Philippe Cudré-Mauroux Jeff Heflin
Evren Sirin Tania Tudorache
Jérôme Euzenat Manfred Hauswirth
Josiane Xavier Parreira Jim Hendler
Guus Schreiber Abraham Bernstein
Eva Blomqvist (Eds.)

LNCS 7650

The Semantic Web – ISWC 2012

11th International Semantic Web Conference
Boston, MA, USA, November 2012
Proceedings, Part II

Springer

Lecture Notes in Computer Science 7650

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Philippe Cudré-Mauroux Jeff Heflin
Evren Sirin Tania Tudorache
Jérôme Euzenat Manfred Hauswirth
Josiane Xavier Parreira Jim Hendler
Guus Schreiber Abraham Bernstein
Eva Blomqvist (Eds.)

The Semantic Web – ISWC 2012

11th International Semantic Web Conference
Boston, MA, USA, November 11-15, 2012
Proceedings, Part II

Springer

Volume Editors

Philippe Cudré-Mauroux; University of Fribourg, Switzerland; pcm@unifr.ch

Jeff Heflin; Lehigh University, USA; heflin@cse.lehigh.edu

Evren Sirin; Clark & Parsia, USA; evren@clarkparsia.com

Tania Tudorache; Stanford University, USA; tudorache@stanford.edu

Jérôme Euzenat; INRIA & LIG, France; jerome.euzenat@inria.fr

Manfred Hauswirth; DERI, NUI Galway, Ireland; manfred.hauswirth@deri.org

Josiane Xavier Parreira; DERI, NUI Galway, Ireland; josiane.parreira@deri.org

Jim Hendler; Rensselaer Polytechnic Institute (RPI), USA; hendler@cs.rpi.edu

Guus Schreiber; VU University Amsterdam, The Netherlands; guus.schreiber@vu.nl

Abraham Bernstein; University of Zurich, Switzerland; bernstein@ifi.uzh.ch

Eva Blomqvist; Linköping University, Sweden; eva.blomqvist@liu.se

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-35172-3 e-ISBN 978-3-642-35173-0
DOI 10.1007/978-3-642-35173-0
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012952320

CR Subject Classification (1998): I.2.3-4, H.2.3-5, H.2.1, H.2.8, I.2.6, H.3.4-5, D.2.12, D.3.2, I.7.2, H.5.3-4, J.1

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

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The Semantic Web has come a long way. What started as a vision of a machine-readable Web over ten years ago now consists of a vibrant community of researchers, practitioners, enthusiast, companies, and, finally, users. Topics that were once cutting-edge research have now arrived in the mainstream and have even become part of political agendas. The sharing of public information in the form of linked data has become a major argument for the transparency of administrations, and institutions around the globe are putting their data online. Companies from various sectors such as the BBC, Google, IBM, or *The New York Times* release products that are based on Semantic Web technologies. Against all prophecies of failure, the Semantic Web is flourishing.

The International Semantic Web Conference is the premier forum for Semantic Web research, where cutting-edge scientific results and technological innovations are presented, where problems and solutions are discussed, and where the future of this vision is being developed. It brings together specialists in fields such as artificial intelligence, databases, social networks, distributed computing, Web engineering, information systems, human–computer interaction, natural language processing, and the social sciences for tutorials, workshops, presentations, keynotes, and sufficient time to have detailed discussions.

This volume contains the main proceedings of the 11th International Semantic Web Conference (ISWC 2012), which was held in Boston, USA, in November 2012. Even though the economic times are anything but easy we received tremendous response to our calls for papers from a truly international community of both researchers and practitioners. Every paper was thoroughly evaluated following practices appropriate for its track and its evaluation measure. The breadth and scope of the papers finally selected for inclusion in this volume speak to the quality of the conference and to the contributions made by researchers whose work is presented in these proceedings. As such, we were all honored and proud that we were invited to serve the community in the stewardship of this edition of ISWC.

The Research Track of the conference attracted 186 submissions, 41 of which were accepted, resulting in a 22% acceptance rate. Each paper received at least three, and sometimes as many as five, reviews from members of the Program Committee. After the first round of reviews, authors had the opportunity to submit a rebuttal, leading to further discussions among the reviewers, a meta-review and a recommendation from a member of the Senior Program Committee (SPC). The SPC held a 10-hour virtual meeting in order to select the final set of accepted papers, paying special attention to papers that were borderline or had at least one recommendation for acceptance. In many cases, additional last-minute reviews were sought out to better inform the SPC's decision.

As the Semantic Web develops, we find a changing variety of subjects that emerge. This year the keywords of accepted papers were distributed as follows (frequency in parentheses): knowledge representation and reasoning (13), querying the Semantic Web and database technologies (10), ontology engineering (7), machine learning and information extraction (7), data mining and analysis (6), ontology mapping (6), linked data (5), languages, tools and methodologies (4), interacting with Semantic Web data (4), instance mapping (4), evaluation (4), social and emergent semantics (4), cleaning, assurance, and provenance (4), search and information retrieval (3), federated/distributed systems (3), scalable systems (3), Semantic Web services (3), exploiting the social Web (3), knowledge acquisition (2), natural language processing (2), query languages (2), uncertainty (2), modeling users and contexts (2), semantic streams and sensors (2), ontology learning (1), user interfaces (1), mashing up data and processes (1), trust, privacy and security (1), and personalized access (1).

This edition of the International Semantic Web Conference marks the introduction of the Evaluations and Experiments Track. The goal of this track is to consolidate research material and to gain new scientific insights and results by providing a place for in-depth experimental studies of significant scale. It aims at promoting experimental evaluations in Semantic Web/Linked Data domains where availability of experimental datasets and reproducibility of experiments are highly important.

The Evaluations and Experiments track received 35 submissions from all areas of the Semantic Web: including reasoning, querying, searching, matching, and annotating. Papers were of two main categories, namely, evaluation (comparing several approaches to a problem) and corpus analysis. To our surprise, testing a hypothesis through an experiment was not explicitly considered. We also received very few papers aiming at reproducing existing experiments. Eight papers were accepted, corresponding to a 23% acceptance rate. Each paper was reviewed by at least three members of the Program Committee paying special attention to the reproducibility criteria. In spite of the limited number of accepted papers, they address a large range of areas, such as linked stream data, federated query processing, tag recommendation, entity summarization, and OWL reasoning.

The Semantic Web In-Use Track received 77 submissions. At least three members of the In-Use Program Committee provided reviews for each paper. Seventeen papers were accepted – a 22% acceptance rate. The large number of submissions this year demonstrated the increasingly diverse breadth of applications of Semantic Web technologies in practice. The papers demonstrated how semantic technologies are applied in a variety of domains, including eGovernment, smart cities, biomedicine, or question answering. Several papers dealt with applying reasoning for a variety of use cases, while others dealt with streaming data and

processing complex events. A number of infrastructure papers contributed to the state of art for Linked Open Data and for querying large data sets. Very exciting application papers demonstrated how semantic technologies are applied in diverse ways, starting from using linked data in mobile environments to employing full-fledged artificial intelligence methods in real-time use cases.

The Doctoral Consortium is a key event at the ISWC conference. PhD students in the Semantic Web field get an opportunity to present their thesis proposals and to interact with leading academic and industrial scientists in the field, who act as their mentors. Out of 21 submissions to the Doctoral Consortium, six were accepted as for presentation at the conference. For discussion at the special Consortium-only session on 12 November, nine additional proposals were selected. The Doctoral Consortium day is organized as a highly interactive event, in which students present their proposals and receive extensive feedback and comments from the mentors as well as from their peers.

A unique aspect of the ISWC conference is the Semantic Web Challenge, now in its 10th year, with the goal of demonstrating practical progress toward achieving the vision of the Semantic Web. Organized this year by Diana Maynard and Andreas Harth, the competition enables practitioners and scientists to showcase leading-edge real-world applications of Semantic Web technology.

The keynote talks given by leading scientists or practitioners in their field further enriched the ISWC program. Thomas W. Malone, the director of the Center for Collective Intelligence at the Massachusetts Institute of Technology, discussed the phenomenon of collective intelligence and how it interrelates with the Semantic Web. Jeanne Holm, an evangelist for data.gov, discussed the changing global landscape of data sharing and the role the Semantic Web is playing in it. Mark Musen, a professor of medicine of the Stanford Center for Biomedical Informatics Research, discussed how the fundamental challenges of AI are still with us and await embracing to fulfill the vision of the Semantic Web. And last but not least, Nigel Shadbolt, Deputy Head of the School of Electronics and Computer Science at the University of Southampton, gave a lively dinner talk.

As in previous ISWC editions, the conference included an extensive Tutorial and Workshop program. Claudia d'Amato and Thomas Scharrenbach, the Chairs of this track, selected a stellar and diverse collection of 9 tutorials and 18 workshops, where the only problem that the participants faced was which of the many exciting workshops to attend. Workshops and tutorials were chosen on the ground of two different but complementary criteria: maintaining the history of the most promising, challenging, and highly attended workshops such as the Ontology Matching Workshop, the Consuming Linked Data Workshop, the Ontology Patterns Workshop, or the Uncertainty Reasoning for the Semantic Web Workshop and highlighting the attention on new, challenging, and visionary research trends as testified by the Programming the Semantic Web Workshop, the Semantic Sensor Network Workshop, the Web of Linked Entities Workshop, the Semantic Technologies Applied to Biomedical Informatics and Individualized Medicine Workshop, the Web of Data for E-Commerce Tutorial, the Machine Learning for Linked Data Tutorial, the Linked Data for Development Tutorial, or

the Financial Information Management using the Semantic Web Tutorial. Also, particular attention was dedicated to the heterogeneity and scalability issues and related aspects, which explains the choice of the Workshop on Large and Heterogeneous Data and Quantitative Formalization in the Semantic Web, the Tutorial on RDF Query Processing in the Cloud, and the Tutorial on Scalable Semantic Processing of Hudge, Distributed Real-Time Streams.

We would like to thank Birte Glimm and David Huynh for organizing a lively Poster and Demo Session. As in 2011, the Posters and Demos were introduced in a Minute Madness Session, where every presenter got 60 seconds to provide a teaser for their poster or demo.

Ivan Herman, Tom Heath, and Tim Berners-Lee coordinated a top-flight Industry Track where end-users of Semantic Web and Linked Data technologies shared their "warts and all" experiences with the research community. The track attracted presentations from enterprises of all scales, from startups through to software, hardware, and retail giants such as Oracle, Cray, Cisco, EMC, and BestBuy.

We are indebted to Eva Blomqvist, our Proceedings Chair, who provided invaluable support in compiling the volume that you now hold in your hands (or see on your screen) and exhibited superhuman patience in allowing the other Chairs to stretch deadlines to the absolute limits. Many thanks to Jen Golbeck, the Fellowship Chair, for securing and managing the distribution of student travel grants and thus helping students who might not have otherwise attended the conference to come to Boston. Peter Mika and David Wood were tireless in their work as Sponsorship Chairs, knocking on every conceivable virtual "door" and ensuring an unprecedented level of sponsorship this year. We are especially grateful to all the sponsors for their generosity.

As has been the case in the past, ISWC 2012 also contributed to the linked data cloud by providing semantically annotated data about many aspects of the conference. This contribution would not have been possible without the efforts of Li Ding our Metadata Chair. Oshani Seneviratne, our Publicity Chair, was tirelessly twittering and sending old-fashioned (and highly appreciated) announcements on the mailing lists, creating far more lively "buzz" than ISWC ever had.

Our very special thanks go to the Local Organization Team, led by Lalana Kagal. She did an outstanding job of handling local arrangements, thinking of every potential complication way before it arose, often doing things when members of the Organizing Committee were only beginning to think about asking for them. We managed to juggle so many balls, that some of us were dizzy just looking at it. Special thanks go to the staff of MIT conference services – Cathi Di Iulio Levine, Nicole Silva, Lynne Alyson Lenker, and Eva Cabone – for their enormous resourcefulness, foresight, and anticipation of the conference needs and requirements. Also many thanks for the designers at the University of Zurich Multimedia and e-Learning Services, who provided all the design work often going beyond the call of any duty.

Finally, we would like to thank all members of the ISWC Organizing Committee not only for handling their tracks superbly, but also for their wider contribution to the collaborative decision-making process in organizing the conference.

September 2012

Philippe Cudré-Mauroux
Jeff Heflin
Program Committee Co-chairs
Research Track

Manfred Hauswirth
Josie Xavier Perreira
Jérôme Euzenat
Program Committee Co-chairs
Evaluation Track

Tania Tudorache
Evren Sirin
Program Committee Co-chairs
Semantic Web In-Use Track

Claudia d'Amato
Thomas Scharrenbach
Workshop & Tutorials Co-chairs

Jim Hendler
Guus Schreiber
Doctoral Consortium Chairs

Abraham Bernstein
Conference Chair

Conference Organization

Organizing Committee

General Chair

Abraham Bernstein	University of Zurich, Switzerland

Local Chair

Lalana Kagal	Massachusetts Institute of Technology, USA

Program Chairs–Research Track

Philippe Cudré-Mauroux	University of Fribourg, Switzerland
Jeff Heflin	Lehigh University, USA

In-Use Track Chairs

Tania Tudorache	Stanford University, USA
Evren Sirin	Clark&Parsia, USA

Evaluations and Experiments Track Chairs

Jérôme Euzenat	INRIA & LIG, France
Manfred Hauswirth	DERI, NUI Galway, Ireland
Josiane Xavier Parreira	DERI, NUI Galway, Ireland

Industry Track Chair

Tim Berners-Lee	W3C, USA
Tom Heath	Talis, UK
Ivan Herman	W3C & CWI, The Netherlands

Doctoral Consortium Chairs

Guus Schreiber	VU University Amsterdam, The Netherlands
Jim Hendler	RPI, USA

Fellowship Chair

Jen Golbeck	University of Maryland, USA

Posters and Demos Chairs

David Huynh	Google, USA
Birte Glimm	University of Ulm, Germany

Workshops and Tutorials Chairs

Claudia d'Amato	University of Bari, Italy
Thomas Scharrenbach	University of Zurich, Switzerland

Semantic Web Challenge Chairs

Diana Maynard	University of Sheffield, UK
Andreas Harth	KIT, Germany

Sponsorship Chairs

Peter Mika	Yahoo! Research, Spain
David Wood	3 Round Stones Inc., USA

Metadata Chair

Li Ding	Qualcomm, USA

Webmaster

Stéphane Corlosquet	MGH, USA

Proceedings Chair

Eva Blomqvist	Linköping University, Sweden

Publicity Chair

Oshani Seneviratne	MIT, USA

Senior Program Committee–Research

Harith Alani	The Open University, UK
Lora Aroyo	Vrije Universiteit Amsterdam, The Netherlands
Jérôme Euzenat	INRIA, France
Andreas Harth	Karlsruhe Institute of Technology, Germany
Pascal Hitzler	Wright State University, USA
Ian Horrocks	University of Oxford, UK
Lalana Kagal	MIT, USA
David Karger	MIT, USA
Manolis Koubarakis	National and Kapodistrian University of Athens, Greece
Diana Maynard	University of Sheffield, UK
Deborah L. McGuinness	RPI, USA
Natasha F. Noy	Stanford University, USA
Peter Patel-Schneider	Nuance Communications, USA

Axel Polleres	Siemens AG Österreich, Austria & DERI, Galway, Ireland
Heiner Stuckenschmidt	University of Mannheim, Germany
Jie Tang	Tsinghua University, China
Hai Zhuge	Chinese Academy of Sciences, China

Program Committee–Research

Karl Aberer
José Luis Ambite
Grigoris Antoniou
Kemafor Anyanwu
Sören Auer
Jie Bao
Michael Benedikt
Sonia Bergamaschi
Christian Bizer
Eva Blomqvist
Paolo Bouquet
John Breslin
Dan Brickley
Paul Buitelaar
Diego Calvanese
Huajun Chen
Key-Sun Choi
Benoit Christophe
Vassilis Christophides
Fabio Ciravegna
Lin Clark
Oscar Corcho
Isabel Cruz
Bernardo Cuenca Grau
Carlo Curino
Richard Cyganiak
Claudia d'Amato
Mathieu d'Aquin
David De Roure
Mike Dean
Stefan Decker
Renaud Delbru
Gianluca Demartini
Ian Dickinson
Stefan Dietze
Li Ding
John Domingue
Miriam Fernandez
Tim Finin
Giorgos Flouris
Achille Fokoue
Enrico Franconi
Bo Fu
Avigdor Gal
Fabien Gandon
Aldo Gangemi
Birte Glimm
Jennifer Golbeck
Alasdair J. G. Gray
Paul Groth
Tudor Groza
Michael Gruninger
Nicola Guarino
Christophe Guéret
Gerd Gröner
Volker Haarslev
Harry Halpin
Siegfried Handschuh
Michael Hausenblas
Sandro Hawke
Tom Heath
Martin Hepp
Michiel Hildebrand
Aidan Hogan
Vasant Honavar
Matthew Horridge
Andreas Hotho
Wei Hu
Arantza Illarramendi
Krzysztof Janowicz
Yannis Kalfoglou
Anastasios Kementsietsidis
Brian Kettler
Pavel Klinov

Craig Knoblock
Adila A. Krisnadhi
Markus Krötzsch
Freddy Lecue
Alexander Löser
Christopher Matheus
Yutaka Matsuo
Peter Mika
Prasenjit Mitra
Luc Moreau
Boris Motik
Yohei Murakami
Wolfgang Nejdl
Yuan Ni
Jeff Pan
Bijan Parsia
Alexandre Passant
Paulo Pinheiro Da Silva
Dimitris Plexousakis
Alexandra Poulovassilis
Valentina Presutti
Abir Qasem
Guilin Qi
Yuzhong Qu
Marie-Christine Rousset
Sebastian Rudolph
Marta Sabou
Satya Sahoo
Manuel Salvadores
Bernhard Schandl
Francois Scharffe
Stefan Schlobach
Daniel Schwabe
Juan F. Sequeda
Luciano Serafini
Amit Sheth
Pavel Shvaiko
Wolf Siberski
Elena Simperl
Michael Sintek
Spiros Skiadopoulos
Kavitha Srinivas
Steffen Staab
Umberto Straccia
Rudi Studer
York Sure-Vetter
Valentina Tamma
Letizia Tanca
Kerry Taylor
Sergio Tessaris
Philippe Thiran
Thanh Tran
Raphael Troncy
Christos Tryfonopoulos
Tania Tudorache
Frank Van Harmelen
Maria Esther Vidal
Haofen Wang
Josiane Xavier Parreira
Lina Zhou
Esteban Zimanyi
Antoine Zimmermann

