterns Types

The structure and naming in ontologies reflect diverse modelling styles of their creators. Typically, their aim is to ease the comprehensibility of the ontology for themselves and for others. We can observe symptoms of modelling styles as (design) patterns, which have been used either consciously, e.g. based on recommendations provided by the SWBPD group<sup>3</sup>, or unconsciously. We are particularly interested in two types of ontology design patterns: *naming conventions* and *structural patterns*.

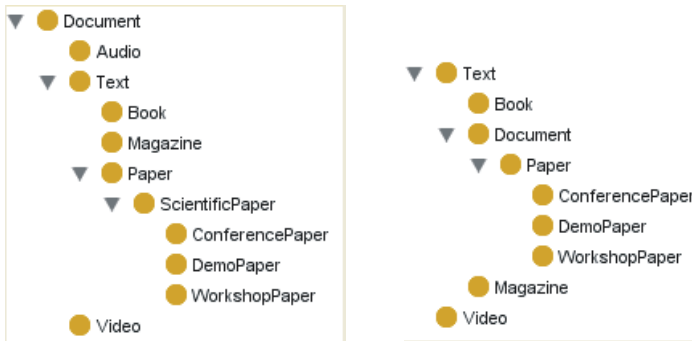
Naming conventions are related to naming classes, properties and/or instances. The way concepts are named can reveal their character. For example, in Figure 4 classes named with an appositive and noun (ConferencePaper) are subclasses of the class named with the noun (Paper). Another example could be the naming of inverse properties, e.g. *hasAuthor* vs. *authorOf*. Such heuristics are often globally used by the mapping tools, our aim however is to assess the degree in which each such pattern is a part of the author’s modelling style and then seek it systematically in the ontology in question.

Structural patterns concern the modelling choices in using certain ontology entities and connecting them together. An example is a situation when some part of an ontology can be either ‘class-centric’ or ‘property-centric’, e.g. one designer may prefer to use a property ‘holdsSeasonalTicket’ (of which the domain is ‘Person’), while another uses the class ‘SeasonalTicketHolder’ (as subclass of ‘Person’). Another example could be the situation where one ontology is more granular than other. E.g. in the left-hand ontology in Fig. 4 there is an intermediate class ‘ScientificPaper’ in contrast with the right-hand ontology where this concept is missing.

### 3.2 Initial Experiments: Impact of Patterns on Similarity Flooding

I just started to work on exploring how particular methods tackle with occurrences of patterns. The Similarity Flooding (SF) algorithm [5] is based on the intuition that elements of two distinct models are similar when their adjacent elements are similar. At the beginning, initial mappings are typically set; the similarity of two elements is then iteratively (partly) propagated to their respective neighbors.

<sup>3</sup> <http://www.w3.org/2001/sw/BestPractices/OEP/>



**Fig. 4.** Fragments of ontologies exemplifying design patterns

For initial setting, we used the string matcher distributed together with the SF code, which is based on comparing common prefixes and suffixes of literals. The question was whether SF would improve the initial string-based mapping specifically in the surroundings of patterns. The ontologies to be mapped were artificial variations of real ontologies from the OAEI ‘conference’ track. Patterns were systematically introduced to them, both naming conventions such as the couple ‘ConferencePaper’ and ‘Paper’ (helping the initial string matching) and structural patterns such as having ‘ScientificPaper’ as additional intermediate class in the left-hand ontology in Fig. 4. SF algorithm improves the result of string comparison of ‘ScientificPaper’ and ‘Paper’ (which could presumably be equal) from 0.03 to 0.89. In the case of ‘Paper’ (left) and ‘Paper’ (right) concepts, SF algorithm decreases the similarity from 1 to 0.31, which could perhaps be desirable, as in the right-hand ontology the semantics of ‘Paper’ seems to be connected to that of its subclasses (types of scientific paper), while in the left-hand ontology it is broader than ‘Scientific Paper’ and can thus also include non-scientific literature. On the other side, SF algorithm yields the similarity of concepts ‘ConferencePaper’ from the left-hand ontology and ‘ConferencePaper’ from the right-hand ontology as 0.25, while string method yields it correctly as 1.

There thus seems to be a trade-off between the gains/losses of the string-based and graph based decisions. However, if we could automatically recognise an instance of pattern consisting of co-occurrence of three siblings in both ontologies (which are even compound from an appositive and noun), we could prevent the graph-based method from reverting the correct decision yielded by the string-based method.

## 4 Related Work

*Mapping patterns* are implicitly considered in [4]; however, they focus on ‘heterogeneous mappings’ (class to property) as special kind of pattern. We also considered this, but it appeared too infrequently to allow for meaningful data mining. Our planned work on *design patterns* shares the general underlying idea with e.g. projects presented by Aleksovski [1] and Sabou [6], which also try to overcome inefficiencies of mapping

methods using some kind of additional ontological knowledge. However, both these projects exploit the content of additional ontologies, while we focus on detecting pre-defined patterns in the original ontologies to be mapped.

## 5 Schedule of Future Work

The main direction of my PhD work in the next cca 6 months will consist in evaluating different mapping methods in terms of sensitivity to diverse design patterns, as suggested in section 3. Based on the results of this first step, I will try, in another 6 months, to automatically recognise these patterns and exploit them in the mapping process.

As secondary topic, I would also like to consider a richer variety of ontology mapping patterns as input to the data mining process as posterior evaluation of mappings (cf. section 2).

*I would like to thank to my supervisor Vojtěch Svátek for directing my work and much invaluable advice. The research leading to this paper was partially supported by the IGA VSE grants no.12/06 Integration of approaches to ontological engineering: design patterns, mapping and mining, no.20/07 “Combination and comparison of ontology mapping methods and systems” and by the Knowledge Web Network of Excellence (IST FP6-507482).*

## References

1. Aleksovski, Z., ten Kate, W., van Harmelen, F.: Exploiting the Structure of Background Knowledge Used in Ontology Matching In: Workshop on Ontology Matching at ISWC 2006 (2006). In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
2. Euzenat, J., Mochol, M., Shvaiko, P., Stuckenschmidt, H., Šváb, O., Svátek, V., van Hage, W.R., Yatskevich, M.: Results of the Ontology Alignment Evaluation Initiative 2006. In: ISWC 2006. Workshop on Ontology Matching (2006)
3. Gangemi, A.: Ontology Design Patterns for Semantic Web Content. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
4. Ghidini, C., Serafini, L.: Reconciling concepts and relations in heterogeneous ontologies. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)
5. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity flooding: A versatile graph matching algorithm. In: ICDE 2002 (2002)
6. Sabou, M., d'Aquin, M., Motta, E.: Using the Semantic Web as Background Knowledge for Ontology Mapping. In: ISWC-2006. Workshop on Ontology Matching (2006)
7. Šváb, O., Svátek, V.: Combining Ontology Mapping Methods Using Bayesian Networks. In: ISWC 2006. Workshop on Ontology Matching (2006)
8. Šváb, O., Svátek, V., Stuckenschmidt, H.: A Study in Empirical and ‘Casuistic’ Analysis of Ontology Mapping Results. In: Accepted for ESWC 2007 (2007)



# Integrating Uncertainty into Ontology Mapping

Ying Wang

School of Electronics, Electrical Engineering and Computer Science,  
Queen's University Belfast, Belfast, BT7 1NN, UK  
ywang14@qub.ac.uk

**Abstract.** This paper gives an outline of my PhD thesis which describes the integration of managing uncertainty into ontology mapping. Ontology mapping is one of the most important tasks for ontology interoperability and its main aim is to find semantic relationships between entities (i.e. concept, attribute, and relation) of two ontologies. However, in the process of mapping, uncertainty and incompleteness of semantics in the syntactic representation and description of relations between entities in ontologies will lead to imprecise results. If we want to obtain better results, it becomes more significant for the ontology mapping to be able to deal with uncertainty.

## 1 Introduction to the Problem

Ontologies are cores in the Semantic Web because they are the carriers of the meaning contained in the Semantic Web. However in many cases, different domains define different ontologies containing the same concepts. Even in the same domain, different organizations construct different ontologies. Therefore, it is necessary to find mappings between ontologies.

In recent years, researchers have developed a number of tools for finding these mappings. In most cases, the mappings produced are imprecise. For instance, most automatic ontology mapping tools use heuristics or machine learning techniques, which are imprecise by their very nature. Even experts sometimes could not be sure about the exact match between concepts and just assign some certainty ratings to matches [1]. The reason why we get imprecise mapping results is to some extent because of uncertainty. Uncertainty means that an agent, i.e. a computer or a human, has only partial knowledge about the truth value of a given piece of information. Peter Haase thought that uncertainty can be distinguished to objective uncertainty and subjective uncertainty [2]. I think that subjective uncertainty plays the most prominent role that prevent ontology mapping from getting better results. For instance, in the process of mapping, one may use diverse range of rules to judge the mapping, but different rules imply different uncertainty and one does not know the accuracy and reliability of rules. There has been little focus on dealing with uncertainty in ontology mapping. However this problem began to attract attention of researchers in this area.

The rest of the paper is organized as follows. Section 2 presents the current state of existing solutions. Section 3 discusses the current approaches we have achieved so far and also problems we have encountered. Section 4 concludes the paper.

## 2 Uncertainty, Ontology Mapping and Ontology Mapping with Uncertainty

**Uncertainty.** There are many models to deal with uncertain information in artificial intelligence and the most popular methods are:

**1.Probability Theory.** It attempts to quantify the notion of probable. An increasing important approach to using probability theory in computing is probabilistic networks, also called Bayesian networks or Causal networks. Bayesian networks can be used to represent and reason more efficiently with probabilistic information.

**2.Evidence Theory.** The Dempster-Shafer theory (Evidence Theory) is a mechanism for representing and reasoning with uncertainty, imprecise and incomplete information [3]. Evidence theory has an ability to model information flexibly and it provides a convenient and simple mechanism for combining two or more pieces of evidence when they come from distinct sources.