Additional Reviewers–Research

Alan Akbik
Pramod Anantharam
Manuel Atencia
George Baryannis
Domenico Beneventano
Jürgen Bock
Georgeta Bordea
Mohamed Boukhebouze
Stephane Campinas
Michele Catasta
Pierre-Antoine Champin
Francesco Corcoglioniti
Evangelia Daskalaki
Brian Davis
Chiara Di Francescomarino
Jianfeng Du
Songyun Duan
Jinan El-Hachem
Philipp Frischmuth
Sidan Gao
Andrés García-Silva
Paolo Garza

Mouzhi Ge
Francesco Guerra
Dalkandura Gunaratna
Robert Gwadera
Lushan Han
Holmer Hemsen
Daniel M. Herzig
Aidan Hogan
Robert Isele
Nophadol Jekjantuk
Pavan Kapanipathi
Ernesto Jimenez-Ruiz
Pavan Kapanipathi
Yevgeny Kazakov
Hyeongsik Kim
Ilianna Kollia
Haridimos Kondylakis
Kostis Kyzirakos
John Liagouris
Chang Liu
Diana Maynard
Varish Mulwad
Rammohan Narendula
Nadeschda Nikitina
Charalampos Nikolaou
Shohei Ohsawa
Alexandra Olteanu
Fabrizio Orlandi
Mirko Orsini
Matteo Palmonari
Evan Patton
Sean Policarpio
Behrang Qasemizadeh
Emanuele Rabosio
Paraskevi Raftopoulou
Padmashree Ravindra
Timothy Redmond
Yuan Ren
Martin Rezk
Benedicto Rodriguez-Castro
Mariano Rodriguez-Muro
Simon Scheider
Michael Schneider
Jennifer Sleeman
Serena Sorrentino
Jie Tang
Vinhtuan Thai
Joerg Unbehauen
Larysa Visengeriyeva
Jens Wissmann
Zhixian Yan
Jun Zhao
Yuting Zhao
Dmitriy Zheleznyakov

Program Committee–Semantic Web In-Use

Jans Aasman
Anupriya Ankolekar
Sören Auer
Christian Bizer
Cecile Bothorel
Ivan Cantador
Vinay Chaudhri
Oscar Corcho
Gianluca Correndo
Mathieu D'Aquin
Mike Dean
Leigh Dodds
Federico Michele Facca
Hugh Glaser
Mark Greaves
Tudor Groza
Peter Haase
Armin Haller
Siegfried Handschuh
Michael Hausenblas
Tom Heath
Ivan Herman
Pascal Hitzler
Rinke Hoekstra
Matthew Horridge
Wei Hu
Krzysztof Janowicz
Pavel Klinov
Matthias Klusch
Jens Lehmann

Yuan-Fang Li
Thorsten Liebig
Adrian Mocan
Lyndon Nixon
Vit Novacek
Natasha Noy
Massimo Paolucci
Alexandre Passant
Carlos Pedrinaci
Héctor Pérez-Urbina
Yves Raimond
Cartic Ramakrishnan
Marco Rospocher
Matthew Rowe
Marta Sabou
Manuel Salvadores
Patrick Sinclair
Milan Stankovic
Nenad Stojanovic
Markus Strohmaier
Jamie Taylor
Giovanni Tummarello
Michael Uschold
Willem Robert van Hage
Jacco Van Ossenbruggen
Holger Wache
Kewen Wang
Shenghui Wang
Zhe Wu
Fouad Zablith
Amal Zouaq

Additional Reviewers–In-Use

Michelle Cheatham
Brian Davis
Laura Dragan
Cosmin Dumitru
Christian Huetter
Prateek Jain
Christoph Lange
Akshay Maan
Marc Schaaf
Michael Schmidt
William Smith
Claus Stadler
Jesse Wang
Amapali Zaveri

Program Committee–Evaluations and Experiments

Denilson Barbosa
Payam Barnaghi
Oscar Corcho
Thierry Declerck
Renaud Delbru
Jérôme Euzenat
Raúl García-Castro
Asunción Gómez-Pérez
Siegfried Handschuh
Andreas Harth
Alois Haselboeck
Manfred Hauswirth
Conor Hayes
Ryutaro Ichise
Antoine Isaac
Manolis Koubarakis
Thomas Krennwallner
Christian Meilicke
Marta Sabou
Sherif Sakr
Kai-Uwe Sattler
Francois Scharffe
Ondrej Svab-Zamazal
Martin Theobald
Christos Tryfonopoulos
Giovanni Tummarello
Willem Robert Van Hage
Maria Esther Vidal
Josiane Xavier Parreira
Shenghui Wang

Additional Reviewers–Evaluations and Experiments

Nitish Aggarwal
Stephane Campinas
Davide Ceolin
Brian Davis
Laura Dragan
Aidan Hogan
Ioana Hulpus
Vit Novacek
Magdalena Ortiz
Paraskevi Raftopoulou
Patrik Schneider
Guohui Xiao

Program Committee–Doctoral Consortium

Harith Alani
Oscar Gorcho
Fausto Guinchiliglia
Ian Horrocks
Diana Maynard
Natasha Noy
Elena Simperl
Steffen Staab
Frank van Harmelen

Sponsors

Gold Sponsors

AI Journal
fluid Operations
Oracle
systap
Yahoo! Research

Silver Sponsors

IBM
Pitney Bowes

Driving Innovation with Open Data and Interoperability

(Keynote Talk)

Jeanne Holm

Data.gov, U.S. General Services Administration
jeanne.m.holm@jpl.nasa.gov

Abstract

Data.gov, a flagship open government project from the US government, opens and shares data to improve government efficiency and drive innovation. Sharing such data allows us to make rich comparisons that could never be made before and helps us to better understand the data and support decision making. The adoption of open linked data, vocabularies and ontologies, the work of the W3C, and semantic technologies is helping to drive Data.gov and US data forward. This session will help us to better understand the changing global landscape of data sharing and the role the semantic web is playing in it.

This session highlights specific data sharing examples of solving mission problems from NASA, the White House, and many other governments agencies and citizen innovators.

The Semantic Web and Collective Intelligence
(Keynote Talk)

Thomas Malone

MIT Sloan School of Management, Cambridge, MA
malone@mit.edu

Abstract

The original vision of the Semantic Web was to encode semantic content on the web in a form with which machines can reason. But in the last few years, we've seen many new Internet-based applications (such as Wikipedia, Linux, and prediction markets) where the key reasoning is done, not by machines, but by large groups of people.

This talk will show how a relatively small set of design patterns can help understand a wide variety of these examples. Each design pattern is useful in different conditions, and the patterns can be combined in different ways to create different kinds of collective intelligence. Building on this foundation, the talk will consider how the Semantic Web might contribute to–and benefit from–these more human-intensive forms of collective intelligence.

Tackling Climate Change: Unfinished Business from the Last "Winter"

(Keynote Talk)

Mark A. Musen

Stanford Center for Biomedical Informatics Research, Stanford University
musen@stanford.edu

Abstract

In the 1990s, as the World Wide Web became not only world wide but also dense and ubiquitous, workers in the artificial intelligence community were drawn to the possibility that the Web could provide the foundation for a new kind of AI. Having survived the AI Winter of the 1980s, the opportunities that they saw in the largest, most interconnected computing platform imaginable were obviously compelling. With the subsequent success of the Semantic Web, however, our community seems to have stopped talking about many of the issues that researchers believe led to the AI Winter in the first place: the cognitive challenges in debugging and maintaining complex systems, the drift in the meanings ascribed to symbols, the situated nature of knowledge, the fundamental difficulty of creating robust models. These challenges are still with us; we cannot wish them away with appeals to the open-world assumption or to the law of large numbers. Embracing these challenges will allow us to expand the scope of our science and our practice, and will help to bring us closer to the ultimate vision of the Semantic Web.

Table of Contents – Part II

In-Use Track

Evaluations and Experiments Track

Doctoral Consortium – Long Papers

Doctoral Consortium – Short Papers

Table of Contents – Part I

Research Track

Managing the Life-Cycle of Linked Data with the LOD2 Stack

Sören Auer, Lorenz Bühmann, Christian Dirschl, Orri Erling, Michael Hausenblas, Robert Isele, Jens Lehmann, Michael Martin, Pablo N. Mendes, Bert van Nuffelen, Claus Stadler, Sebastian Tramp, and Hugh Williams

LOD2 Project*, c/o Universität Leipzig, Postfach 100920, 04009 Leipzig, Germany
http://lod2.eu

Abstract. The LOD2 Stack is an integrated distribution of aligned tools which support the whole life cycle of Linked Data from extraction, authoring/creation via enrichment, interlinking, fusing to maintenance. The LOD2 Stack comprises new and substantially extended existing tools from the LOD2 project partners and third parties. The stack is designed to be versatile; for all functionality we define clear interfaces, which enable the plugging in of alternative third-party implementations. The architecture of the LOD2 Stack is based on three pillars: **(1)** Software integration and deployment using the Debian packaging system. **(2)** Use of a central SPARQL endpoint and standardized vocabularies for knowledge base access and integration between the different tools of the LOD2 Stack. **(3)** Integration of the LOD2 Stack user interfaces based on REST enabled Web Applications. These three pillars comprise the methodological and technological framework for integrating the very heterogeneous LOD2 Stack components into a consistent framework. In this article we describe these pillars in more detail and give an overview of the individual LOD2 Stack components. The article also includes a description of a real-world usage scenario in the publishing domain.

Keywords: Linked Data, application integration, provenance.

1 Introduction

The LOD2 Stack is an integrated distribution of aligned tools which support the whole life cycle of Linked Data from extraction, authoring/creation via enrichment, interlinking, fusing to maintenance. The LOD2 Stack comprises new and substantially extended existing tools from the LOD2 partners and third parties. The major components of the LOD2 Stack are open-source in order to facilitate wide deployment and scale to knowledge bases with billions of triples and large numbers of concurrent users. Through an agile, iterative software development

* The research leading to these results has received funding under the European Commission's Seventh Framework Programme from ICT grant agreement LOD2, no. 257943. Corresponding author is auer@uni-leipzig.de.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 1–16, 2012.

approach, we aim at ensuring that the stack fulfills a broad set of user requirements and thus facilitates the transition to a Web of Data. The stack is designed to be versatile; for all functionality we define clear interfaces, which enable the plugging in of alternative third-party implementations. We also plan a stack configurer, which enables potential users to create their own personalized version of the LOD2 Stack, which contains only those functions relevant for their usage scenario. In order to fulfill these requirements, the architecture of the LOD2 Stack is based on three pillars:

- *Software integration and deployment using the Debian packaging system.* The Debian packaging system is one of the most widely used packaging and deployment infrastructures and facilitates packaging and integration as well as maintenance of dependencies between the various LOD2 Stack components. Using the Debian system also allows to facilitate the deployment of the LOD2 Stack on individual servers, cloud or virtualization infrastructures.
- *Use of a central SPARQL endpoint and standardized vocabularies for knowledge base access and integration between different tools.* All components of the LOD2 Stack access this central knowledge base repository and write their findings back to it. In order for other tools to make sense out of the output of a certain component, it is important to define vocabularies for each stage of the Linked Data life-cycle.
- *Integration of the LOD2 Stack user interfaces based on REST enabled Web Applications.* Currently, the user interfaces of the various tools are technologically and methodologically quite heterogeneous. We do not resolve this heterogeneity, since each tool's UI is specifically tailored for a certain purpose. Instead, we develop a common entry point for accessing the LOD2 Stack UI, which then forwards a user to a specific UI component provided by a certain tool in order to complete a certain task.

These three pillars comprise the methodological and technological framework for integrating the very heterogeneous LOD2 Stack components into a consistent framework. This article is structured as follows: After briefly introducing the linked data life-cycle in Section 2, we describe these pillars in more detail (Section 3). We describe a real-world use-case for the Stack in Section 4 and conclude with an outlook on future work in Section 5.

2 The Linked Data Life-Cycle

The different stages of the Linked Data life-cycle (depicted in Figure 1) include:

Storage. RDF Data Management is still more challenging than relational Data Management. We aim to close this performance gap by employing column-store technology, dynamic query optimization, adaptive caching of joins, optimized graph processing and cluster/cloud scalability.

Authoring. LOD2 facilitates the authoring of rich semantic knowledge bases, by leveraging Semantic Wiki technology, the WYSIWYM paradigm (What You See Is What You Mean) and distributed social, semantic collaboration and networking techniques.

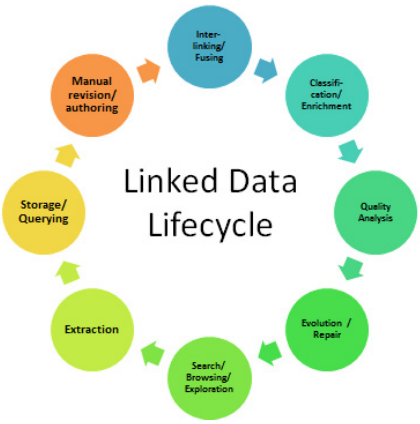


Fig. 1. Stages of the Linked Data life-cycle supported by the LOD2 Stack

Interlinking. Creating and maintaining links in a (semi-)automated fashion is still a major challenge and crucial for establishing coherence and facilitating data integration. We seek linking approaches yielding high precision and recall, which configure themselves automatically or with end-user feedback.

Classification. Linked Data on the Web is mainly raw instance data. For data integration, fusion, search and many other applications, however, we need this raw instance data to be linked and integrated with upper level ontologies.

Quality. The quality of content on the Data Web varies, as the quality of content on the document web varies. LOD2 develops techniques for assessing quality based on characteristics such as provenance, context, coverage or structure.

Evolution/Repair. Data on the Web is dynamic. We need to facilitate the evolution of data while keeping things stable. Changes and modifications to knowledge bases, vocabularies and ontologies should be transparent and observable. LOD2 also develops methods to spot problems in knowledge bases and to automatically suggest repair strategies.

Search/Browsing/Exploration. For many users, the Data Web is still invisible below the surface. LOD2 develops search, browsing, exploration and visualization techniques for different kinds of Linked Data (i.e. spatial, temporal, statistical), which make the Data Web sensible for real users.

These life-cycle stages, however, should not be tackled in isolation, but by investigating methods which facilitate a mutual fertilization of approaches developed to solve these challenges. Examples for such mutual fertilization between approaches include:

- The detection of mappings on the schema level, for example, will directly affect instance level matching and vice versa.
- Ontology schema mismatches between knowledge bases can be compensated for by learning which concepts of one are equivalent to which concepts of another knowledge base.

- Feedback and input from end users (e.g. regarding instance or schema level mappings) can be taken as training input (i.e. as positive or negative examples) for machine learning techniques in order to perform inductive reasoning on larger knowledge bases, whose results can again be assessed by end users for iterative refinement.
- Semantically enriched knowledge bases improve the detection of inconsistencies and modelling problems, which in turn results in benefits for interlinking, fusion, and classification.
- The querying performance of RDF data management directly affects all other components, and the nature of queries issued by the components affects RDF data management.

As a result of such interdependence, we should pursue the establishment of an improvement cycle for knowledge bases on the Data Web. The improvement of a knowledge base with regard to one aspect (e.g. a new alignment with another interlinking hub) triggers a number of possible further improvements (e.g. additional instance matches).

The challenge is to develop techniques which allow exploitation of these mutual fertilizations in the distributed medium Web of Data. One possibility is that various algorithms make use of shared vocabularies for publishing results of mapping, merging, repair or enrichment steps. After one service published its new findings in one of these commonly understood vocabularies, notification mechanisms (such as *Semantic Pingback* [15]) can notify relevant other services (which subscribed to updates for this particular data domain), or the original data publisher, that new improvement suggestions are available. Given proper management of provenance information, improvement suggestions can later (after acceptance by the publisher) become part of the original dataset.

3 Integrating Heterogeneous Tools into the LOD2 Stack

The LOD2 Stack serves two main purposes. Firstly, the aim is to ease the distribution and installation of tools and software components that support the Linked Data publication cycle. As a distribution platform, we have chosen the well established Debian packaging format. The second aim is to smoothen the information flow between the different components to enhance the end-user experience by a more harmonized look-and-feel.

3.1 Deployment Management Leveraging Debian Packaging

In the *Debian package management system* [12], software is distributed in architecture-specific binary packages and architecture-independent source code packages. A Debian software package comprises two types of content: **(1)** control information (incl. metadata) of that package, and **(2)** the software itself.

The control information of a Debian package will be indexed and merged together with all other control information from other packages available for the system. This information consists of descriptions and attributes for:

(a) The software itself (e.g. licenses, repository links, name, tagline, ...),
(b) Its relation to other packages (dependencies and recommendations),
(c) The authors of the software (name, email, home pages), and
(d) The deployment process (where to install, pre and post install instructions).

The most important part of this control information is its relations to other software. This allows the deployment of a complete stack of software with one action. The following dependency relations are commonly used in the control information:

Depends: This declares an absolute dependency. A package will not be configured unless all of the packages listed in its Depends field have been correctly configured. The Depends field should be used if the depended-on package is required for the depending package to provide a significant amount of functionality. The Depends field should also be used if the install instructions require the package to be present in order to run.

Recommends: This declares a strong, but not absolute, dependency. The Recommends field should list packages that would be found together with this one in all but unusual installations.

Suggests: This is used to declare that one package may be more useful with one or more others. Using this field tells the packaging system and the user that the listed packages are related to this one and can perhaps enhance its usefulness, but that installing this one without them is perfectly reasonable.

Enhances: This field is similar to Suggests but works in the opposite direction. It is used to declare that a package can enhance the functionality of another package.

Conflicts: When one binary package declares a conflict with another using a Conflicts field, dpkg will refuse to allow them to be installed on the system at the same time. If one package is to be installed, the other must be removed first.

All of these relations may restrict their applicability to particular versions of each named package (the relations allowed are <<, <=, =, >= and >>). This is useful in forcing the upgrade of a complete software stack. In addition to this, dependency relations can be set to a list of alternative packages. In such a case, if any one of the alternative packages is installed, that part of the dependency is considered to be satisfied. This is useful if the software depends on a specific functionality on the system instead of a concrete package (e.g. a mail server or a web server). Another use case of alternative lists are meta-packages. A meta-package is a package which does not contain any files or data to be installed. Instead, it has dependencies on other (lists of) packages.