**3.Possibility Theory.** It was introduced in 1978 by L.A. Zadeh [4], in connection with the fuzzy set theory, to allow a reasoning to be carried out on imprecise or vague knowledge, making it possible to deal with uncertainties on knowledge. Two typical merging operations in possibility theory are max and min which combine two possibility distribution into one.

**Ontology Mapping.** Ontology mapping is a critical problem in many application domains and many different mapping solutions have been proposed with diverse range of mapping techniques so far. For example, an integrated ontology mapping approach [5] is proposed based on rules and the method in quick ontology mapping [6] puts attention to the runtime of program. Anchor-PROMPT is a tool for ontology merging and mapping [7][8]. It contains a sophisticated prompt mechanism for possible mapping entities. In [9], researchers from several universities are working together to create an ontology mapping approach based on information flow. The approach of semantic enrichment for ontology mapping exploits text categorization to automatically assign documents to the concept in the ontology and use the documents to calculate the similarities between concepts in ontologies [10].

**Ontology Mapping with Uncertainty.** In the mapping process, if only syntactic or element-level matching is performed, as in the case for name matching without the use of a thesaurus, inaccuracies can occur [11]. It affects the results of mapping, but so far only a few ontology mapping methods considered dealing with uncertainty issue.

Nagy et al [12] and Besana [13] both recognized the importance of uncertainty in ontology mapping, and both of them used Dempster-Shafer theory to assist mapping. They believed that different matchers have uncertainties associated with them, so they combine the results obtained from different matchers using DS theory and it is possible to give a uniform interpretation, consistent with the uncertainty inherited in the problem.

In [14] a Bayesian Networks based approach was designed and a system called BayesOWL was proposed. In this approach, the source and target ontologies are

first translated into Bayesian networks (BN); the concept mapping between the two ontologies are treated as evidential reasoning between the two translated BNs. Probabilities, that are required for constructing conditional probability tables (CPT) during translation and for measuring semantic similarity during mapping, are learned using text classification techniques where each concept in an ontology is associated with a set of semantically relevant text documents, which are obtained by ontology guided web mining.

Holi and Hyvönen [15] thought that in the real world, concepts are not always subsumed by each other, and cannot always be organized in a crisp subsumption hierarchies. Many concepts only partly overlap with each other, so they present a new probabilistic method to model conceptual overlap in taxonomies, and an algorithm to compute the overlap between a selected concept and other concepts of a taxonomy by using Bayesian networks.

Zhao et al [16] proposed a novel similarity measure method based on rough set theory and formal concept analysis (RFCA) to realize ontology mapping tasks. The authors combined rough set theory into the similarity computation formula of formal concept analysis (FCA). Although the authors did not consider uncertainty in the process of mapping explicitly, they applied the rough set theory to measure the similarity of concepts of ontologies. So, in some case, they considered the uncertainty problem.

### 3 My Approaches

We have proposed a new ontology mapping approach called ACAOM [17]. It first uses WordNet to calculate similarities between words for the node names comparison in ontologies. WordNet is organized as a classified structure in which synset is the basic unit. Every set containing many words expresses a single meaning and we can utilize jwnl to access WordNet, thereby to compute similarities between names of nodes. This is a name-based mapping strategy. Then in the instance-based strategy we follow the following assumption: the more semantics is explicitly specified about the ontologies, the more feasible their comparison becomes. Here, the instances are documents assigned to the concept nodes. According to the definition of ontology mapping given above, given a node in one ontology, a mapping function searches the node in another ontology which has the most similar semantics to it. We employ approaches in information retrieval to assist ontology mapping. In this way, we can denote the documents that have been bound with nodes by using vector space models, and then the numerical degrees of similarity give a way to rank the matches.

Following that we are developing a new mapping method which uses three different and independent matchers: *Edit distance-based matcher*, *Linguistic-based matcher* and *Structure-based matcher*. In these three matchers, a new matcher is the *structure-based matcher* which utilizes the similarity measures between two words ( $w_1$  and  $w_2$ ), a father node of  $w_1$  with  $w_2$  and all the child nodes of  $w_1$  with  $w_2$ . This matcher takes both the semantics and the structure of an ontology into account. We then discuss how the mapping results from different matchers can be combined. We consider both the Dempster Shafer theory of

**Table 1.** Comparison of Experiment Results

Datasets	DS			PT			falcon			ola			ctxMatch2-1		
	p	r	f	p	r	f	p	r	f	p	r	f	p	r	f
101-103	100	98.97	99.48	100	98.97	99.48	100	100	100	100	100	100	87	34	48
101-104	100	98.97	99.48	100	98.97	99.48	100	100	100	100	100	100	87	34	48.89
101-205	46.88	46.39	46.63	30.29	29.90	30.09	88	87	87.5	43	42	42.5	36	4	7.2
101-223	100	98.97	99.48	100	98.97	99.48	100	100	100	100	100	100	83	31	45.14
101-302	45.83	45.83	45.83	43.75	43.75	43.75	97	67	79.26	37	33	34.89	0	0	0

evidence (DS theory) and Possibility Theory and apply them to combine the outcomes obtained by three different and independent matchers.

In our evaluation, we choose **Test 101**, **Test 103**, **Test 104**, **Test 205**, **Test 223** and **Test 302** of OAEI 2006 benchmark tests and take **Test 101** as the reference ontology. All of other ontologies are compared with **Test 101**. We use the combination mechanisms in both DS theory and Possibility Theory to combine the matching results from our three matchers. We now compare the outputs from the two combination rules to the results obtained from *falcon*, *ola* and *ctxMatch2-1* algorithms which were used in the EON 2005 Ontology Alignment Contest<sup>1</sup>, and the details are given in Table 1. In Table 1, *p* for precision, *r* for recall, *f* for f-measure, DS for Dempster’s combination rule, and PT for the minimum merging operator in Possibility Theory.

## 4 Conclusion

During my PhD study, my research will mainly focus on dealing ontology mapping with uncertainty, so I think I can begin from two aspects: ontology mapping and uncertainty theory. For ontology mapping, I need to develop more scientific matchers to find out mapping relations between ontologies. These matchers should put attention on how to find different kinds of mappings and how to improve the accuracy of mappings. For uncertainty theory, firstly, I will study the different methods of representing uncertainty. Secondly, I will try to utilize these different representing methods to represent ontology mappings. Thirdly, I will study how to use different combination rules of uncertainty in different situations. I hope that I can use different methods to handle different uncertainty problems in the process of ontology mapping successfully in the future.

## References

1. Choi, N., Song, I.Y., Han, H.: A survey on ontology mapping. SIGMOD Record 35(3), 34–41 (2006)
2. Haase, P., Völker, J.: Ontology learning and reasoning - dealing with uncertainty and inconsistency. In: The International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2005), collocated with the 4th International Semantic Web Conference (ISWC 2005), pp. 45–55 (2005)

<sup>1</sup> <http://oaei.ontologymatching.org/2005/results/>

3. Liu, W.: Propositional, Probabilistic and Evidential Reasoning: Integrating Numerical and Symbolic Approaches (2001)
4. Zadeh, L.A.: Fuzzy sets as a basis for a theory of possibility. *Journal of Fuzzy Sets and Systems* 100, 9–34 (1999)
5. Ehrig, M., Sure, Y.: Ontology mapping - an integrated approach. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) *ESWS 2004*. LNCS, vol. 3053, pp. 76–91. Springer, Heidelberg (2004)
6. Ehrig, M., Staab, S.: Qom - quick ontology mapping. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) *ISWC 2004*. LNCS, vol. 3298, pp. 683–697. Springer, Heidelberg (2004)
7. Noy, N.F., Musen, M.A.: Prompt: Algorithm and tool for automated ontology merging and alignment. In: *Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI 2000)*, pp. 450–455 (2000)
8. Noy, N.F., Musen, M.A.: Anchor-prompt: Using non- local context for semantic matching. In: *IJCAI 2001. Workshop on Ontologies and Information Sharing at the 17th International Joint Conference on Artificial Intelligence* (2001)
9. Kalfoglou, Y., Schorlemmer, W.M.: Information-flow-based ontology mapping. In: *Proceedings of the International Federated Conferences (CoopIS/DOA/ODBASE'02)*, pp. 1132–1151 (2002)
10. Su, X., Gulla, J.A.: Semantic enrichment for ontology mapping. In: Meziane, F., Métais, E. (eds.) *NLDB 2004*. LNCS, vol. 3136, pp. 217–228. Springer, Heidelberg (2004)
11. Cross, V.: Uncertainty in the automation of ontology matching. In: *ISUMA 2003. Proceedings of the 4th International Symposium on Uncertainty Modeling and Analysis*, pp. 135–140 (2003)
12. Nagy, M., Vargas-Vera, M., Motta, E.: Dssim-ontology mapping with uncertainty. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) *ISWC 2006*. LNCS, vol. 4273, Springer, Heidelberg (2006)
13. Besana, P.: A framework for combining ontology and schema matchers with Dempster-Shafer. In: *The International Workshop on Ontology Matching (OM 2006)*, collocated with the 5th International Semantic Web Conference (ISWC 2006), pp. 196–200 (2006)
14. Pan, R., Ding, Z., Yu, Y., Peng, Y.: A bayesian network approach to ontology mapping. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) *ISWC 2005*. LNCS, vol. 3729, pp. 563–577. Springer, Heidelberg (2005)
15. Markus Holli, E.H.: Modeling degrees of conceptual overlap in semantic web ontologies. In: *The International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2005)*, collocated with the the 4th International Semantic Web Conference (ISWC 2005), pp. 98–99 (2005)
16. Zhao, Y., Wang, X., Halang, W.A.: Ontology mapping based on rough formal concept analysis. *Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW 2006)* 180 (2006)
17. Wang, Y., Gong, J., Wang, Z., Zhou, C.: A composite approach for ontology mapping. In: *Proceedings of the 23rd British National Conference on Databases (BN-COD'23)*, pp. 282–285 (2006)