Example of meta-packaging: OntoWiki. To build an appropriate package structure, the first step is to inspect the manual deployment of the software, its variants and the dependencies of these variants. *OntoWiki* is a browser-based collaboration and exploration tool as well as an application for linked data publication. There are two clusters of dependencies: the runtime environment and

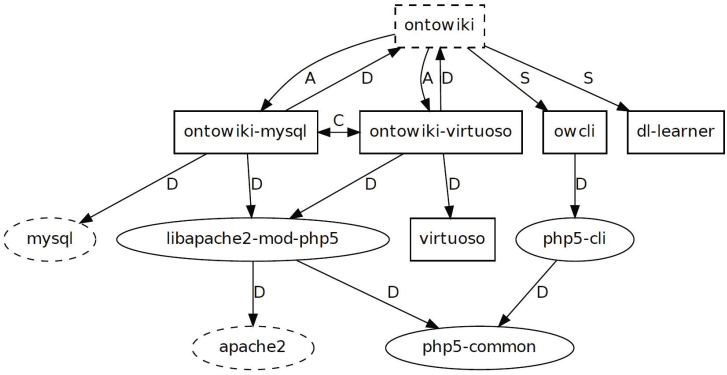


Fig. 2. Example DEB-package dependency tree (OntoWiki). Some explanation: Boxes are part of the LOD2 Stack, Ellipses are part of the Debian/Ubuntu base system, Dashed forms are meta-packages, Relations: Depends (D), Depends alternative list (A), Conflicts (C) and Suggests (S).

the backend. Since OntoWiki is developed in the scripting language *PHP*, it's architecture-independent but needs a web server running PHP. More specifically, OntoWiki needs PHP5 running as an Apache 2 module. OntoWiki currently supports two different back-ends which can be used to store and query RDF data: *Virtuoso* and MySQL. Virtuoso is also part of the LOD2 Stack while *MySQL* is a standard package in all Debian-based systems. In addition to OntoWiki, the user can use the OntoWiki command line client *owcli* and the *DL-Learner* from the LOD2 Stack to enhance its functionality.

The dependency tree (depicted in Figure 2) is far from being complete, since every component also depends on libraries and additional software which is omitted here. Given this background information, we can start to plan the packaging. We assume that users either use MySQL or Virtuoso as a backend on a server, so the first decision is to split this functionality into two packages: `ontowiki-mysql` and `ontowiki-virtuoso`. These two packages are abstracted by the meta-package `ontowiki`, which requires either `ontowiki-mysql` or `ontowiki-virtuoso`, and which can be used by other LOD2 Stack packages to require OntoWiki. Since both the MySQL backend and the Virtuoso backend version use the same system resources, we need to declare them as conflicting packages.

Installing the LOD2 Stack. The LOD2 Stack is available at `http://stack.lod2.eu`. Our reference OS is *Ubuntu 12.04 LTS*. Most of the components run on old or more recent releases without a problem. In general, deploying the LOD2 software stack or parts of it is simple. There are only two steps to execute in order to install LOD2 Stack software: **(1)** Add the LOD2 Stack package repository to the system's repository list and update the repository index. **(2)** Install desired software packages by using a graphical or text-based package management application. The procedure can be executed using graphical front-ends like Synaptic. Using the command line the LOD2 Stack installation is performed as follows[1]:

[1] More information, tutorials and FAQs can be found at `http://wiki.lod2.eu`.

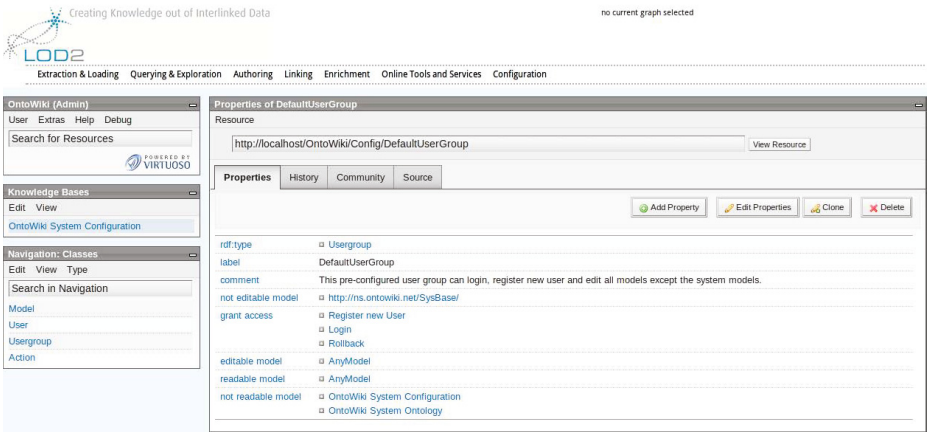


Fig. 3. The LOD2 Stack demonstrator is an interface to explore and use all the different stack tools in an integrated way

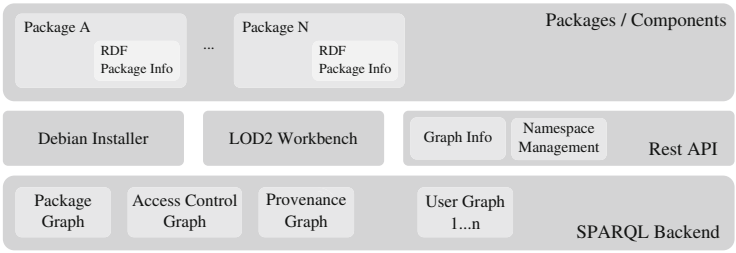


Fig. 4. Basic architecture of a local LOD2 Stack

```
# download the repository package
wget http://stack.lod2.eu/lod2repository_current_all.deb
# install the repository package
sudo dpkg -i lod2repository_current_all.deb
# update the repository database
sudo apt-get update

# lod2demo is a meta root package that installs all LOD2 components
sudo apt-get lod2demo
```

3.2 Data Integration Based on SPARQL, WebID and Vocabularies

The basic architecture of a local LOD2 Stack installation is depicted in Figure 4. All components in the LOD2 Stack act upon RDF data and are able to communicate via SPARQL with the central system-wide RDF quad store (i.e. SPARQL backend). This quad store (Openlink Virtuoso) manages user graphs (knowledge bases) as well as a set of specific system graphs where the behaviour and status of the overall system is described. The following system graphs are currently used:

Package Graph: In addition to the standard Debian package content, each LOD2 Stack package consists of a RDF package info which contains:

- The basic package description, e.g. labels, dates, maintainer info (this is basically DOAP data and redundant to the classic Debian control file)
- Pointers to the place where the application is available (e.g. the menu entry in the LOD2 Stack workbench)
- A list of capabilities of the packed software (e.g. resource linking, RDB extraction). These capabilities are part of a controlled vocabulary. The terms are used as pointers for provenance logging, access control definition and a future capability browser of the LOD2 workbench.

Upon installation, the package info is automatically added to the package graph to allow the workbench / demonstrator to query which applications are available and what is the user able to do with them.

Access Control Graph: This system graph is related to WebID[2] authentication and describes which users are able to use which capabilities and have access to which graphs. The default state of this graph contains no restrictions, but could be used to restrict certain WebIDs to specific capabilities. Currently, only OntoWiki takes this graph into account and the access control definition is based on the WebAccessControl schema[3].

Provenance Graph: Each software package is able to log system wide provenance information to reflect the evolution of a certain knowledge base. Different ontologies are developed for that use-case. To keep the context of the LOD2 Stack, we use the controlled capability vocabulary as reference points.

In addition to the SPARQL protocol endpoint, application packages can use a set of APIs which allow queries and manipulation currently not available with SPARQL alone (e.g. fetching graph information and manipulating namespaces). Two authorized administration tools are allowed to manipulate the package and access control graphs:

- The Debian system installer application automatically adds and removes package descriptions during install / upgrade and remove operations.
- The LOD2 Workbench (Demonstrator) is able to manipulate the access control graph.

All other packages are able to use the APIs as well as to create, update and delete knowledge bases. Table 1 gives an overview on the current LOD2 Stack components in alphabetic order. In the following, we give a brief summary on some of the most important packages.

Apache Stanbol [4] is an open source modular software stack and reusable set of components (exposed via RESTful interfaces) for semantic content management. One application is to extend traditional content management systems

[2] http://www.w3.org/wiki/WebID
[3] http://www.w3.org/wiki/WebAccessControl
[4] Apache Stanbol is a result of the IKS project http://iks-project.eu.

Table 1. Overview on LOD2 Stack components

Tool	Category	Supported Stages
Apache Stanbol [3]	NLP Middleware Server	Extraction
CubeViz	Statistical data browser	Visualization
DBpedia Spotlight [10]	Entity Recognition and Linking	Extraction
D2RQ [2]	RDB2RDF Mapping	Extraction
DL-Learner [6,7,9]	Machine Learning in OWL	Schema Enrichment
OntoWiki [1]	Generic Data Wiki	Authoring, Exploration
ORE [8]	Knowledge Base Debugging	Repair
PoolParty [14]	SKOS Taxonomy Editor	Authoring, Exploration
SemMap	Spatial data browser	Browsing, Exploration
Sig.ma EE [16]	Data Browser	Search, Exploration
Sieve [11]	Quality Assessment and Fusion	Quality, Repair
SILK [5]	Linking Workbench	Interlinking
LIMES [13]	Linking Workbench	Interlinking
Virtuoso [4]	Hybrid RDBMS/Graph Column Store	Storage / Querying
Valiant	XML2RDF transformation	Extraction

with (internal or external) semantic services. In the LOD2 Stack, Apache Stanbol can be used for NLP services which rely on the stack internal knowledge bases, such as named entity recognition and text classification.

CubeViz [5] is a widget for visualizing statistical data being published adhering to the DataCube vocabulary. CubeViz analyses the DataCube data structure definitions and generates menus for selecting dimensions, slices and measures to be visualized employing different diagram types (e.g. bar, pie, line charts).

DBpedia Spotlight is a tool for automatically annotating mentions of DBpedia resources in text, providing a solution for linking unstructured information sources to the Linked Open Data cloud through DBpedia. DBpedia Spotlight recognizes that names of concepts or entities have been mentioned (e.g. "Michael Jordan"), and subsequently matches these names to unique identifiers (e.g. `dbp:Michael_I._Jordan`, the machine learning professor or `dbp:Michael_Jordan`, the basketball player). Besides common entity classes (i.e. People, Locations, Organisations), Spotlight also spots concepts from the 320 classes in the DBpedia Ontology. It is integrated with Apache Stanbol and can thus be combined with other NLP tools.

D2RQ [6] is a system for integrating relational databases (RDBMS) in RDF-based data integration workflows. D2RQ allows querying a non-RDF database using SPARQL, accessing the content of the database as Linked Data over the Web, creating custom dumps of the database in RDF formats for loading into an RDF store, and accessing information in a non-RDF database using the *Apache Jena API*. D2RQ powers hundreds of public Linked Data sites around the Web. D2RQ supports RDBMSs from all major vendors. Cur-

[5] `http://aksw.org/Projects/CubeViz`
[6] `http://d2rq.org/`

rent work focuses on extending D2RQ and making it compliant with *W3C's R2RML* and *Direct Mapping* standards[7].

DL-Learner framework provides a set of (semi-)supervised machine learning algorithms for knowledge bases, specifically for OWL ontologies and SPARQL endpoints. The goal of DL-Learner is to support knowledge engineers in constructing knowledge and learning about the data they created, by generating axioms and concept descriptions which fit the underlying data.

ORE (Ontology Repair and Enrichment) allows knowledge engineers to improve an OWL ontology or SPARQL endpoint backed knowledge base by fixing logical errors and making suggestions for adding further axioms to it. ORE uses state-of-the-art methods to detect errors and highlight the most likely sources for the problems. To harmonise schema and data in the knowledge base, algorithms of the DL-Learner framework are integrated.

OntoWiki is a *PHP5* / Zend-based Semantic Web application for collaborative knowledge base editing. It facilitates the visual presentation of a knowledge base as an information map, with different views of instance data. It enables intuitive authoring of semantic content, with an inline editing mode for editing RDF content, similar to WYSIWYG for text documents.

PoolParty is a tool to create and maintain multilingual *SKOS* (Simple Knowledge Organisation System) thesauri, aiming to be easy to use for people without a Semantic Web background or special technical skills. PoolParty is written in Java and uses the *SAIL API*, whereby it can be utilized with various triple stores. Thesaurus management itself (viewing, creating and editing SKOS concepts and their relationships) can be performed in an *AJAX* front-end based on the Yahoo User Interface (YUI) library.

SemMap [8] allows to visualize knowledge bases having a spatial dimension. It provides a map view for selecting and exploring a spatial area and a faceted navigation for filtering objects of a particular type or with particular properties in the selected area. The SemMap visualization widget is implemented in JavaScript and interacts with the triple store solely via SPARQL.

Sig.ma EE (Sig.ma Enterprise Edition) is a standalone, deployable, customizable version of the on-the-fly Web of Data mashup creation interface Sig.ma. Sig.ma EE is deployed as a web application and performs on-the-fly data integration from both local LOD2 Stack data sources and remote services.

Sieve includes a quality assessment module and a data fusion module. The quality of Linked Data sources on the Web is varies widely, as values may be out of date, incomplete or incorrect. Moreover, data sources may provide conflicting values for a single real-world object. Sieve's quality assessment module leverages user-selected metadata as quality indicators to produce quality assessment scores through user-configured scoring functions. The data fusion module is able to use quality scores in order to perform user-configurable conflict resolution tasks.

Silk is a link discovery framework that supports data publishers in setting explicit links between two datasets. Using the declarative Silk - Link

[7] http://www.w3.org/2001/sw/rdb2rdf/

[8] http://aksw.org/Projects/SemMap

Fig. 5. The visualization widgets CubeViz (statistic) and SemMap (spatial data)

Specification Language (Silk-LSL), developers can specify which types of RDF links should be discovered between data sources as well as which conditions data items must fulfill in order to be interlinked. These link conditions may combine various similarity metrics and can take the graph around a data item into account using an RDF path language.

LIMES is a link discovery framework for the Web of Data. It implements time-efficient approaches for large-scale link discovery based on the characteristics of metric spaces. It is easily configurable via a web interface. It can also be downloaded as a standalone tool for carrying out link discovery locally. In addition, the Colanut GUI implements mechanisms for the automatic suggestion of link configurations.

Virtuoso is an enterprise grade multi-model data server. It delivers a platform agnostic solution for data management, access, and integration. Virtuoso provides a fast quad store with *SPARQL* endpoint and WebID support.

Valiant is an extraction/transformation tool that uses *XSLT* to transform XML documents into RDF. The tool can access data from the file system or a Web-DAV repository. It outputs the resulting RDF to disk, WebDAV or directly to an RDF store. For each input document a new graph is created.

3.3 REST Integration of User Interfaces

Many of the components come with their own user interface. For example, the Silk Workbench is a user interface for the Silk linking engine (cf. **??**). This workbench supports the creation of linking specifications, executing them and improving them using the feedback from the user on the created links. With the OntoWiki linked data browsing and authoring tool, a user can browse and update information in a knowledge base (cf. Figure 3). By using both tools together, the user gains the ability to study the input sources' content structure and to create links between them.

Many stack components request similar information from the user. For example, selecting the graph of interest. To provide the end-user the feeling of a harmonized single application, we develop supportive REST-based WebAPIs. These APIs offer a common application view of the LOD2 Stack. The more tools support this API, the more harmonized and integrated the end-user experience gets. Currently, the LOD2 Stack WebAPI consists of:

- *Graph management*: The set of graphs is not easy to maintain. SPARQL does not support retrieval of all graphs. The only possible query which selects all graphs that have at least one triple is performance wise quite costly: `SELECT DISTINCT ?g WHERE GRAPH ?g ?s ?p ?o` The WebAPI also standardizes some meta information like *being a system graph*. When LOD2 Stack components use this common graph management WebAPI, the end-user obtains a uniform look-and-feel with respect to graph management.
- *Prefix management*: To make RDF resources more readable, prefixes are used to abbreviate URI namespaces. Typically, each application manages its own namespace mapping. Using this REST API, a central namespace mapping is maintained, thus producing consistency among stack components. The end-user is freed from updating the individual component mappings. Moreover, an update in one component is immediately available to another.

In addition to creating supportive REST-based APIs, the LOD2 Stack encourages component owners to open up their components using REST based WebAPIs. For example, the semantic-spatial browser, a UI tool that visualizes RDF data containing geospatial information on a map, is entirely configurable by parameters encoded within its invocation URL. Similarly other visualization and exploration widgets (such as the CubeViz statistical data visualization) can directly interact with the SPARQL endpoint (cf. Figure 5). This makes it easy to integrate into (third party) applications into the stack.

3.4 Enlarging the LOD Volume and Facilitating Dataset Discovery

All the above effort to improve the software support for Linked Data publishing must have an effect in the daily practice of Linked Data publishing. For that reason the LOD2 project collaborates with data providers. One such data provider is the LOD2 partner *Wolters Kluwer*. Other collaborations include the *European Commission DG INFSO*, with its *Digital Agenda Scoreboard*, and the *National Statistical Office of Serbia*. Both improvements to the tools and data are returned to the public.

To ease reuse, data must be easily found. Therefore, we enhanced the data portal *CKAN*[9]. This portal is being extended to allow SPARQL queries over the repository. With that, we close the whole Linked Open Data cycle. Data is accessed and transformed into RDF using extraction and storage components, then it is augmented and interlinked with other data sources (found through online data portals) and finally the newly created dataset is published as a new datasource on the web, announcing itself to the world via a data portal and ready to be used. Both, announcement as well as discovery via CKAN is an integral part of the LOD2 Stack.

4 Facilitating Data Flows at a Global Publisher

Wolters Kluwer is a global knowledge and information service provider with more than 19.000 employees worldwide and core competencies in the legal, tax and

[9] `http://ckan.net`

business domains. Wolters Kluwer offers information for the professional in any format including folio, software and services.

The Linked Data life-cycle mirrored to publishing business. The steps described in the life-cycle highly resemble traditional workflow steps in a publishing house. Therefore, conceptually adopting this life-cycle for the publishing business is very reasonable. In traditional publishing, the focus is mainly on textual information, starting from the authoring process up to layout and printing. Metadata has recently become prominent with increasing use of digital libraries with sophisticated search functionalities. This shift of scope is still ongoing, and new company internal processes and skills must be developed and implemented. Since the LOD2 Stack tools are, by definition, (meta-)data oriented and highly standard compliant, they have great potential to fill the gap between very efficient content processing and very flexible and powerful metadata management. As a first step, we have focused on the following parts of the life-cycle:

Extraction: Usually, the content in a publisher's house is stored in XML, and stored in the same file as the text. Therefore, the extraction of the metadata is an important step in the overall process.
Storage: All metadata must be accessible to all tools exploiting it.
Authoring: Human editors must be able to code their knowledge domain in an easy way, which also means that features for proper maintenance and development must be in place.
Interlinking: When the publishing industry talks about "linking", it is mainly referring to hyperlinks in text. The capabilities here are different, meaning linking different knowledge sources in order to create a semantic network.
Search/Browse/Exploration: Allowing editorial staff to interact with the data is key in an operational environment. The gap between technological representation and semantic human interpretation must be bridged by using metaphors and on-the-fly mapping between URIs and human-readable labels.

Based on these core tasks, tools from the LOD2 Stack were selected in order to fulfill the respective requirements. This resulted in a working prototype called *Pebbles*, using and integrating the following tools:

- Virtuoso triple store for storage of the triples, along with its WebDAV environment for storage of the accompanying XML source files.
- PoolParty for maintaining all the controlled vocabularies, including domain taxonomies and thesauri. This environment is also used for publishing Linked Data. Initially, labour law thesaurus and a court thesaurus have been made publicly available under a Creative Commons license.
- SILK framework for mapping between the Wolters Kluwer knowledge bases and external sources like DBpedia or the EUROVOC thesaurus.
- OntoWiki as the user interface for human end users. Features for taxonomy browsing and filtering, but also for metadata management like adding or deleting or changing an instance, are used. There is also a connector to the original XML file, so that the basic text information can be displayed in parallel, rendered in HTML.

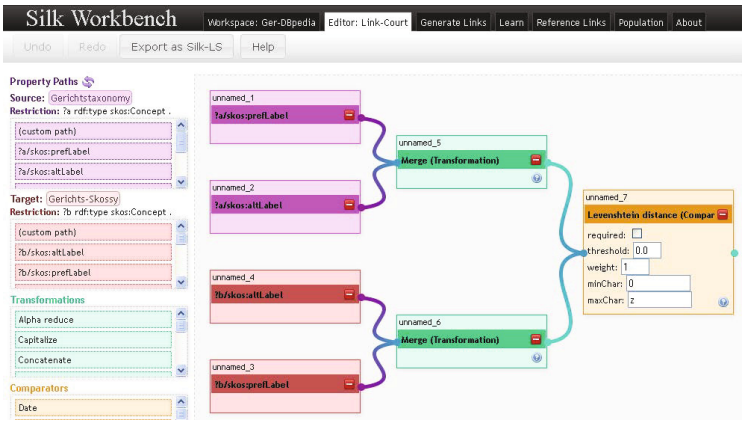


Fig. 6. Silk workbench with loaded linking specification to link law courts from different datasets

– Valiant and VEnrich (a wrapper around PoolParty Extractor) tools to make the extraction process efficient and performant.

The resulting prototype (in fact an LOD2 Stack adoption) is currently being evaluated by the operational editorial team, to assess its appropriateness as a basis for an internal knowledge base within Wolters Kluwer Germany.

Our evaluation included, in particular, a dataset extracted from approximately 800.000 semi-structured XML documents from the German legal domain. From these documents, 46.651.884 facts have been extracted. This process was run in batch mode on a server with 8 GB memory and takes approximately 4 hours. The data is strongly linked within itself (the documents refer to other documents in the document set). The exploration of linking to external public sources has started. One of the few German sources available is the German DBpedia. Using Silk, we were able to discover links to all German laws.

Lessons learned. Technical and project issues we encountered were:

– The creation of the extraction rules to create for each XML document an associated RDF graph is an interactive and iterative process which requires to combine technical knowledge and domain knowledge. In that process deciding what will be the "controlled" terms (elements that are under some editorial control, for example, exact names of courts of Germany) are utmost important. These should best be represented as rdf:Resource elements with a stable URI. The process applied is depicted in figure Y. It shows a non-trivial feedback loop where many people are involved.
– To support the process new software had to be developed: Valiant for XSLT batch processing; A webservice for the PoolParty extractor to map produced RDF to controlled vocabularies; an adaptation of OntoWiki supporting nested RDF graphs.

- We also faced the challenge of finding ways for bridging the gap between technical partners mainly coming from an academic world and the requirements of an industrial partner.
- Modeling and representing information from the legal domain in Europe is extremely challenging due to the diversity and variety throughout Europe.

Opportunities beyond local business. The technology at hand has three main characteristics, which make it a candidate for usage in a global environment: it is about semantics, it is about connecting these semantics and it is about referring to official international standards. Imagine a global publisher with businesses in more than 40 countries worldwide. In order to offer cross-country offerings in different languages, there are three approaches possible:

- Approaching each and every country individually and collecting the data on an individual basis.
- Introducing a global content repository.
- Introducing a semantic layer on top of every local repository for automatic extraction and bundling of data.

The first approach needs many effective and controlled workflows in place, in order to be effective and efficient. The second approach is very expensive and time consuming to implement. The third approach seems to be the best compromise and most sustainable solution and is thus favored at Wolters Kluwer.

Summary and next steps. The LOD2 Stack serves the needs of a publishing use case in many respects: The LOD life cycle reflects very well the tasks a publishing house has to perform; getting a grip on semantics will be a key skill of professionals and therefore also of their service providers; the wide usage of standards ensures the flexibility of not being locked in to a specific tool or vendor. The Pebbles prototype has shown that some of the tools in the stack are mature enough, so that they can be used in an industrial environment.

Currently, we are looking at expanding the usage of the LOD2 Stack in our use case, mainly by including NLP tools in order to address the classification/enrichment step of the life-cycle. If this is successful, a lot of additional added-value to our data and therefore to our products can automatically be exploited. In addition, we want to use the publishing capabilities of *PoolParty* in order to publish our data as LOD data and therefore get in touch with the developer community. We seek a win-win situation, where our data is more widely used and requirements for additional or completely new data can be met by us.

5 Conclusion and Outlook

In this article we presented the LOD2 Stack, the result of a large-scale effort to provide technological support for the life-cycle of Linked Data. We deem this a first step in a larger research and development agenda, where derivatives of the LOD2 Stack are employed to create corporate enterprise knowledge hubs withing the Intranets of large companies such as the publisher Wolters Kluwer. In

order to realize our vision, we aim to further strengthen the light-weight REST-API based integration between the components of the stack. The overall stack architecture and guidelines can also serve as a blue-print for similar software stacks in other areas. For the next iterations of the LOD2 Stack, we plan to increase tool coverage and to include more 3rd party developed tools.

References

1. Auer, S., Dietzold, S., Riechert, T.: OntoWiki – A Tool for Social, Semantic Collaboration. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 736–749. Springer, Heidelberg (2006)
2. Bizer, C.: D2r map - a database to rdf mapping language. In: WWW (Posters) (2003)
3. Christ, F., Nagel, B.: A reference architecture for semantic content management systems. In: 4th Int. Workshop on Enterprise Modelling and Information Systems Architectures, EMISA 2011. LNI, vol. 190, pp. 135–148. GI (2011)
4. Erling, O.: Virtuoso, a hybrid rdbms/graph column store. IEEE Data Eng. Bull. 35(1), 3–8 (2012)
5. Jentzsch, A., Isele, R., Bizer, C.: Silk - generating rdf links while publishing or consuming linked data. In: ISWC 2010 Posters & Demo Track, vol. 658. CEUR-WS.org (2010)
6. Lehmann, J.: DL-Learner: learning concepts in description logics. Journal of Machine Learning Research (JMLR) 10, 2639–2642 (2009)
7. Lehmann, J., Auer, S., Bühmann, L., Tramp, S.: Class expression learning for ontology engineering. Journal of Web Semantics 9, 71–81 (2011)
8. Lehmann, J., Bühmann, L.: ORE - A Tool for Repairing and Enriching Knowledge Bases. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part II. LNCS, vol. 6497, pp. 177–193. Springer, Heidelberg (2010)
9. Lehmann, J., Hitzler, P.: Concept learning in description logics using refinement operators. Machine Learning Journal 78(1-2), 203–250 (2010)
10. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia spotlight: Shedding light on the web of documents. In: 7th I-Semantics (2011)
11. Mendes, P.N., Mühleisen, H., Bizer, C.: Sieve: Linked Data Quality Assessment and Fusion. In: 2nd Int. WS on Linked Web Data Mgmt (LWDM 2012) at EDBT 2012 (2012)
12. Murdock, I.: The Debian Manifesto (1994), http://www.debian.org/doc/manuals/project-history/ap-manifesto.en.html
13. Ngonga Ngomo, A.-C., Auer, S.: Limes - a time-efficient approach for large-scale link discovery on the web of data. In: IJCAI (2011)
14. Schandl, T., Blumauer, A.: PoolParty: SKOS Thesaurus Management Utilizing Linked Data. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part II. LNCS, vol. 6089, pp. 421–425. Springer, Heidelberg (2010)
15. Tramp, S., Frischmuth, P., Ermilov, T., Auer, S.: Weaving a Social Data Web with Semantic Pingback. In: Cimiano, P., Pinto, H.S. (eds.) EKAW 2010. LNCS (LNAI), vol. 6317, pp. 135–149. Springer, Heidelberg (2010)
16. Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., Decker, S.: Sig.ma: Live views on the web of data. J. Web Sem. 8(4), 355–364 (2010)

Achieving Interoperability through Semantics-Based Technologies: The Instant Messaging Case

Amel Bennaceur[1], Valérie Issarny[1], Romina Spalazzese[2], and Shashank Tyagi[3]

[1] Inria, Paris-Rocquencourt, France
[2] University of L'Aquila, L'Aquila, Italy
[3] Institute of Technology, Banaras Hindu University, India

Abstract. The success of pervasive computing depends on the ability to compose a multitude of networked applications dynamically in order to achieve user goals. However, applications from different providers are not able to interoperate due to incompatible interaction protocols or disparate data models. Instant messaging is a representative example of the current situation, where various competing applications keep emerging. To enforce interoperability at runtime and in a non-intrusive manner, *mediators* are used to perform the necessary translations and coordination between the heterogeneous applications. Nevertheless, the design of mediators requires considerable knowledge about each application as well as a substantial development effort. In this paper we present an approach based on ontology reasoning and model checking in order to generate correct-by-construction mediators automatically. We demonstrate the feasibility of our approach through a prototype tool and show that it synthesises mediators that achieve efficient interoperation of instant messaging applications.

Keywords: Interoperability, Composition, Ontology, Verification, Mediation, Universal Instant Messaging.

1 Introduction

Pervasive computing promises a future where a multitude of networked applications dynamically discover one another and seamlessly interconnect in order to achieve innovative services. However, this vision is hampered by a plethora of independently-developed applications with *compatible functionalities* but which are unable to interoperate as they realise them using disparate interfaces (data and operations) and protocols. Compatible functionalities means that at a high enough level of abstraction, the functionality provided by one application is semantically equivalent to that required by the other.

The evolution of instant messaging (IM) applications provides a valuable insight into the challenges facing interoperability between today's communicating applications. Indeed, the number of IM users is constantly growing – from around 1.2 billion in 2011 to a predicted 1.6 billion in 2014 [21] – with an increasing

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 17–33, 2012.

emphasis on mobility – 11% of desktop computers and 18% of smartphones have instant messaging applications installed [19]– and the scope of IM providers is expanding to include social networking such as Facebook that embeds native IM services onto their Web site. Consequently, different versions and competing standards continue to emerge. Although this situation may be frustrating from a user perspective, it seems unlikely to change. Therefore, many solutions that aggregate the disparate systems, without rewriting or modifying them, have been proposed [12]. These solutions use intermediary middleware entities, called *mediators* [24] – also called mediating adapters [25], or converters [4] – which perform the necessary coordination and translations to allow applications to interoperate despite the heterogeneity of their data models and interaction protocols.

Nevertheless, creating mediators requires a substantial development effort and thorough knowledge of the application-domain. Moreover, the increasing complexity of today's software systems, sometimes referred to as Systems of Systems [15], makes it almost impossible to manually develop 'correct' mediators, i.e., mediators guaranteeing deadlock-free interactions and the absence of unspecified receptions [25]. Starlink [3] assists developers in this task by providing a framework that performs the necessary mediation based on a domain-specific description of the translation logic. Although this approach facilitates the development of mediators, developers are still required to understand both systems to be bridged and to specify the translations.

Furthermore, in pervasive environments where there is no *a priori* knowledge about the concrete applications to be connected, it is essential to guarantee that the applications associate the same *meaning* to the data they exchange, i.e., semantic interoperability [10]. *Ontologies* support semantic interoperability by providing a machine-interpretable means to automatically reason about the meaning of data based on the shared understanding of the application domain [1]. Ontologies have been proposed for Instant Messaging although not for the sake of protocol interoperability but rather for semantic archiving and enhanced content management [8]. In a broader context, ontologies have also been widely used for the modelling of Semantic Web Services, and to achieve efficient service discovery and composition [17]. Semantic Markup for Web Services[1] (OWL-S) uses ontologies to model both the functionality and the behaviour of Web services. Besides semantic modelling, Web Service modelling Ontology (WSMO) supports runtime mediation based on pre-defined mediation patterns but without ensuring that such mediation does not lead to a deadlock [6]. Although ontologies have long been advocated as a key enabler in the context of service mediation, no principled approach has been proposed yet to the automated synthesis of mediators by systematically exploiting ontologies [2].

This paper focuses on distributed applications that exhibit compatible functionalities but are unable to interact successfully due to mismatching interfaces or protocols. We present an approach to synthesise mediators automatically to ensure the interoperation of heterogeneous applications based on the semantic compatibility of their data and operations. Specifically, we rely on a domain-specific

[1] http://www.w3.org/Submission/OWL-S/

ontology (e.g., an IM ontology) to infer one-to-one mappings between the operations of the applications' interfaces and exploit these mappings to generate a correct-by-construction mediator. Our contribution is threefold:

- *Formal modelling of interaction protocols.* We introduce an ontology-based process algebra, which we call Ontology-based Finite State Processes (OFSP), to describe the observable behaviour of applications. The rationale behind a formal specification is to make precise and rigorous the description and the automated analysis of the observable behaviour of applications.
- *Automated generation of mediators for distributed systems.* We reason about the semantics of data and operations of each application and use a domain ontology to establish, if they exist, one-to-one mappings between the operations of their interfaces. Then, we verify that these mappings guarantee the correct interaction of the two applications and we generate the corresponding mediator.
- *Framework for automated mediation.* We provide a framework that refines the synthesised mediator and deploys it in order to automatically translate and coordinate the messages of mediated applications.

Section 2 examines in more detail the challenges to interoperability using the IM case. Section 3 introduces the ontology-based model used to specify the interaction protocols of application. Section 4 presents our approach to the automated synthesis of mediators that overcome data and protocol mismatches of functionally compatible applications and illustrates it using heterogeneous instant messaging applications. Section 5 describes the tool implementation while Section 6 reports the experiments we conducted with the instant messaging applications and evaluate the approach. The results show that our solution significantly reduces the programming effort and ensures the correctness of the mediation while preserving efficient execution time. Section 7 examines related work. Finally, Section 8 concludes the paper and discusses future work.

2 The Instant Messaging Case

Instant messaging (IM) is a popular application for many Internet users and is now even embedded in many social networking systems such as Facebook. Moreover, since IM allows users to communicate in real-time and increases their collaboration, it is suitable for short-lived events and conferences such as Instant Communities for online interaction at the European Future Technologies Conference and Exhibition[2] (FET'11) that took place in May 2011.

Popular and widespread IM applications include Windows Live Messenger[3] (commonly called MSN messenger), Yahoo! Messenger[4], and Google Talk[5]

[2] http://www.fet11.eu/
[3] http://explore.live.com/windows-live-messenger/
[4] http://messenger.yahoo.com/
[5] http://www.google.com/talk/

which is based on the Extensible Messaging and Presence Protocol[6] (XMPP) standard protocol. These IM applications offer similar functionalities such as managing a list of contacts or exchanging textual messages. However, a user of Yahoo! Messenger is unable to exchange instant messages with a user of Google Talk. Indeed, there is no common standard for IM. Thus, users have to maintain multiple accounts in order to interact with each other (see Figure 1). This situation, though cumbersome from a user perspective, unfortunately reflects the way IM – like many other existing applications – has developed.

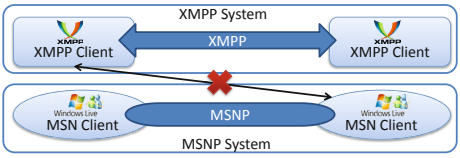


Fig. 1. Interoperability issue between heterogeneous IM systems

A solution that guarantees interoperability between heterogeneous IM applications has to cope with the following heterogeneity dimensions:

- *Data heterogeneity.* MSN Messenger protocol (MSNP), the protocol used by Windows Live Messenger, uses text-based messages whose structure includes several constants with predefined values. On the other hand, the Yahoo! Messenger Protocol (YMSG) defines binary messages that include a header and key-value pairs. As for XMPP messages, they are defined according to a given XML Schema.
- *Protocol heterogeneity.* Even though IM applications are simple and quite similar, each one communicates with its own proprietary application server used to authenticate and to relay the messages between instant messaging clients. Consequently, each application has its own interaction protocol.

Achieving interoperability between independently developed systems has been one of the fundamental goals of middleware research. Prior efforts have largely concentrated on solutions where conformance to the same standard is required e.g., XMPP. However, compliance to a unique standard is not always feasible given the competitive pressures in the marketplace.

Middleware-based approaches define a common abstraction interface (e.g., Adium[7]) or an intermediary protocol (e.g., J-EAI[8] and CrossTalk [18]) promote interoperability in a transparent manner. However, relying on a fixed intermediary interface or protocol might become restrictive over time as new functionalities and features emerge. By synthesising mediators automatically and rigorously we relieve developers from the burden of implementing or specifying such mediators and further ensures their correctness.