# Mutual Contextualization in Tripartite Graphs of Folksonomies

Ching-man Au Yeung, Nicholas Gibbins, and Nigel Shadbolt

Intelligence, Agents and Multimedia Group (IAM),  
School of Electronics and Computer Science,  
University of Southampton,  
Southampton SO17 1BJ, UK  
{cmay06r,nmg,nrs}@ecs.soton.ac.uk

**Abstract.** The use of tags to describe Web resources in a collaborative manner has experienced rising popularity among Web users in recent years. The product of such activity is given the name folksonomy, which can be considered as a scheme of organizing information in the users' own way. This research work attempts to analyze tripartite graphs – graphs involving users, tags and resources – of folksonomies and discuss how these elements acquire their semantics through their associations with other elements, a process we call mutual contextualization. By studying such process, we try to identify solutions to problems such as tag disambiguation, retrieving documents of similar topics and discovering communities of users. This paper describes the basis of the research work, mentions work done so far and outlines future plans.

## 1 Introduction

The use of freely-chosen words or phrases called tags to classify Web resources has experienced rising popularity among Web users in recent years. Through the use of tags, Web users come to share and organize their favourite Web resources in different collaborative tagging systems, such as del.icio.us<sup>1</sup> and Flickr<sup>2</sup>. The result of such social and collaborative tagging is given the name folksonomy [1].

Collaborative tagging and folksonomy possess a number of advantages which account for its popularity [4]. These include its simplicity as well as the freedom enjoyed by the users to choose their own tags. Folksonomies have also been considered to be a possible solution to construct ontologies which can be used in the Semantic Web [7]. However, some limitations and shortcomings, such as the problem of ambiguous meanings of tags and the existence of synonyms, also affect its effectiveness to organize resources on the Web [4].

In this research, we focus on analysis of tripartite graphs of folksonomies, graphs which involves the three basic elements of collaborative tagging, namely users, tags and resources. We investigate how these elements come to acquire

---

<sup>1</sup> <http://del.icio.us/>

<sup>2</sup> <http://www.flickr.com/>

their own semantics through their associations with other elements in the graphs, a process which we call mutual contextualization. The study aims to acquire a more thorough understanding of the characteristics of folksonomies, and to devise mechanisms that allow applications to better utilize the power of folksonomies. Methods of network analysis and community discovery will be employed to reveal the semantics hidden in the seemingly chaotic folksonomy data.

## 2 Mutual Contextualization in Folksonomies

The power of folksonomies lies in the interrelations between the three elements. A tag is only a symbol if it is not assigned to some Web resources. A tag is also ambiguous without a user's own interpretation of its meaning. Similarly, a user, though identified by its username, is characterized by the tags it uses and the resources it tags. Finally, a document is given semantics because tags act as a form of annotation. Hence, it is obvious that each of these elements in a folksonomy would be meaningless, or at least ambiguous in meaning, if they are considered independently. In other words, the semantics of one element depends on the context given by the other two, or all, elements that are related to it.

To further understand this kind of mutual contextualization, we examine each of the three elements in a folksonomy in details. For more specific discussions, we assume that the Web resources involved are all Web documents. In addition, we define the data in a social tagging system, a folksonomy, as follows.

**Definition 1.** *A folksonomy  $F$  is a tuple  $F = (U, T, D, A)$ , where  $U$  is a set of users,  $T$  is a set of tags,  $D$  is a set of Web documents, and  $A \subseteq U \times T \times D$  is a set of annotations.*

As we have mentioned earlier, the three elements forming the tripartite graph of a social tagging system are users, tags and documents (resources). The tripartite graph can be reduced into a bipartite graph if, for example, we focus on a particular tag and extract only the users and documents that are affiliated to it. Since there are three types of elements, there can be three different types of bipartite graphs. This is similar to those introduced by Mika [7], but we distinguish our method by only concentrating on one instance of an element instead of all the instances of the same element.

### 2.1 Users

By focusing on a single user  $u$ , we obtain a bipartite graph  $TD_u$  defined as follows:

$$TD_u = \langle T \times D, E_{td} \rangle, E_{td} = \{(t, d) | (u, t, d) \in A\}$$

In other words, an edge exists between a tag and a document if the user has assigned the tag to the document. The graph can be represented in matrix form, which we denote as  $\mathbf{X} = \{x_{ij}\}$ ,  $x_{ij} = 1$  if there is an edge connecting  $t_i$  and  $d_j$ . The bipartite graph represented by the matrix can be folded into two one-mode networks [7]. We denote one of them as  $\mathbf{P} = \mathbf{X}\mathbf{X}'$ , and another as  $\mathbf{R} = \mathbf{X}'\mathbf{X}$ .

$\mathbf{P}$  represents a kind of semantic network which shows the associations between different tags. It should be noted that this is unlike the light-weight ontology mentioned in [7], as it only involves tags used by a single user. In other words, this is the personal vocabulary used by the particular user, which can be called a personomy [5].

The matrix  $\mathbf{R}$  represents the personal repository of the user. Links between documents are weighted by the number of tags that have been assigned to both documents. Thus, documents having higher weights on the links between them are those that are considered by the particular user as more related.