[6] http://www.xmpp.org/
[7] http://adium.im/
[8] http://www.process-one.net

Semantics-based solutions (e.g., SAM [8] and Nabu[9]) use ontologies to enhance the functionalities of IM applications by reasoning about the content of messages and overcoming mismatches at the data level but assume the use of the same underlying communication protocol. Hence, even though an enormous amount of work is being carried out on the development of concrete interoperability solutions that rely on ontologies to overcome application heterogeneity, none propose an approach to generate mediators able to overcome both data and protocol heterogeneity. In the next section, we introduce our ontology-based approach to interoperability that automatically synthesises mediators to transparently solve both data and protocol mismatches between functionally compatible applications at runtime.

3 Ontology-Based Modelling of Interaction Protocols

Automated mediation of heterogeneous applications requires the adequate modelling of their data and interaction protocols. In this section, we introduce OFSP (Ontology-based Finite State Processes), a semantically-annotated process algebra to model application behaviour.

3.1 Ontologies in a Nutshell

An ontology is *a shared, descriptive, structural model, representing reality by a set of concepts, their interrelations, and constraints under the open-world assumption* [1]. The Web Ontology Language[10] (OWL) is a W3C standard language to formally model ontologies in the Semantic Web. Concepts are defined as OWL classes. Relations between classes are called OWL properties. Ontology reasoners are used to support automatic inference on concepts in order to reveal new relations that may not have been recognised by the ontology designers. OWL is based on description logics (DL), which is a knowledge representation formalism with well-understood formal properties [1]. To verify the interaction of networked applications, we are in particular interested in specialisation/generalisation relations between their concepts. In this sense, DL resemble in many ways type systems with concept *subsumption* corresponding to type subsumption. Nevertheless, DL are by design and tradition well-suited for domain-specific services and further facilitate the definition and reasoning about composite concepts, e.g., concepts constructed as disjunction or conjunction of other concepts. Subsumption is the basic reasoning mechanism and can be used to implement other inferences, such as satisfiability and equivalence, using pre-defined reductions [1]:

Definition 1 ($\sqsubseteq$: Subsumption). *A concept C is subsumed by a concept D in a given ontology $\mathcal{O}$, written $C \sqsubseteq D$, if in every world consistent with the axioms of the ontology $\mathcal{O}$ the set denoted by C is a subset of the set denoted by D.*

[9] http://nabu.opendfki.de/

[10] http://www.w3.org/TR/owl2-overview/

The subsumption relation is both transitive and reflexive and defines a hierarchy of concepts. This hierarchy always contains a built-in top concept *owl:Thing* and bottom concept *owl:Nothing*.

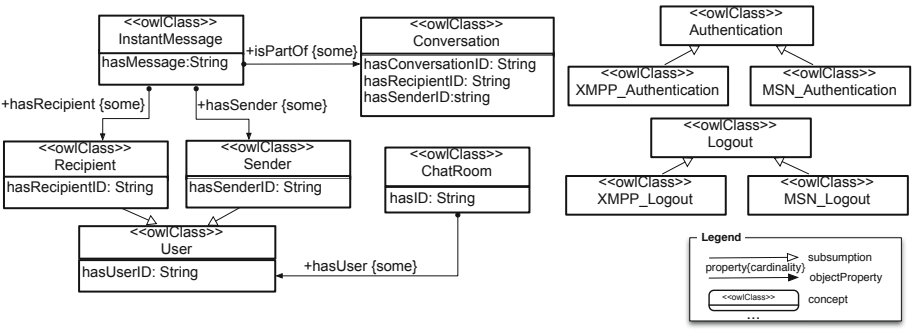


Fig. 2. The instant messaging ontology

Figure 2 depicts the instant messaging ontology. An InstantMessage class has at least one sender hasSender{some}, one recipient hasRecipient{some}, and one message hasMessage. hasSender{some} and hasRecipient{some} are object properties that relate an instant message to a sender or a recipient while hasMessage is a data property associated with the InstantMessage class. The Sender and Recipient classes are subsumed by the User class. Indeed, any instance of the two former classes is also an instance of the latter. A Conversation is performed between a sender (who initialises it) and a recipient, and the conversation has its own identifier. An instant message isPartOf a conversation. A ChatRoom represents a venue where multiple users can join and exchange messages.

3.2 Modelling Protocols Using Ontology-Based FSP

The interaction protocol of an application describes how the operations of its *interface* are coordinated in order to achieve a specified functionality. We build upon state-of-the-art approaches to formalise interaction protocols using process algebra, in particular Finite State Processes (FSP) [14]. FSP has proven to be a convenient formalism for specifying concurrent systems. Although another process algebra would have worked equally well, we choose FSP for convenience and to exploit the Labelled Transition System Analyser (LTSA) in order to automate reasoning and analysis of interaction protocols specified as finite processes.

Each process P is associated with an interface αP that defines the set of observable actions that the application requires from/provides to its running environment. We structure these actions and annotate them using a domain ontology $\mathcal{O}$ so as to specify their semantics, resulting in Ontology-based FSP (OFSP). An *input action* $a = <op, I, O>$ specifies a required operation $op \in \mathcal{O}$ for which the application produces a set of input data $I = \{in \in \mathcal{O}\}$ and consumes a set of

output data $O = \{out \in \mathcal{O}\}$. The dual *output action*[11] $\overline{b} = <\overline{op}, I, O>$ refers to a provided operation *op* for which the application uses the inputs I and produces the corresponding outputs O. Note that all actions are annotated using the same domain ontology $\mathcal{O}$ describing the application-specific concepts and relations. The rationale behind this notation is to enable behavioural analysis based on the semantics of process actions. Indeed, only if both sides of communication assign the same semantics to their actions, can they interact correctly. In addition, τ is used to denote an internal action that cannot be observed by the environment. There are two types of processes: *primitive processes* and *composite processes*. Primitive processes are constructed through action prefix ($\rightarrow$), choice ($|$), and sequential composition ($;$). Composite processes are constructed using parallel composition ($\|$).

The semantics of OFSP builds upon the semantics of FSP, which is given in terms of Labelled Transition Systems (LTS) [13]. The LTS interpreting an OFSP process P is a directed graph whose nodes represent the process states and each edge is labelled with an action $a \in \alpha P$ representing the behaviour of P after it engages in an action a. $P \xrightarrow{a} P'$ denotes that P transits with action a into P'. $P \overset{s}{\Rightarrow} P'$ is a shorthand for $P \xrightarrow{a_1} P_1 ... \xrightarrow{a_n} P', s = \langle a_1, ..., a_n \rangle, a_i \in \alpha P \cup \tau$. There exists a start node from which the process begins its execution. The END state indicates a successful termination. $traces(P)$ denotes the set of all successfully-terminating traces of P. When composed in parallel, processes synchronise on dual actions while actions that are in the alphabet of only one of the two processes can occur independently of the other process.

MSNClient = (<*MSN_Authentication_Request*, {*UserID*}, {*Challenge*} >
→ <*MSN_Authentication_Response*, {*Response*}, {*Authentication_ok*} >
→ ExchangeMsgs).
ExchangeMsgs = (<*CreateChatRoom*, {*UserID*}, {*ConversationID*} >
→ <*JoinChatRoom*,{*UserID*},{*Acceptance*} >→P1
| < $\overline{\textit{JoinChatRoom}}$, {*UserID*}, {*Acceptance*} >
→ < {*ChatRoomInfo*, ∅, {*ConversationID*} > →P1),
P1 = (<$\overline{\textit{InstantMessage}}$, {*UserID*, *ConversationID*, *Message*}, ∅ > → P1
| <$\overline{\textit{InstantMessage}}$, {*UserID*, *ConversationID*, *Message*}, ∅ > → P1
| <*MSN_Logout*, {*UserID*}, ∅ > → END).

Fig. 3. OFSP specification of MSNP

XMPPClient = (< *XMPP_Authentication_Request*, {*UserID*}, {*Challenge* } >
→ <*XMPP_Authentication_Response*, {*Response*}, {*Authentication_ok*} >
→ ExchangeMsgs).
ExchangeMsgs = (<$\overline{\textit{InstantMessage}}$, {*SenderID*, *RecepientID*, *Message*}, ∅ >→ ExchangeMsgs
| <$\overline{\textit{InstantMessage}}$, {*SenderID*, *RecipientID*, *Message*}, *emptyset* >
→ ExchangeMsgs
| < *XMPP_Logout*, {*UserID*}, *emptyset* >→ END).

Fig. 4. OFSP specification of XMPP

The concepts and properties defined in the IM ontology are used to specify MSNP and XMPP clients using OFSP, as illustrated in Figures 3 and 4

[11] Note the use of an overline as a convenient shorthand notation to denote output actions

respectively, focusing on message exchange. Each IM application performs authentication and logout with the associated server. Before exchanging messages, the MSNP application has to configure a *chat room* where the MSN conversation can take place between the user that initiates this conversation (sender) and the user who accepts to participate in this conversation (recipient). In XMPP each message simply contains both the sender and the recipient identifiers.

4 Ontology-Based Approach to Mediator Synthesis

In this section we consider two functionally-compatible applications, described through OFSP processes P_1 and P_2, that are unable to interoperate due to differences in their interfaces or protocols. Functional compatibility means that their required/provided high-level functionalities are semantically equivalent [12]. Our aim is to enforce their interoperation by synthesising a mediator that addresses these differences and guarantees their *behavioural matching*. The notion of behavioural matching is formally captured through *refinement* [11]. A process Q *refines* a process P if every trace of Q is also a trace of P, i.e., $traces(Q) \subseteq traces(P)$. However, this notion of refinement analyses the dynamic behaviour of processes assuming close-world settings, i.e., the use of the same interface to define the actions of both processes. What is needed is a notion of compatibility that takes into account the semantics of actions while relying on a mediator process M to compensate for the syntactic differences between actions and guarantees that the processes communicate properly.

To this end, we first reason about the semantics of actions so as to infer the correspondences between the actions of the processes' interfaces and generate the mapping processes that perform the necessary translations between *semantically compatible actions*. Various mapping relations may be defined. They primarily differ according to their complexity and inversely proportional flexibility. In this paper we focus on one-to-one mappings, i.e., direct correspondences between actions. During the synthesis step, we explore the various possible mappings in order to produce a correct-by-construction mediator, i.e., a mediator M that guarantees that the composite process $P_1 \| M \| P_2$ reaches an END state, or determines that no such mediator exists.

In this section we introduce the semantic compatibility of actions, and use it to define behavioural matching. Then, we present the automated synthesis algorithm.

4.1 Semantic Compatibility of Actions

A *sine qua non* condition for two processes P_1 and P_2 to interact is to agree on the data they exchange. However, independently-developed applications often define different interfaces. The mediator can compensate for the differences between interfaces by mapping their actions if and only if they have the same semantics. We first define the notion of *action subsumption* and then, use it to define the *semantic compatibility* of actions.

Definition 2 ($\sqsubseteq_{\mathcal{O}}$: Action Subsumption). *An action $a_1 = <op_1, I_1, O_1>$ is subsumed by an action $\overline{a_2} = <\overline{op_2}, I_2, O_2>$ according to a given ontology $\mathcal{O}$, noted $a_1 \sqsubseteq_{\mathcal{O}} a_2$, iff: (i) $op_2 \sqsubseteq op_1$, (ii) $\forall i_2 \in I_2, \exists i_1 \in I_1$ such that $i_1 \sqsubseteq i_2$, and (iii) $\forall o_1 \in O_1, \exists o_2 \in O_2$ such that $o_2 \sqsubseteq o_1$.*

The idea behind this definition is that an input action can be mapped to an output one if the required operation is less demanding; it provides richer input data and needs less output data. This leads us to the following definition of semantic compatibility of actions:

Definition 3 ($\approx_{\mathcal{O}}$: Semantic Compatibility of Actions). *An action a_1 is semantically compatible with an action a_2, denoted $a_1 \approx_{\mathcal{O}} a_2$, iff a_1 is subsumed by a_2 (i.e., a_1 is required and a_2 provided) or a_2 is subsumed by a_1 (a_2 is required and a_1 provided) .*

The semantic compatibility between two actions allows us to generate an action mapping process as follows:

$$\mathsf{M}_{\mathcal{O}}(a_1, a_2) = \begin{cases} a_1 \overset{\mathcal{O}}{\longmapsto} \overline{a_2} \text{ if } a_1 \text{ is subsumed by } \overline{a_2} \\ a_2 \overset{\mathcal{O}}{\longmapsto} \overline{a_1} \text{ if } a_2 \text{ is subsumed by } \overline{a_1} \end{cases}$$

The process that maps action a_1 to action $\overline{a_2}$, written $a_1 \overset{\mathcal{O}}{\longmapsto} \overline{a_2}$ captures each input data from the input action, assigns it to the appropriate input of the output action ($i_2 \leftarrow i_1$), then takes each output data of the output action and assigns it to the expected output of the input action ($o_1 \leftarrow o_2$). This assignment is safe since an instance of i_1 (resp. o_2) is necessarily an instance i_2 of (resp. o_1).

Let us consider $a_1 = <InstantMessage, \{UserID, ConversationID, Message\}, \emptyset>$ associated to the MSN client and $\overline{a_2} = <\overline{InstantMessage}, \{SenderID, RecipientID, Message\}, \emptyset>$ associated to the XMPP client. The IM ontology indicates that (i) Sender is subsumed by User, and (ii) $ConversationID$ identifies a unique Conversation, which includes a $RecipientID$ attribute. Consequently, a_1 is subsumed by $\overline{a_2}$.

4.2 Behavioural Matching through Ontology-Based Model Checking

We aim at assessing behavioural matching of two processes P_1 and P_2 given the semantic compatibility of their actions according to an ontology $\mathcal{O}$. To this end, we first filter out communications with third party processes [20]. The communicating trace set of P_1 with P_2, noted $traces(P_1)\uparrow_{\mathcal{O}} P_2$ is the set of all successfully-terminating traces of P_1 restricted to the observable actions that have semantically compatible actions in αP_2.

Definition 4 ($\uparrow_{\mathcal{O}}$: Communicating Trace Set). $traces(P_1)\uparrow_{\mathcal{O}} P_2 \overset{\text{def}}{=}$
$\{s = \langle a_1, a_2, ..., a_n \rangle , a_i \in \alpha P_1 \mid P_1 \overset{s}{\Rightarrow} \mathsf{END}$ *such that* $\forall a_i, \exists b_i \in \alpha P_2 | a_i \approx_{\mathcal{O}} b_i\}$

As an illustration, both the MSNP and XMPP IM clients perform their authentication and logout with their respective servers. Additionally, MSNP also performs the actions related to the configuration of the chat room with its servers.

Consequently their communicating traces sets are restricted to instant message exchange.

Then, two traces $s_1 = \langle a_1 a_2 ... a_n \rangle$ and $s_2 = \langle b_1 b_2 ... b_n \rangle$ *semantically match*, written $s_1 \equiv_{\mathcal{O}} s_2$, iff their actions semantically match in sequence.

Definition 5 ($\equiv_{\mathcal{O}}$: Semantically Matching Traces)

$$s_1 \equiv_{\mathcal{O}} s_2 \stackrel{\text{def}}{=} a_i \approx_{\mathcal{O}} b_i \quad 1 \leq i \leq n$$

The associated mapping is then as follows:

$$\mathsf{Map}_{\mathcal{O}}(s_1, s_2) = \mathsf{M}_{\mathcal{O}}(a_1, b_1); ...; \mathsf{M}_{\mathcal{O}}(a_n, b_n)$$

Based on the semantic matching of traces, a process P_2 *ontologically refines* a process P_1 ($P_1 \models_{\mathcal{O}} P_2$) iff each trace of P_2 semantically matches a trace of P_1:

Definition 6 ($\models_{\mathcal{O}}$: Ontological Refinement)

$$P_1 \models_{\mathcal{O}} P_2 \stackrel{\text{def}}{=} \forall s_2 \in traces(P_2)\uparrow_{\mathcal{O}} P_1, \exists s_1 \in traces(P_1)\uparrow_{\mathcal{O}} P_2 : s_2 \equiv_{\mathcal{O}} s_1$$

By checking ontological refinement between P_1 and P_2, we are able to determine the following *behavioural matching* relations:

- *Exact matching*–$(P_1 \models_{\mathcal{O}} P_2) \wedge (P_2 \models_{\mathcal{O}} P_1)$: assesses compatibility for symmetric interactions such as peer-to-peer communication where both processes provide and require the similar functionality.
- *Plugin matching*–$(P_1 \models_{\mathcal{O}} P_2) \wedge (P_2 \not\models_{\mathcal{O}} P_1)$: evaluates compatibility for asymmetric interactions such as client-server communication where P_1 is providing a functionality required by P_2.
- *No matching*–$(P_1 \not\models_{\mathcal{O}} P_2) \wedge (P_2 \not\models_{\mathcal{O}} P_1)$: identifies behavioural mismatch.

Behavioural matching is automated through *ontology-based model checking*. Model checking is an attractive and appealing approach to ensure system correctness that proved to be a very sound technique to automatically verify concurrent systems. The gist of model checking approaches is the exhaustive state exploration. This exploration is performed by model checkers using efficient algorithms and techniques that make it possible to verify systems of up to 10^{1300} states in few seconds [7]. However, even if these techniques effectively handle very large systems, the actions of the models they consider are usually simple strings and the verification matches actions based on their syntactic equality. We build upon these model checking techniques but further match actions based on their semantic compatibility. The semantic compatibility of actions is defined based on the domain knowledge encoded within a given ontology.

Referring to the IM case, all the traces of MSNP and XMPP processes semantically match. Subsequently, these two processes are in exact matching relation, and a mediator can be synthesised to perform action translations and enable their correct interaction.