By studying and comparing the documents and tags associated with different users, we can have a better understanding of the vocabularies as well as the interests or expertise [6] of different users. This will lead to more efficient and effective ways for matching user interests and discovering community of users.

## 2.2 Tags

By using a similar method as described above, we can obtain a bipartite graph  $UD_t$  regarding to a particular tag  $t$ :

$$UD_t = \langle U \times D, E_{ud} \rangle, E_{ud} = \{(u, d) | (u, t, d) \in A\}$$

In words, an edge exists between a user and a document if the user has assigned the tag  $t$  to the document. The graph can once again be represented in matrix form, which we denote as  $\mathbf{Y} = \{y_{ij}\}$ ,  $y_{ij} = 1$  if there is an edge connecting  $u_i$  and  $d_j$ . This bipartite graph can be folded into two one-mode networks, which we denote as  $\mathbf{S} = \mathbf{Y}\mathbf{Y}'$ , and  $\mathbf{C} = \mathbf{Y}'\mathbf{Y}$ .

The matrix  $\mathbf{S}$  shows the affiliation between the users who have used the tag  $t$ , weighted by the number of documents to which they have both assigned the tag. Since a tag can be used to represent different concepts (such as *sf* for *San Francisco* or *Science Fiction*), and a document provides the necessary content to identify the contextual meaning of the tag, this network is likely to connect users who use the tag for the same meaning.

$\mathbf{C}$  can be considered as another angle of viewing the issue of polysemous or homonymous tags. Thus, with the edges weighted by the number of users who have assigned tag  $t$  to both documents, this network is likely to connect documents which are related to the same sense of the given tag.

The existence of ambiguous tags and synonymous tags is a major problem in folksonomies. These tags affect the precision of describing resources as well as that of retrieval of relevant documents. By examining the documents and users associated with a tag, we are actually trying to interpret the meaning of a tag by placing it in the right context. Thus, this will probably provide a possible way to understanding the semantics of tags in a folksonomy.

## 2.3 Documents

Finally, a bipartite graph  $UT_d$  can also be obtained by considering a particular document  $d$ . The graph is defined as follows:

$$UT_d = \langle U \times T, E_{ut} \rangle, E_{ut} = \{(u, t) | (u, t, d) \in A\}$$



In words, an edge exists between a user and a tag if the user has assigned the tag to the document  $d$ . The graph can be represented in matrix form, which we denote as  $\mathbf{Z} = \{z_{ij}\}$ ,  $z_{ij} = 1$  if there is an edge connecting  $u_i$  and  $t_j$ . Like in the cases of a single user and a single tag, this bipartite graph can be folded into two one-mode networks, which we denote as  $\mathbf{M} = \mathbf{ZZ}'$ , and  $\mathbf{V} = \mathbf{Y}'\mathbf{Y}$ .

The matrix  $\mathbf{M}$  represent a network in which users are connected based on the documents commonly tagged by them. Since a document may provide more than one kind of information, and users do not interpret the content from a single perspective, the tags assigned by different users will be different, although tags related to the main theme of the document are likely to be used by most users. Hence, users linked to each other by edges of higher weights in this network are more likely to share a common perspective, or are more likely to concern a particular piece of information provided by the document.

On the other hand, the matrix  $\mathbf{V}$  represents a network in which tags are connected and weighted by the number of users who have assigned them to the document. Hence, the network is likely to reveal the different perspective of the users from which they interpret the content of the document.

The bipartite graph obtained by focusing on a single document gives us information on how different users interpret the content of the document, and which aspects in the document do they focus on. A better understanding of these issues will surely benefit applications such as Web page recommender systems.

From the above discussions, we can see that different relations between the users, the tags and the documents in a folksonomy will affect how a single user, tag or document is interpreted in the system. Each of these elements provide an appropriate context such that the semantics of the elements can be understood without ambiguity.

### 3 Research Progress and Future Plan

The current stage of this research work focuses on bipartite graphs obtained by examining a single tag. As noted by several other authors [14,8], there are actually quite a lot of ambiguous tags in existing folksonomies. One of the objectives of studying the bipartite graphs is to discover effective methods for tag meaning disambiguation. To illustrate the approach consider the following example. The tag *sf* is observed to be used to represent two distinctive concepts, namely “science fiction” and “San Francisco.” The result shows that users and documents which are associated with the same meanings of the tag tend to be clustered with each other. This shows that it is possible to disambiguate tags by studying the bipartite graphs. We are currently carrying out research on how different community discovering techniques (e.g. [2,3]) can be employed to develop an algorithm for tag disambiguation.

The future plan of this research involves the following steps. Firstly, after the study of tag disambiguation, we will move on to study bipartite graphs obtained by examining a single document or a single user. We will investigate how similar techniques of network analysis can be applied to these graphs to discover useful

information regarding interests and expertise of users, communities of users with shared interests, and clusters of documents with similar topics. In the end we hope to develop a unified theory of the process of mutual contextualization in folksonomies, which gives a thorough picture of the semantics of the elements involved. Finally, we will investigate how these results can be applied to applications such as social network analysis, Web page recommender systems and construction of ontologies.