4.3 Automated Mediator Synthesis

In the case where P_1 and P_2 match, that is exact matching in the case of peer-to-peer communication or plugin matching in the case of client/server communication, we synthesise the mediator that makes them properly interact. The algorithm incrementally builds a mediator M by forcing the two protocols to progress synchronously so that if one requires an action a, the other must provide a semantically compatible action $\overline{b}$. The mediator compensates for the syntactic differences between their actions by performing the necessary transformations, which is formalised as follows:

$$\mathsf{Mediator}_{\mathcal{O}}(P_1, P_2) = \|\mathsf{Map}(s_1, s_2) \text{ such that}$$
$$s_2 \in traces(P_2)\uparrow_{\mathcal{O}} P_1, s_1 \in traces(P_1)\uparrow_{\mathcal{O}} P_2 : s_2 \equiv_{\mathcal{O}} s_1$$

In the IM case, we are able to produce the mediator for the MSNP and XMPP processes as illustrated in Figure 5. The mediator intercepts an instant message sent by an MSNP user and forwards it to the appropriate XMPP user. Similarly, each instant message sent by an XMPP user, is forwarded by the mediator to the corresponding MSNP user.

$$\begin{aligned}
\mathsf{Map}_1 &= (<InstantMessage, \{SenderID, RecepientID, Message\}, \emptyset > \\
&\quad \rightarrow <\overline{InstantMessage}, \{UserID, ConversationID, Message\}, \emptyset > \rightarrow \mathsf{END}). \\
\mathsf{Map}_2 &= (< InstantMessage, \{UserID, ConversationID, Message\}, \emptyset > \\
&\quad \rightarrow <\overline{InstantMessage}, \{SenderID, RecepientID, Message\}, \emptyset > \rightarrow \mathsf{END}). \\
\|\mathsf{Mediator} &= (\mathsf{Map}_1 \| \mathsf{Map}_2).
\end{aligned}$$

Fig. 5. OFSP specification of the Mediator between MSNP and XMPP

5 Implementation

In order to validate our approach, we have combined the LTSA[12] model checker with an OWL-based reasoner to achieve ontological refinement leading to the OLTSA tool (Figure 6-❶). LTSA is a free Java-based verification tool that automatically composes, analyses, graphically animates FSP processes and checks safety and liveness properties against them.

In the case where the processes match, a concrete mediator that implements the actual message translation is deployed atop of the Starlink framework [3], see Figure 6-❸. Starlink interprets the specification of mediators given in a domain-specific language called Message Translation Logic (MTL). An MTL specification describes a set of *assignments* between message *fields*. The messages correspond to action names and the fields to the name of input/output data. Note that the OFSP description focuses on the ontological annotations and not the the actual name. Therefore, we refine the OFSP specification of the mediator so as to generate the associated MTL before deploying the mediator atop of Starlink, see Figure 6-❷.

[12] http://www.doc.ic.ac.uk/ltsa/

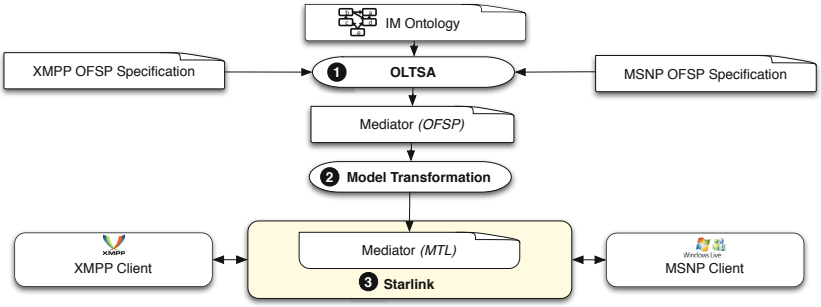


Fig. 6. Mediation Architecture

Let us consider the mapping Map_1 (see Figure 5), which transforms an XMPP input action to the associated MSNP action. Figure 7 shows a small fragment of the associated translation logic described in MTL and which corresponds to the assignment of the $UserID$ field of the XMPP message ($ReceivedInstantMessage$) to the $SenderID$ field of of the MSNP message (SDG) with the mediator transiting from state $XS1$ to state $MR1$.

The tool, the IM ontology, and a video demonstration are available at http://www-roc.inria.fr/arles/download/imInteroperability/.

```
<translationlogic>
 <assignment>
  <field>
   <statelabel>MR1</statelabel><message>SDG</message>
   <xpath>/field/primitiveField[label='UserID']/value</xpath>
  </field>
  <field>
   <statelabel>XS1</statelabel><message>ReceivedInstantMessage</message>
   <xpath>/field/primitiveField[label='SenderID']/value</xpath>
  </field></assignment> ....
</translationlogic>
```

Fig. 7. Translation logic to map MSNP and XMPP instant messages

6 Assessment

In this section we first report a set of experiments we conducted to evaluate the effectiveness of our approach when applying it to the instant messaging case. Then, we discuss some of its quality properties.

6.1 Experimental Results

We have evaluated the time for translating one protocol to the other by the synthesised mediator and the effort required by the developer to enable mediation. We have hand-coded a mediator that makes MSNP, YMSG, XMPP interoperable in order to gauge the complexity of the mediation. We considered the

Windows Live Messenger for MSNP, Yahoo! Messenger for YMSG, and Pidgin[13] for XMPP. We run the OLTSA tool and the Starlink framework on a Mac computer with a 2,7 GHz processor and 8 GB of memory.

In the first experiment, we measured the time taken to translate from one protocol to another. We repeated the experiments 50 times and reported the mean time for each case in Table 1. The hand-coded mediator is approximately 3 times faster than the synthesised one. This is mainly due to the fact that the models are first interpreted then executed by Starlink at runtime whereas the hand-coded mediator is already compiled and hence more efficient.

In the second experiment, we measured the time for synthesising the mediator (see Table 2). One can note that action mapping is the most time consuming step as it necessitates ontology reasoning in order to infer semantically matching actions while the behavioural matching is performed is less than 1 ms. Nevertheless, this step needs to be performed once only and is definitely faster than hand-coding the mediator or even specifying it. Moreover, for each new version of one of the protocols, the hand-coded mediator has to be re-implemented and re-compiled, Starlink requires the specification of the translation logic to be re-specified whereas the automated synthesis requires only the specification of the protocol to be re-loaded.

Table 1. Translation time (ms)

	Hand-coded	Mediator atop Starlink
YMSG ↔ MSNP	22	69
MSNP ↔ XMPP	52	131
YMSG ↔ XMPP	44	126

Table 2. Time for Synthesis (ms)

	Act. mapping	Beh. match.
YMSG ↔ MSNP	306	<1
MSNP ↔ XMPP	252	<1
YMSG ↔ XMPP	244	<1

The third experiment measures the effort demanded from the developer to produce mediators between different IM applications. We calculate the number of Java code lines of the hand-coded mediator, the number of lines of DSL specification that need to be specified for Starlink and those needed to specify the individual applications for the automated synthesis.

Table 3. Development effort

	Hand-Coded	Starlink	Automated
YMSG ↔ MSNP	1172	258	96
MSNP ↔ XMPP	750	198	84
YMSG ↔ XMPP	945	168	76

The results are given in Table 3. One can notice that although Starlink reduces considerably (around 4 times) the lines of code that need to be written,

[13] http://www.pidgin.im/

the automated approach requires the OFSP specifications only and decreases this number drastically (around 10 times). This is mainly due to (i) the use of OFSP to model the interaction protocols, which introduces an ontology-based domain-specific language grounded in process algebra and especially targeted for concurrent systems. For example, the MSNP behaviour is described in Starlink using 30 XML lines and only 6 lines with our approach (ii) Further, the translation code need not be specified. More importantly, unlike the hand-coded or the Starlink versions where the developer is required to know both protocols and define the translation manually, the protocols are specified separately in the automated version. Thus, each IM provider can independently specify its own protocol. Finally, we are investigating within the CONNECT[14] project learning-based techniques to infer such a specification automatically [2].

To sum up, our automated approach to interoperability significantly reduces the programming effort and ensures the correctness of the translation while requiring a negligible time for synthesising the mediator and guaranteeing good performances at translation time.

6.2 Qualitative Assessment

In addition to the above-mentioned performances, our approach satisfies the following properties:

- *Correctness by construction.* The correctness of the mediation, i.e., the absence of deadlock and unspecified receptions [25], is guaranteed by construction. Indeed, if there is an exact match between P_1 and P_2 then the parallel composition $P_1 \| M \| P_2$ is deadlock free. Exact matching means that each trace of P_1 (P_2) has a corresponding semantically-matching trace in P_2 (P_1), which amounts to setting P_1 (P_2) as a safety property that needs to be verified by P_2 (P_1). This verification is performed by exhaustively exploring the state space. Note though that efficient model checkers use optimisation techniques to reduce the space if possible. The reduction techniques are even more efficient in the case of process algebra.
- *Formal yet tractable DSL specification.* OFSP introduces an ontology-based domain-specific language grounded in process algebra. Process algebra constitute a very expressive behavioural specification language for complex concurrent systems while ontologies are the model of choice to describe data semantics. Furthermore, standard modelling languages that developers are familiar with (e.g., BPEL or CDL) can be used to specify the interaction protocols and then automatically translate them to FSP using existing tools[15].
- *Dealing with encryption.* When encryption is enforced (e.g., Google Talk encrypts XMPP messages), the mediator cannot parse or modify these messages all the way between the initial sender and the ultimate receiver. Transparency cannot be ensured anymore. Instead, the user get involved and handles some of the translation tasks [23]. In the Google Talk case, the mediator

[14] http://connect-forever.eu/
[15] http://www.doc.ic.ac.uk/ltsa/bpel4ws/

uses a robot (bot) that the user adds to its contact list. The robot manages a set of commands, e.g., `IM <destinationID> <message>` to send a message `message` to user `destinationID`.

7 Related Work

The problem of mediating applications has been studied in different domains. Middleware solutions focus on providing abstraction and execution environments that enable interoperation by providing an abstract interface and exploiting reflection [9], by translating into a common intermediary protocol such as in the case of Enterprise Service Buses [16] or by proposing a domain-specific language to describe the translation logic and automatically generate the corresponding gateways [3]. However, these solutions require the developer to specify the translation to be made and hence to know both protocols in advance whereas in our approach, each protocol is independently specified and the translation is produced automatically. The Web Service Execution Environment (WSMX) performs the necessary translation on the basis of pre-defined mediation patterns. However, the composition of these patterns is not considered, and there is no guarantee that it will not lead to a deadlock. Vaculín *et al.* [22] devise a mediation approach for OWL-S processes. They first generate all requester paths, then find the appropriate mapping for each path by simulating the provider process. This approach deals only with client/server Web service interactions. It is not able to deal with the heterogeneity of instant messaging applications for example. Calvert and Lam [4] propose an approach to reason about the existence of a mediator by projecting both systems into a common sub-protocol. However, this common sub-protocol needs to be specified using an intuitive understanding of the protocols. In their seminal paper, Yellin and Strom [25] propose an algorithm for the automated synthesis of mediators based on predefined correspondences between messages. By considering the semantics of actions, we are able to infer the correspondences between messages automatically. Finally, Cavallaro *et al.* [5] also consider the semantics of data and relies on model checking to identify mapping scripts between interaction protocols automatically. However, they do not take into account the actual semantics of the operations. Moreover, they propose to perform the interface mapping beforehand so as to align the vocabulary of the processes, but many mappings may exist and should be considered during the generation of the mediator. Hence, even though there exists a significant amount of work to achieve interoperability, none of the existing approaches proposes to generate automatically mediators that are able to deal with both data and protocol mismatches.

8 Conclusion

Achieving interoperability between heterogeneous distributed applications without actually modifying their interfaces or behaviour is desirable and often

necessary in today's pervasive systems. Mediators promote the seamless interconnection of distributed applications by performing the necessary translations between their messages and coordinating their behaviour. In this paper, we have presented a principled approach to the automated synthesis of mediators at runtime. We first infer mappings between application interfaces by reasoning about the semantics of their data and operations annotated using a domain-specific ontology. We then use these mappings to automatically synthesise a correct-by-construction mediator. This principled approach to generating mediators removes the need to develop *ad hoc* bridging solutions and fosters future-proof interoperability. We evaluated the approach using a case study involving heterogeneous instant messaging applications and showed that it can successfully ensure their interoperation.

Work in progress includes the definition of many-to-many operation mappings to manage a broader set of heterogeneous systems. We are also investigating the synthesis of mediators between more than a pair of networked applications. This is for example the case when IM conversations involve multiple users. Our work further integrates with complementary work ongoing within the CONNECT European project so as to develop a framework to support the interoperability lifecycle by using semantic technologies to synthesise mediators dynamically and ensure their evolution to respond efficiently to changes in the individual systems or in the ontology. A further direction is to consider improved modelling capabilities that take into account the probabilistic nature of systems and the uncertainties in the ontology. This would facilitate the construction of mediators where we have only partial knowledge about the system.

Acknowledgment. This work is carried out as part of the European CONNECT and EternalS projects.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook. Cambridge University Press (2003)
2. Blair, G.S., Bennaceur, A., Georgantas, N., Grace, P., Issarny, V., Nundloll, V., Paolucci, M.: The role of ontologies in emergent middleware: Supporting interoperability in complex distributed systems. In: Kon, F., Kermarrec, A.-M. (eds.) Middleware 2011. LNCS, vol. 7049, pp. 410–430. Springer, Heidelberg (2011)
3. Bromberg, Y.D., Grace, P., Réveillère, L.: Starlink: Runtime interoperability between heterogeneous middleware protocols. In: Proc. ICDCS (2011)
4. Calvert, K.L., Lam, S.S.: Formal methods for protocol conversion. IEEE Journal on Selected Areas in Comm. (1990)
5. Cavallaro, L., Di Nitto, E., Pradella, M.: An Automatic Approach to Enable Replacement of Conversational Services. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) ICSOC-ServiceWave 2009. LNCS, vol. 5900, pp. 159–174. Springer, Heidelberg (2009)
6. Cimpian, E., Mocan, A.: WSMX process mediation based on choreographies. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 130–143. Springer, Heidelberg (2006)

7. Clarke, E.M., Grumberg, O., Long, D.E.: Model checking and abstraction. ACM Trans. Program. Lang. Syst. (1994)
8. Franz, T., Staab, S.: SAM: Semantics aware instant messaging for the networked semantic desktop. In: Proc. International Sem. Web Conf. Workshops (2005)
9. Grace, P., Blair, G.S., Samuel, S.C.: ReMMoC: A Reflective Middleware to Support Mobile Client Interoperability. In: Meersman, R., Schmidt, D.C. (eds.) CoopIS/DOA/ODBASE 2003. LNCS, vol. 2888, pp. 1170–1187. Springer, Heidelberg (2003)
10. Heiler, S.: Semantic interoperability. ACM Surv. (1995)
11. Hoare, C.A.R.: Communicating Sequential Processes. Prentice-Hall (1985)
12. Issarny, V., Bennaceur, A., Bromberg, Y.-D.: Middleware-Layer Connector Synthesis: Beyond State of the Art in Middleware Interoperability. In: Bernardo, M., Issarny, V. (eds.) SFM 2011. LNCS, vol. 6659, pp. 217–255. Springer, Heidelberg (2011)
13. Keller, R.M.: Formal verification of parallel programs. Commun. ACM (1976)
14. Magee, J., Kramer, J.: Concurrency: State models and Java prog. Wiley (2006)
15. Maier, M.W.: Integrated modeling: A unified approach to system engineering. Journal of Syst. and Softw. (1996)
16. Menge, F.: Enterprise Service Bus. In: Proc. Free and Open Source Soft. Conf. (2007)
17. Ben Mokhtar, S., Kaul, A., Georgantas, N., Issarny, V.: Efficient Semantic Service Discovery in Pervasive Computing Environments. In: van Steen, M., Henning, M. (eds.) Middleware 2006. LNCS, vol. 4290, pp. 240–259. Springer, Heidelberg (2006)
18. Motoyama, M.A., Varghese, G.: CrossTalk: scalably interconnecting instant messaging networks. In: Proc. ACM Workshop on Online Social Networks (2009)
19. Nielsen: Games Dominate America's Growing Appetite for Mobile Apps (2010)
20. Spalazzese, R., Inverardi, P., Issarny, V.: Towards a formalization of mediating connectors for on the fly interoperability. In: WICSA/ECSA (2009)
21. The Radicati Group: Instant Messaging Market 10-14 (2010)
22. Vaculín, R., Neruda, R., Sycara, K.P.: The process mediation framework for semantic web services. Journal of Agent-Oriented Softw. Eng. (2009)
23. Vassilakis, C., Kareliotis, C.: A framework for adaptation in secure web services. In: Medi. Conf. on Info. Syst. (2009)
24. Wiederhold, G.: Mediators in the architecture of future info. syst. Computer (1992)
25. Yellin, D.M., Strom, R.E.: Protocol specifications and component adaptors. ACM Trans. Prog. Lang. Syst. (1997)

Linking Smart Cities Datasets with Human Computation – The Case of UrbanMatch

Irene Celino[1], Simone Contessa[1], Marta Corubolo[2], Daniele Dell'Aglio[1], Emanuele Della Valle[2,1], Stefano Fumeo[1], and Thorsten Krüger[3]

[1] CEFRIEL – Politecnico di Milano, Milano, Italy
{name.surname}@cefriel.it
[2] Politecnico di Milano, Milano, Italy
{name.surname}@polimi.it
[3] Siemens, Corporate Technology, Munich, Germany
thorsten.krueger@siemens.com

Abstract. To realize the Smart Cities vision, applications can leverage the large availability of open datasets related to urban environments. Those datasets need to be integrated, but it is often hard to automatically achieve a high-quality interlinkage. Human Computation approaches can be employed to solve such a task where machines are ineffective. We argue that in this case not only people's background knowledge is useful to solve the task, but also people's physical presence and direct experience can be successfully exploited. In this paper we present UrbanMatch, a Game with a Purpose for players in mobility aimed at validating links between points of interest and their photos; we discuss the design choices and we show the high throughput and accuracy achieved in the interlinking task.

1 Introduction

Cities are defined smart when their investments in the human and social capital, as well as in the communication infrastructures are aimed at fuelling a sustainable economic growth and a high quality of life [6]. Specifically, current research investigates the impact of ICT on the development and improvement of smart cities with respect to several dimensions, from people to government, from mobility to environment, etc. In this context, a key to realize smart cities is to involve smart citizens by raising their awareness, participation and contribution.

Big industrial players are focusing their research and innovation around smart cities; some examples are the initiatives carried out by Siemens[1], IBM[2] and CISCO[3]. Public authorities are also becoming more and more attentive to adapt their political agenda to fulfil this smart cities vision, in particular through an open data strategy.

Geo-spatial data and information related to entities located in the physical world are among the first sources that are published openly – and often also

[1] http://www.usa.siemens.com/sustainable-cities/
[2] http://www.ibm.com/uk/smarterplanet
[3] http://www.cisco.com/web/strategy/smart_connected_communities.html

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 34–49, 2012.

freely – on the Web; valuable examples are Ordnance Survey location data in the UK[4], GeoLinkedData.es in Spain[5], GeoNames geographical database[6] and the community-driven OpenStreetMap[7]. The Semantic Web community also has showed interest in geo-spatial data: OpenStreetMap was turned into Linked Data by the LinkedGeoData project [26] and the Open Geospatial Consortium is standardizing GeoSPARQL[8], a spatial extension of the SPARQL language.