## 4 Conclusion

This paper describes a research work on the analysis of tripartite graphs of folksonomies. In particular, we study the process of mutual contextualization between the three basic elements in a folksonomy, in the hope of understanding the semantics of these elements. Promising results have been obtained in preliminary studies of disambiguation of tags, and plans for future works have been drawn. We believe this work will contribute to a better understanding of folksonomies and benefit different applications on the Web.

## References

1. Adam, M.: Folksonomies - cooperative classification and communication through shared metadata (2004), <http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html>
2. Clauset, A.: Finding local community structure in networks. *Physical Review E* 72(026132) (2005)
3. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA* 99(7821) (2002)
4. Golder, S., Huberman, B.A.: Usage patterns of collaborative tagging systems. *Journal of Information Science* 32(2), 198–208 (2006)
5. Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: Information retrieval in folksonomies: Search and ranking. In: Sure, Y., Domingue, J. (eds.) *ESWC 2006*. LNCS, vol. 4011, pp. 411–426. Springer, Heidelberg (2006)
6. John, A., Seligmann, D.: Collaborative tagging and expertise in the enterprise. In: *WWW 2006: Proceedings of the 15th international conference on World Wide Web*, ACM Press, New York (2006)
7. Mika, P.: Ontologies are us: A unified model of social networks and semantics. In: *International Semantic Web Conference*, pp. 522–536 (2005)
8. Wu, X., Zhang, L., Yu, Y.: Exploring social annotations for the semantic web. In: *WWW 2006. Proceedings of the 15th international conference on World Wide Web*, pp. 417–426. ACM Press, New York (2006)

# Author Index

- Abel, Fabian 1, 871  
Alani, Harith 708  
Ambite, José Luis 15  
Angeletou, Sofia 423  
Anutariya, Chutiporn 566  
Arndt, Richard 30  
Aroyo, Lora 879, 887  
Auer, Sören 722
- Bachmann-Gmür, Reto 537  
Bao, Shenghua 680  
Bellekens, Pieter 887  
Benson, Edward 830  
Bernstein, Abraham 281, 295  
Besana, Paolo 44  
Bizer, Christian 722  
Bloehdorn, Stephan 58  
Blomqvist, Eva 911  
Bobillo, Fernando 72  
Bontcheva, Kalina 142, 843  
Bouffier, Amanda 916
- Cardillo, Elena 764  
Ceausu, Valentina 736  
Chi, Mingmin 623  
Christophides, Vassilis 637  
Cimiano, Philipp 523  
Cimino, James 816  
Cuenca Grau, Bernardo 183  
Cunningham, Hamish 142  
Cyganiak, Richard 722
- d'Aquin, Mathieu 423  
Daga, Enrico 764  
Darlington, John 708  
Davis, Brian 142  
de Bruijn, Jos 86  
De Coi, Juri Luca 1  
Dean, Mike 830  
Decker, Stefan 197, 211  
Delbru, Renaud 552  
Delgado, Miguel 72  
Denker, Grit 750  
Desprès, Sylvie 736  
Di Noia, Tommaso 395
- Di Sciascio, Eugenio 395  
Dolby, Julian 100, 816  
Domingue, John 802  
Donini, Francesco M. 395  
Dorner, Eric 830  
Dupplaw, David 708
- Elenius, Daniel 750  
Erling, Orri 537
- Falconer, Sean M. 114  
Fan, James 100  
Fokoue, Achille 100, 816  
Ford, Reginald 750  
Frank, Mischa 871  
Friedrich, Gerhard 466  
Fu, Linyun 128  
Funk, Adam 142, 843
- Gangemi, Aldo 764  
Geleijnse, Gijs 156  
Gibbins, Nicholas 966  
Giunchiglia, Fausto 623, 666  
Gliozzo, Alfio 764  
Golbreich, Christine 169  
Gómez-Pérez, Asunción 508  
Gómez-Romero, Juan 72  
Gorgels, Peter 879  
Gracia, Jorge 423  
Groza, Tudor 197
- Haase, Peter 508  
Halaschek-Wiener, Christian 183  
Handschuh, Siegfried 142, 197, 802  
Hardman, Lynda 30  
Harth, Andreas 211  
Heath, Tom 802, 895  
Henze, Nicola 1, 871  
Hepp, Martin 480  
Heymans, Stijn 86  
Hitzler, Pascal 310  
Hoffmann, Jörg 480  
Hogan, Aidan 211  
Horridge, Matthew 169, 267  
Horrocks, Ian 169

- Houben, Geert-Jan 887  
 Hu, Wei 225  
 Huang, Xuanjing 623  
 Hunter, Anthony 381  
 Huynh, David F. 239, 903  
 Hyvönen, Eero 409, 778  
  