For the last years, we have been experimenting with geo-spatial data – especially with those related to urban environments – in order to build Linked Data-enhanced applications and services. The used datasets and the applications objectives were diverse: points of interest and event data to plan journeys [9]; traffic sensors data and road topography to predict the most suitable path [10]; urban regulations to update road sign information [17]; social media to provide location-based recommendations of restaurants [3].

While the large availability of urban data is an advantage in realizing such kind of services, the poor quality or the doubtful trustworthiness of the information source strongly hamper a large-scale adoption of those data. Imprecise or outdated information, sparse or heterogeneous distribution of data are just some examples of the obstacles to a proper reuse of geo-spatial (linked) data. Our experience tells that inconsistencies and imprecise data can be detected – and their quality improved – by a small amount of manual work that does not require specific skills, but often the physical presence in the urban environment [17].

Human Computation [29] is the paradigm to leverage human capabilities to solve tasks that computers are not yet able to properly undertake. A Human Computation approach is often employed to solve data quality tasks.

Our research question can be formulated as follows: is it possible to exploit people's *physical presence* in the environment to improve geo-spatial data quality? Can we build a new generation of Human Computation techniques based on the contributors' *direct experience* (instead of a specific domain expertise)?

To check our hypothesis, we built UrbanMatch [7], a location-based Game with a Purpose [28] in the form of a mobile application[9]. Specifically, UrbanMatch is aimed at exploiting players' experience of the urban environment to correctly link points of interests in the city with their most representative photos retrieved from Web sources. The paper's *contribution* lies in the modelling of the POI-photo linking as a record linkage problem and the realization of the game using this formalization; we also experimentally determined the best combination of the model's parameters, in order to optimize the trade-off between the number of links created per playing hour (system throughput) and the accuracy of those links.

The remainder of this paper is organized as follows. Section 2 introduces the related work; Section 3 defines the problem statement, while the process to

[4] http://www.ordnancesurvey.co.uk/
[5] http://geo.linkeddata.es/
[6] http://geonames.org/
[7] http://www.openstreetmap.org/
[8] http://www.opengeospatial.org/standards/requests/80
[9] UrbanMatch is available on iTunes app store at http://bit.ly/um-itunes.

achieve the game purpose is detailed in Section 4. Section 5 explains the mechanics of the UrbanMatch game, while the evaluation is illustrated in Section 6; finally Section 7 draws some conclusions and future work.

2 Background and Related Work

Our work focuses on user interaction for link elicitation and validation for Linked Data in urban scenarios. It is centred on Linked Data and it is based on the results of three research areas: data linking, data quality, and human computation.

2.1 Data Linking and Linked Data

Data Linking is the problem of deciding whether resources belonging to different data sources are referring to the same entity. It is rooted in the record linkage problem studied in the databases community since the 1960s [13,20,32].

Record linkage is a challenging task, as deciding if records match is often computationally expensive and application specific [5]. The former is because a combination of string similarity algorithms have to be used, the latter because it is difficult to provide a general solution which works well with heterogeneous datasets. For instance, the techniques used in linking scientific datasets will be different from the ones used for linking CRM datasets.

In this paper, we are particularly interested in referring to the formal definition of record linkage introduced by Felligi and Sunter in [13], which we use in the rest of the paper. When linking the records in two databases A and B, the idea is a) to classify links in the comparison space $\Gamma = A \times B$ into M – the set of matches –, and U – the set of non-matches –; b) to compute for each link γ a score s as ratio of probabilities $P(\gamma \in \Gamma|M)/P(\gamma \in \Gamma|U)$; and c) to use the score s to divide the comparison space in three disjoint sets using an $UPPER$ thresholds, a $LOWER$ threshold and a decision rule. If $s > UPPER$, then the link is correct; if $LOWER \leq s \leq UPPER$, then the link needs to be assessed by an expert; if $s < LOWER$, then the link is incorrect.

Establishing links between datasets published as linked data [27] is a problem rooted in record linkage, but can benefit from the availability of ontologies describing the datasets to be linked, and, thus, from existing ontology matching solutions [22,12]. At the time of writing, SERIMI [2], Zhishi.links [21] and AgreementMaker [8] are the best data linking solutions emerged from the Data Interlinking track of the OAEI 2011 challenge [11].

2.2 Data Quality and Linked Data

Data Quality [23,4] is the discipline that studies the most appropriate and relevant features to describe the value of data.

A key point of Data Quality is the context-dependency: given a dataset, its quality can be very high w.r.t. the fulfilment of some tasks but very bad w.r.t. other ones. As pointed out in [16], "the perception of information quality (on the WWW) is highly dependent on the *fitness for use* being relative to the specific

task that users have at their hands". In other words, it is not relevant (and not always possible) to define absolute quality factors [19]. More specifically, [23] enumerates the following factors contributing to *fitness-for-use*: accuracy, completeness, consistency with other sources, timeliness, accessibility, relevance, comprehensiveness, easy to *read* and easy to *interpret*. In this work, we evaluate our solution using accuracy and throughput (see Section 2.3) as a proxy for completeness.

It is worth noting that the Linked Data best practices alone [15] assure more quality than "raw data" in "closed" databases because: *a*) data becomes *accessible over the Web* rather than being closed up in silos; *b*) the use of *shared vocabularies* makes the data both easier to "read" (i.e. user information needs can be satisfied by a single SPARQL query instead of requiring many dataset-specific queries) and easier to "interpret" (i.e. shared vocabulary semantics can be used to verify data integrity and/or infer implied data); *c*) the *presence of links* makes it also possible to verify consistency across different sources.

However, the assessment of data quality factors like accuracy, completeness, timeliness, relevance and comprehensiveness of data is intrinsically a hard task that Linked Data best practices do not make any easier. As one can expect, the quality of published Linked Data is variable and the community has started to follow data quality with growing interest. Flemming worked on the definition of quality criteria for linked data sources [14]. She grouped the criteria to describe data sources in four categories: *content* (the quality of the data as available in the dataset), *representation* (an evaluation of the data serialization), *usage* (the measurement of the data "fitness for use") and *system* (indicators about the publishing system).

2.3 Human Computation and Linked Data

As we showed in the two previous sections, data linking and data quality are hard problem for computers and subjective in nature. We, as humans, are perfectly capable of both tasks, but we are not necessarily willing to. Human Computation [29], however, demonstrated that "computations" of this kind can be carried out by groups of people if motivated by the right incentives.

The incentives to make people contribute can be of different kinds: they can give the participant an explicit and concrete reward (like in the popular Amazon Mechanical Turk[10] in which people are paid to perform small and simple tasks) or they provide a different kind of implicit or more abstract return, for example by means of entertainment like in Games with a Purpose [28] (GWAP).

In this paper, we are specifically interested in the design and evaluation of GWAPs as UrbanMatch is a GWAP. Having created many GWAPs (e.g., ESP Game, Peekaboom, Phetch, and Verbosity), Luis Von Ahn and Laura Dabbish reports in [31] on three game-structure templates that generalize successful instances of Human Computation games: input-agreement games, inversion-problem games, and output-agreement games. In input-agreement games, players must determine whether they have been given the same input; in inversion-problem games,

[10] http://mturk.com/

given an input, a player produces an output, and another player guesses the input; in output-agreement games, players are given the same input and must agree on an appropriate output. UrbanMatch is an output-agreement game.

UrbanMatch is not the first GWAP proposed by the Semantic Web community. GWAPs have been already used to cover the complete Semantic Web life-cycle [25]. A dedicated community portal was recently set up[11] to collect those games. A good showcase is the Linked Data Movie Quiz [1], that builds a cinematographic game based on the available movie-related Linked Data showing that "the answers are out there; and so are the questions".

The metrics [31] proposed to evaluate GWAPS include throughput and average lifetime play (ALP). The *throughput* of a GWAP is defined as the average number of problem instances solved per human hour. The higher the throughput the more effective the GWAP. However, a GWAP with a high throughput that fails to attract and keep players is useless. The *ALP* is a proxy for the intangible enjoyability of the GWAP. It is defined as the overall amount of time the game is played by each player, averaged across all people who have played it. A successful GWAP like the ESP game [30] has a throughput of 233 problem instances solved per human-hour and an ALP of 91 minutes. We use those metrics to evaluate Urban Match in Section 6.

3 Problem Statement

UrbanMatch aims at linking urban related data sets. More specifically, the purpose of UrbanMatch is to derive meaningful links between a datasets containing the points of interest (POIs) in a urban environment and a dataset with the images depicting those POIs and retrieved from Web social media; among all photos taken in the proximity of a POI, UrbanMatch is designed for linking the most representative ones to that POI.

We selected the first dataset A of POIs from OpenStreetMap/LinkedGeoData and we retrieved the second dataset B collecting photos from social media sharing sources, namely Flickr and Wikimedia Commons. The first edition of the UrbanMatch game is released for the city of Milano in Italy, thus the POIs in dataset A are *tourist attractions* in Milano, i.e. entities in LinkedGeoData [26] that are instances of classes like `lgdo:Monument`, `lgdo:Historic` or `lgdo:Landmark`.

Dataset A contains the POIs aggregated by *playable place*. A playable place is an open area (like a square or a park) that is physically adjacent to at least two tourist attractions. Playable places are retrieved via a spatial-enhanced query on OpenStreetMap. Given the list of playable places with the corresponding POIs, we appealed to an expert judgement (a person, among the authors, familiar with the city and its notable locations) to filter out the irrelevant elements and to complete a list of alternative names/labels to indicate the selected POIs. The result was a set of 14 playable places with 34 POIs in Milano.

To retrieve the dataset B of the photos, we used the POIs geographic coordinates to perform a *spatial query* on the image sources – Flickr and Wikimedia

[11] `http://www.semanticgames.org/`

Commons – by invoking the respective API. This location-based query was enhanced with other information about the POIs: on Flickr API, geographic coordinates were used together with a *keyword search* by using the alternative POIs names/labels. On the other hand, Wikimedia Commons – the media database related to Wikipedia – puts in relation its photos with the Wikipedia page that describes the depicted POI; thus, the retrieval requests mix the geographic coordinates with a *"conceptual" search*, comparing a Wikipedia page with the "concept" of the respective POI. The result of the photo selection was a set of 11,287 photos of Milano POIs.

We can formulate the *data interlinking problem* that UrbanMatch aims to solve as a record linkage problem using the formal definition of record linkage introduced by Felligi and Sunter in [13] (already cited in Section 2.1). Thus, we define the set Γ of all possible links between POIs and photos as the comparison space[12] between the two datasets, i.e. $\Gamma = A \times B$. Each link $\gamma_{p,n} \in \Gamma$ can be seen as an RDF triple of the form:

```
<POI-n> foaf:depiction <photo-p> .
```

in which `<POI-n>` is the URI of the POI n in dataset A and `<photo-p>` is the URI of the photo p in dataset B.

Data interlinking is achieved when all the links γ in the comparison space are classified in two sets: the set M of "matches", i.e. of *correct links*, and the set U of "non-matches", i.e. of *incorrect links*. Each link $\gamma_{p,n}$ is associated with a score $s_{p,n} \in [0..1]$ that represents the probability of the link to be correct; two thresholds are usually defined – $UPPER$ and $LOWER$ – so that:

$$\text{if } s_{p,n} > \text{UPPER} \quad \text{then } \gamma_{p,n} \in M$$

$$\text{if } s_{p,n} < \text{LOWER} \quad \text{then } \gamma_{p,n} \in U$$

The comparison space Γ between two datasets A and B can be seen as divided into three disjoint sets: M, U and the set C of unclassified links, for which:

$$\text{LOWER} \leq s_{p,n} \leq \text{UPPER}$$

Solving the data interlinking problem, therefore, requires the ability to alter the score of the links $\gamma_{p,n} \in C$. Those links represent our *candidate links* that need to be assessed to be classified either in M or in U.

In the case of UrbanMatch, the candidate links in C are those links whose quality is not appropriate according to the *fitness-for-use* principle [23]. For example, a candidate link could connect a POI with a photo that frames the inside of that POI, or a non-evocative detail of the POI, or people in front of the POI. The central idea of UrbanMatch is to use a GWAP to ask players to assess the candidate links in C and to alter the score of each link until it falls either in M or in U.

Each link γ between a POI in the dataset A and a photo in the dataset B is given an initial score s between $LOWER$ and $UPPER$, so that all links

[12] A reduction of the comparison space Γ by partitioning is explained in Section 4.

initially belong to the subset C of candidate links. To bootstrap the UrbanMatch approach, we manually modified the score of some candidate links in C to a value either greater than $UPPER$ or lower than $LOWER$. To this end, we appealed to an expert judgement to select some photos depicting the POIs: on the basis of this selection, some links moved from C to M (because the photos depicts *for sure* the POIs) while some other links moved from C to U (because they do *not* depict the POIs). The result of this preparation phase was a comparison space Γ with 196 links in M, 382 links in U and 37,413 link in C. Table 1 recaps the initial dataset with the detail for each playable place in Milano.

Table 1. The initial dataset of UrbanMatch Milano (created in spring 2012)

Playable Place Name	bootstrapped photos	total photos	POIs	links in M	links in U	links in C
Piazza dei mercanti	16	301	2	16	16	634
Piazza Sant'Ambrogio	12	180	2	10	10	380
Piazza del Duomo	32	3,884	5	32	128	19,580
Piazza Duca d'Aosta	11	1,032	2	11	11	2,086
Via Legnano	10	418	2	10	10	856
Via Conservatorio	13	325	2	13	13	676
Viale Alemagna	12	217	2	12	12	458
Largo Marco Biagi	9	973	2	9	9	1,964
Corso di Porta Ticinese (1)	12	142	2	12	12	308
Corso di Porta Ticinese (2)	11	376	2	11	11	774
Via Gioia	15	110	2	15	15	250
Corso Venezia	10	979	2	10	10	1,978
Piazza della Scala	20	885	4	20	80	3,620
Piazza Cairoli & Viale Petofi	15	1,269	3	15	45	3,852
Total	198	11,089	34	196	382	37,413

4 Achieving the UrbanMatch Purpose

The UrbanMatch game was designed to let the players rate the candidate links. However, presenting players directly with the RDF links is not a user-friendly way to let them solve the task. Moreover, if the players are not in the urban space or if they do not have enough background knowledge about the POIs, it could also be difficult for them to say, for example, if a photo actually depicts "Palazzo della Ragione". For those reasons, UrbanMatch is designed as a single-player mobile game to be played on the go, in which the players are presented only with the photos and are asked to pair those that represent the same POI around them. The players may not know the name of the depicted POI, but if the photo is representative, they can recognize it.

The photo pairs selected by the player are used by UrbanMatch to alter the scores of the candidate links between those photos and the POIs around the player. Let us assume that UrbanMatch shows the player two photos a and b and it wants to assess if the two photos are both linked to the same POI 1. Let

us also assume that the link $\gamma_{a,1} \in M$, i.e. the link between a and POI 1 is correct, and that $\gamma_{b,1} \in C$, i.e. the link between b and POI 1 is candidate. The player can decide whether to pair the two photos or not. If the player pairs the two photos, the score of the link $\gamma_{b,1}$ is modified; the new value of the score $s'_{b,1}$ of the link $\gamma_{b,1}$ between the photo b and the POI 1 is increased using the formula in Equation 1

$$s'_{b,1} = s_{b,1} + K_{pos} \tag{1}$$

where K_{pos} is a positive constant that counts for the *positive evidence* provided by the player.

Otherwise, if the player does not pair the two photos, the new value of the score $s'_{b,1}$ of the link $\gamma_{b,1}$ between the photo b and the POI 1 is decreased using the formula in Equation 2

$$s'_{b,1} = s_{b,1} - K_{neg} \tag{2}$$

where K_{neg} is a positive constant that counts for the *negative evidence* provided by the player. Collecting positive and negative evidences for each link in C, UrbanMatch alters the score of each candidate link until it is categorized as belonging either to M or to U.

An important issue arises when modifying the links' scores: are players reliable? We can certainly trust a large majority of the players to play earnestly, but we need to consider that some players can cheat or misunderstand the task, thus giving wrong answers. As proposed in [31], we can mitigate the risk of trusting erroneous inputs with two strategies: *i*) repeating the same task multiple times to randomly picked users, and *ii*) testing the player reliability.

The approach described in Equations 1 and 2 is ready to support the first strategy, by opportunely tuning the values of K_{pos} and K_{neg}. As noted in [31], this strategy can guarantee the correct assessment of link quality with arbitrarily high probability.

The second strategy can be embedded in Equations 1 and 2 by testing players multiple times per game and evaluating their reliability on the basis of the number of errors they make. As a result of the bootstrapping, a number of incorrect links exist; UrbanMatch puts them in the game as verification cases. For example, let us assume that we have only two POIs (1 and 2), and UrbanMatch shows the player two photos (a and b) each depicting only one of the two POIs: a depicts 1 (i.e., $\gamma_{a,1} \in M$, $\gamma_{a,2} \in U$) and b depicts 2 (i.e., $\gamma_{b,1} \in U$, $\gamma_{b,2} \in M$). If the player pairs a and b he/she makes a mistake, because the two photos certainly depicts different POIs. If ϵ_p is the number of errors done by a player p in a game, the player's reliability can be computed as:

$$r_p = e^{-\frac{\epsilon_p}{2}}$$

Note that r_p is a float in [0..1]: it is equal to 1 when the player makes no error, decreases to $e^{-\frac{1}{2}} = 0.6$ when the player makes 1 error, and drops almost to zero if the player makes 6 errors ($e^{-\frac{1}{6}} = 0.04$).

Considering also the player's reliability, Equations 1 and 2 respectively take the form of Equations 3 and 4:

$$s'_{b,1} = s_{b,1} + K_{pos} * r_p \tag{3}$$

$$s'_{b,1} = s_{b,1} - K_{neg} * r_p \tag{4}$$

Table 2 wraps up the decision rules that allow to increase/decrease the score of a link and to detect errors. More information about the initial value of s, and the values of K_{pos}, K_{neg}, $UPPER$ and $LOWER$ is given in Section 6.