 Idreos, Stratos 324  
 Isaac, Antoine 253  
 Ives, Zachary 722  
  
 Jaffri, Afraz 921  
 Jannach, Dietmar 466  
 Johnson, Mark 750  
 Ju, Qi 623  
  
 Kalyanpur, Aditya 100, 267, 816  
 Kapoor, Dipsy 15  
 Kaptein, Annelies 887  
 Karger, David R. 239, 903  
 Kaufmann, Esther 281  
 Kazakov, Yevgeny 183  
 Kershenbaum, Aaron 100, 816  
 Kiefer, Christoph 295  
 Kim, Hong-Gee 353  
 Kobilarov, Georgi 722  
 Koesling, Arne Wolf 1  
 Kolas, Dave 792  
 Korst, Jan 156  
 Koubarakis, Manolis 324  
 Kozeruk, Olga 466  
 Krause, Daniel 1, 871  
 Kröttsch, Markus 310  
 Krummenacher, Reto 926  
  
 Lambrix, Patrick 494  
 Lehmann, Jens 722  
 Lewen, Holger 508  
 Liarou, Erietta 324  
 Liu, QiaoLing 652  
 Liu, Shengping 857  
  
 Ma, Li 100, 816  
 Mao, Ming 931  
 Martin, David 340, 750  
 Maynard, Diana 843  
 Meinel, Christoph 367  
 Miller, Robert C. 239, 903  
 Möller, Knud 197, 802  
 Morbidoni, Christian 537  
  
 Morris, Chuck 830  
 Motik, Boris 169  
 Motta, Enrico 423, 895  
 Muñoz-García, Óscar 508  
 Murdock, William 100  
  
 Namgoong, Hyun 353  
 Noll, Michael G. 367  
  
 O'Hara, Kieron 708  
 Olmedilla, Daniel 1  
 Oren, Eyal 552  
  
 Pan, Jeff Z. 666  
 Pan, Wei 623  
 Pan, Yue 652, 857  
 Paolucci, Massimo 340  
 Parsia, Bijan 267, 595  
 Patel, Chintan 816  
 Plappert, Daniel 871  
 Predoiu, Livia 936  
 Presutti, Valentina 764  
  
 Qi, Guilin 381  
 Qiu, Zhaoming 857  
 Qu, Yuzhong 225  
  
 Ragone, Azzurra 395, 941  
 Reinecke, Katharina 946  
 Ressler, James 830  
 Robertson, Dave 44  
 Rudolph, Sebastian 310, 523  
 Ruotsalo, Tuukka 409  
 Rutledge, Lloyd 879  
  
 Sabou, Marta 423  
 Saggion, Horacio 843  
 Salvati, Alberto 764  
 Schlobach, Stefan 253  
 Schmidt, Renate A. 438  
 Schonberg, Edith 816  
 Self, Troy 792  
 Serafini, Luciano 452  
 Shadbolt, Nigel 708, 966  
 Shchekotykhin, Kostyantyn 466  
 Shearer, Rob 169  
 Sheridan, John 708  
 Siehdnel, Patrick 871  
 Siorpaes, Katharina 951  
 Sirin, Evren 267

- Srinivas, Kavitha 100, 816  
 Staab, Steffen 30  
 Stash, Natalia 879  
 Stocker, Markus 295  
 Stollberg, Michael 480  
 Storey, Margaret-Anne 114  
 Studer, Rudi 508, 523  
 Sun, Lei 623  
 Suominen, Osma 778  
 Sure, York 58  
 Šváb, Ondřej 956
- Tablan, Valentin 142  
 Tamin, Andrei 452  
 Tan, He 494  
 Tishkovsky, Dmitry 438  
 Topor, Rodney 666  
 Tran, Thanh 508, 523  
 Trif, Diana 197  
 Troiani, Gianluca 764  
 Troncy, Raphaël 30  
 Tullo, Carol 708  
 Tummarello, Giovanni 537, 552  
 Tzitzikas, Yannis 637
- Umbrich, Jürgen 211  
 Ungrangsi, Rachanee 566
- Vacura, Miroslav 30  
 van der Meij, Lourens 253  
 van der Sluijs, Kees 887  
 Viljanen, Kim 778
- Wagner, Matthias 340  
 Wang, Chong 694  
 Wang, Gang 580  
 Wang, Haofen 128, 652, 694  
 Wang, Kewen 666  
 Wang, Shenghui 253  
 Wang, Taowei David 595  
 Wang, Yang 128  
 Wang, Ying 961  
 Wang, Yiwen 879  
 Welty, Christopher 100  
 Wu, Xian 680  
 Wuwongse, Vilas 566
- Xie, Guotong 857  
 Xiong, Miao 694
- Yamauchi, Takashi 609  
 Yang, Yang 857  
 Yeung, Ching-man Au 966  
 Yu, Yong 128, 580, 652, 680, 694
- Zaihrayeu, Ilya 623  
 Zeginis, Dimitris 637  
 Zhang, Huajie 128  
 Zhang, Jie 652  
 Zhang, Lei 652  
 Zhao, Yuting 666  
 Zhou, Mianwei 680  
 Zhou, Qi 694  
 Zhou, Xiongzhi 857  
 Zhu, Haiping 128, 580