Table 2. The two tables above show the decision rules used by UrbanMatch when the player pairs, or does not pair, two photos a and b on the basis of the scores of the links $\gamma_{a,1}$ and $\gamma_{b,1}$ between those two photos and a POI 1. The table on the left shows the case when the player pairs a with b, while the table on the right shows the case when the player does not pair a with b. The symbol *n.a.* means *no action*, $s_{i,j}$++ means that UrbanMatch increments the value of the score of the link $\gamma_{i,j}$ by using Equation 3, $s_{i,j}$- - means that UrbanMatch decrements the value of the score of the link $\gamma_{i,j}$ by using Equation 4, and ϵ_p++ means that UrbanMatch increases the error counter for the player p.

paired with	$\gamma_{a,1} \in M$	$\gamma_{a,1} \in U$	$\gamma_{a,1} \in C$
$\gamma_{b,1} \in M$	n.a.	ϵ_p++	$s_{a,1}$++
$\gamma_{b,1} \in U$	ϵ_p++	n.a.	n.a.
$\gamma_{b,1} \in C$	$s_{b,1}$++	n.a.	n.a.

not paired with	$\gamma_{a,1} \in M$	$\gamma_{a,1} \in U$	$\gamma_{a,1} \in C$
$\gamma_{b,1} \in M$	ϵ_p++	n.a.	$s_{a,1}$- -
$\gamma_{b,1} \in U$	n.a.	n.a.	n.a.
$\gamma_{b,1} \in C$	$s_{b,1}$- -	n.a.	n.a.

Functionally, the proposed solution solves the UrbanMatch problem, but an efficient solution should also consider the need for a high throughput (defined in Section 2.3). This result can be obtained by combining two approaches.

On the one hand, we can reduce the problem space by partitioning[13] the comparison space Γ. The partition is based on the concept of playable places: the comparison space Γ is built by considering only the links between each photo, retrieved in correspondence to a playable place, and all POIs visible from the same playable place. In other words, UrbanMatch discards from the comparison space Γ all the links between the photos, which were retrieved using geo-coordinates and labels of a given playable place, and the POIs in the other playable places.

On the other hand, UrbanMatch splits the set of candidates links C into two subsets: $C_{engaged}$, the set of links currently evaluated by the UrbanMatch game, whose score can be altered by players' actions, and $C_{retained}$, the set of links not yet evaluated by the game. In this way, positive and negative evidences are gathered only for links in $C_{engaged}$ whose score reaches the $UPPER$ or $LOWER$ threshold at the maximum speed. As soon as a link moves from $C_{engaged}$ to M or U, a new link is fetched from $C_{retained}$ and added to $C_{engaged}$ for evaluation.

[13] The comparison space partitioning is a well-known technique in record linkage [18].

5 The UrbanMatch Gameplay

In this section we illustrate how UrbanMatch works internally. We explain the game levels construction and the feedbacks to the players' pairing actions.

5.1 Game Level Definition

When the player starts the UrbanMatch app on her device, her location is detected to make her play with what surrounds her. In case of doubt (e.g. the user is close to more than one playable place), a map with the close-by locations is displayed.

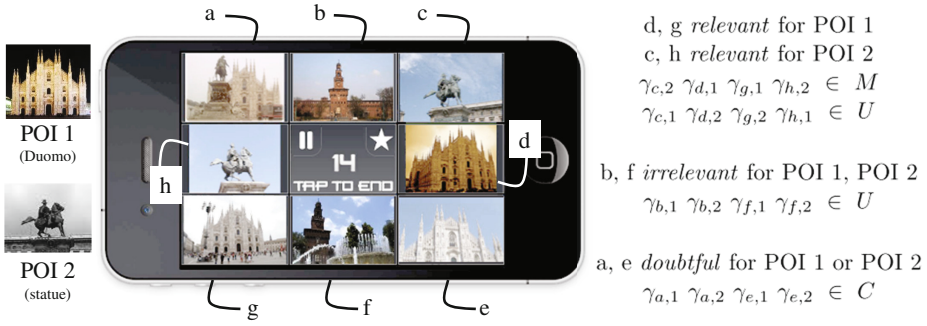


Fig. 1. Explanation of the photos presented in a game level

Once the playable place is selected, the game starts and the players are presented with the first game level; a maximum of six levels are created and given as input to the players in each match. In each level, two POIs (1 and 2) of the playable place are considered and eight different photos (from a to h) are selected and displayed (cf. Figure 1). The photos are selected according to the following policy:

- for each of the two POIs, two *relevant photos* are definitely linked to them (d and g to POI 1, c and h to POI 2), thus representing four links belonging to the M set of correct links and four links belonging to the U set of incorrect links;
- two *irrelevant photos* are definitely linked to other POIs (b and f), not visible from this playable place; those photos are certainly not linked to the current POIs 1 and 2, thus representing four links belonging to the U set of incorrect links;
- the remaining two *doubtful photos* are not certainly linked to the current POIs 1 and 2 (a and e), thus they are representative of four candidate links from the C set of links to be validated.

The players are then asked to pair the photos depicting the same POI, but they must be careful not to select those photos referring to POIs in a different playable place, i.e. those POIs they cannot see around them.

5.2 Feedbacks to Players

Whenever a player pairs two photos, this action is taken as an evidence of the correspondence between the respective links represented by those photos; each evidence is weighted according to Equations 3 and 4 and taken into consideration according to the policy defined in Table 2. Besides the application of the decision rules – which is important for the hidden purpose of data linking – the coupling action is also used to give an immediate feedback to the user within the gameplay. We decided to always give a positive or negative feedback to the player, even when UrbanMatch is in doubt; moreover, we chose to prefer a positive reward to a negative reward in case of doubt, to motivate the user to continue playing.

Whenever a user pairs two photos between the relevant and irrelevant ones, UrbanMatch always knows if the coupling action is right or wrong: either the two photos are certainly linked to the same POI – and thus the player gets a positive feedback and gains points – or they are definitely linked to different POIs – and thus the player gets a negative feedback and loses points.

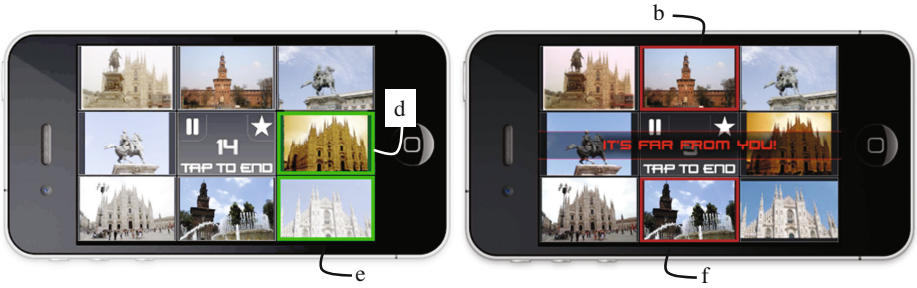


Fig. 2. Positive and negative feedbacks w.r.t. the photo pairs chosen by the player

Every time a user pairs a doubtful photo with another one, UrbanMatch does not know if the coupling action is right or wrong, but it gives the best possible feedback: pairing a doubtful photo with a relevant one or pairing two doubtful photos gives the player a positive feedback; pairing a doubtful photo with an irrelevant one gives a negative feedback.

Figure 2 on the left shows the positive feedback given to a coupling action in a level played in "Piazza del Duomo" playable place in Milano: UrbanMatch displays a green frame around the two selected photos (d and e that both depict the Duomo cathedral of Milano) and plays a "success" sound. On the contrary, the right part of Figure 2 shows the negative feedback given to another pairing action during the same game level: UrbanMatch displays a red frame around the two selected photos (b and f, depicting Castello Sforzesco, are clearly irrelevant for this playable place since the castle is not in the playable place) together with the textual banner and plays a "failure" sound.

Each "successful" or "failure" pairing action is associated in the gameplay with a positive or negative score respectively: the sum of the scores in one level determines if the player can continue to the next level and the total score in all the levels of the match determines the position in the leaderboard.

6 Experimental Deployment and Evaluation

As reported in Section 3, in spring 2012 we experimentally deployed UrbanMatch in Milano using OpenStreetMap as POI dataset, and Flickr together with Wikimedia Commons as photo data sources. For each photo a retrieved by a query related to a POI 1, the initial score $s_{a,1}$ was set to 0.4 if the source was Flickr and 0.6 if the source was Wikimedia Commons; this was because we considered Wikimedia Commons search precision to be higher than Flickr's. For each POI i in the same playable place of POI 1, the score $s_{a,i}$ was set to 0.2, because a photo depicting a POI in a playable place may also partially show other POIs in the same playable place (e.g., see photo a in Figure 1, taken in Duomo square, that depicts both Vittorio Emanuele's statue and the Duomo cathedral). For each POI j in a different playable place, no link of the form $\gamma_{a,j}$ was inserted in the comparison space, according to the comparison space partitioning technique discussed in Section 4.

Between March and May 2012, as consequence of an email advertising campaign, seventy people downloaded UrbanMatch from iTunes and 54 of them played the game at least once, for a total of 781 played levels. The total time all players spent playing UrbanMatch is about 3 hours.

As evaluation metrics for UrbanMatch, we chose *throughput* and *ALP* (as defined in [31]), and *accuracy*. The latter plays an important role in deciding the values of $UPPER$ and $LOWER$. The ALP definition is equivalent to the one in [31], but the notion of *throughput* and *accuracy* need to be redefined as:

$$Throughput = \frac{CM + CU}{PlayedTime} \tag{5}$$

$$Accuracy = \frac{(CM - FP) + (CU - FN)}{CM + CU} \tag{6}$$

in which the symbols have the following meaning:

- CM is the number of candidate links that UrbanMatch was able to move from C to M, i.e., emerged as correct;
- CU is the number of candidate links that UrbanMatch was able to move from C to U, i.e., emerged as incorrect;
- FP is the number of links moved from C to M that should have been classified as incorrect, i.e., the false positive links;
- FN is the number of links moved from C to U that should have been classified as correct, i.e., the false negative links; and
- $PlayedTime$ is the total time spent by the players in playing UrbanMatch.

Note that CM and CU are a direct result of UrbanMatch, while FP and FN were manually assessed by one of the authors that lives in Milano and is thus knowledgeable about the city.

As the reader can expect, the throughput and accuracy of UrbanMatch depend on the value of $UPPER$, $LOWER$, K_{pos} and K_{neg}. Therefore we need to determine the best combination of these values to maximize both throughput and accuracy.

We arbitrarily decided that a *positive evidence* of a reliable player counts as +0.3 (i.e., $K_{pos} = 0.3$) and a *negative evidence* counts as −0.1 (i.e., $K_{neg} = 0.1$). We chose $K_{pos} = 3 * K_{neg}$ because players pair photos in which they recognize the same POI, but they may not pair photos for several reasons, e.g., for the little knowledge about the POI, for inexperience, for lack of time and for mistake.

To determine the best values of $UPPER$ and $LOWER$, we analysed the throughput and the accuracy of UrbanMatch as a function of $UPPER$ and $LOWER$. The values in Table 3 are obtained setting $UPPER = 1$ and assigning $LOWER$ the values 0.05, 0.10, 0.15 and 0.20[14]. Both throughput and accuracy increase when increasing the threshold, so $LOWER$ was set to 0.20.

Table 3. Throughput and Accuracy as a function of $LOWER$ threshold

$LOWER$	0.05	0.10	0.15	**0.20**
CU	321	348	1152	**1216**
FN	4	5	7	**8**
Throughput	108.08	117.17	387.87	**409.42**
Accuracy	98.75%	98.56%	99.39%	**99.34%**

The values in Table 4 are obtained setting $LOWER = 0$ and assigning $UPPER$ values between 0.6 and 0.95 using a 0.05 step[15]. Throughput decreases while increasing $UPPER$, but accuracy increases, therefore we need to find a trade-off between the two performance indicators. Noticing that accuracy increase steeply for $UPPER \leq 0.7$ and slightly for $UPPER > 0.7$, we decided to set $UPPER = 0.7$.

Table 4. Throughput and Accuracy as a function of $UPPER$ threshold

$UPPER$	0.60	0.65	**0.70**	0.75	0.80	0.85	0.90
CM	227	225	**68**	65	61	60	49
FP	48	47	**4**	4	3	3	1
Throughput	76.43	75.75	**22.89**	21.88	20.53	20.20	16.49
Accuracy	78.85%	79.11%	**94.11%**	95.38%	95.08%	95.00%	97.95%

The final results are wrapped up in Table 5. The throughput of UrbanMatch is 485 links per played hour; this is twice as much as the throughput of the ESP game [30]. The ALP of UrbanMatch is a bit more than 3 minutes per player; this is a not an outstanding result (the ALP of the ESP game is 91 minutes per player), but this value could be increased by improving the gaming and entertaining features of UrbanMatch. The accuracy of UrbanMatch is 99.06%, which is a significant result. This allow us to assert that UrbanMatch provides an effective solution for link quality assessment.

[14] 0.2 is the greatest value we can assign to $LOWER$ because it is the minimum value we decided to use when initializing scores to links in C.

[15] 0.6 is the smallest value we can assign to $UPPER$ because it is the maximum value we decided to use when initializing scores to links in C.

Table 5. Final evaluation results

CM	*FP*	*CU*	*FN*	Players	PlayedTime	Throughput	ALP	Accuracy
68	4	1216	8	54	2h 58m 12s	485 links/h	3m 17s	99.06%

7 Conclusions

The problems of interlinking and assessing the quality of information published as Linked Data have been recognized of paramount importance by researchers and practitioners, who are investigating the adoption of different approaches. Most research is focused on automated solutions, but crowdsourcing the interlinking or quality assessment tasks is also possible. Actually, if we consider the *fitness-for-use* principle of data quality [16], involving "human processors" may be the only practical way to obtain high quality links.

UrbanMatch, presented in this paper, adopts the approach of Games with a Purpose to assess the quality of automatically created links between POIs and photos that depict them. However, UrbanMatch is not simply a GWAP for Linked Data: it considers the characteristics of urban-related – or, more broadly, geo-spatially related – Linked Data and the possibility to rely on the *on-site experience* of the players in addition to their knowledge.

Our analysis of the links assessed by UrbanMatch in few month of availability on the iTunes store seems to confirm our research hypothesis. Our work and evaluation is currently oriented to gather further evidence in two directions: repeating the UrbanMatch experience in Munich[16] and exploring a different gaming approach with a new app named Urbanopoly[17].

So far, the development and deployment of UrbanMatch in the German city of Munich allowed us to verify the data preparation step, i.e. obtaining POIs from OpenStreetMap with the help of LinkedGeoData, automatically fetching photos from Flickr and Wikimedia Commons and creating the candidate links to be assessed. A preliminary analysis of the matches played in Munich confirms the results obtained in Milano.

On the other hand, a new game named Urbanopoly was designed to get a higher value of ALP w.r.t. UrbanMatch, while keeping the same level of throughput. In analysing the results of UrbanMatch, we noticed that the players were motivated to continue playing by the presence of the leaderboard and the possibility to "beat" other players. Thus, in designing Urbanopoly, we put more emphasis on those gaming features that require a long-term engagement of the player.

The main lesson learned from UrbanMatch is that Human Computation approaches can be successfully employed to interlink urban-related datasets: the on-site experience of the players helps in gathering links with a clear business value. For example, UrbanMatch allows to learn the locations from which a POI is visible and recognizable. This information can be valuable for a wide range

[16] Cf. `http://bit.ly/um-munchen`
[17] Cf. `http://bit.ly/urbanopoly`

of city stakeholders, like municipalities (for placing information totems) or mobile operators (to deliver effective location-aware mobile advertisement). Games like UrbanMatch may serve to a wide range of Smart Cities services like traffic optimization, environmental sustainability or city planning.

Acknowledgements. This work was partially supported by Siemens Corporate Research and Technologies and by the PlanetData project, co-funded by the European Commission (FP7-257641).

References

1. Álvaro, G., Álvaro, J.: A Linked Data Movie Quiz: the answers are out there, and so are the questions (August 2010), http://bit.ly/linkedmovies
2. Araújo, S., Hidders, J., Schwabe, D., de Vries, A.P.: SERIMI - Resource Description Similarity, RDF Instance Matching and Interlinking. CoRR, abs/1107.1104 (2011)
3. Balduini, M., Celino, I., Dell'Aglio, D., Della Valle, E., Huang, T., Lee, T., Kim, S., Tresp, V.: BOTTARI: An Augmented Reality Mobile Application to deliver Personalized and Location-based Recommendations by Continuous Analysis of Social Media Streams. Journal of Web Semantics: Science, Services and Agents on the World Wide Web (2012), doi: 10.1016/j.websem.2012.06.004
4. Batini, C., Cappiello, C., Francalanci, C., Maurino, A.: Methodologies for data quality assessment and improvement. ACM Comput. Surv. 41(3) (2009)
5. Benjelloun, O., Garcia-Molina, H., Menestrina, D., Su, Q., Whang, S.E., Widom, J.: Swoosh: a generic approach to entity resolution. VLDB J. 18(1), 255–276 (2009)
6. Caragliu, A., Del Bo, C., Nijkamp, P.: Smart cities in Europe. Serie Research Memoranda 0048, VU University Amsterdam, Faculty of Economics, Business Administration and Econometrics (2009)
7. Celino, I., Contessa, S., Corubolo, M., Dell'Aglio, D., Della Valle, E., Fumeo, S., Krüger, T.: UrbanMatch – linking and improving Smart Cities Data. In: Linked Data on the Web Workshop, LDOW 2012 (2012)
8. Cruz, I.F., Stroe, C., Caimi, F., Fabiani, A., Pesquita, C., Couto, F.M., Palmonari, M.: Using AgreementMaker to align ontologies for OAEI. In: Shvaiko, et al. (eds.) [24] (2011)
9. Della Valle, E., Celino, I., Dell'Aglio, D.: The Experience of Realizing a Semantic Web Urban Computing Application. Transactions in GIS 14(2) (2010)
10. Della Valle, E., Celino, I., Dell'Aglio, D., Steinke, F., Grothmann, R., Tresp, V.: Semantic Traffic-Aware Routing for the City of Milano using the LarKC Platform. IEEE Internet Computing 15(6) (2011)
11. Euzenat, J., Ferrara, A., van Hage, W.R., Hollink, L., Meilicke, C., Nikolov, A., Ritze, D., Scharffe, F., Shvaiko, P., Stuckenschmidt, H., Sváb-Zamazal, O., dos Santos, C.T.: Results of the ontology alignment evaluation initiative. In: Shvaiko, et al. [24] (2011)
12. Euzenat, J., Shvaiko, P.: Ontology matching. Springer, Heidelberg (2007)
13. Fellegi, I.P., Sunter, A.B.: A Theory for Record Linkage. Journal of the American Statistical Association 64(328), 1183–1210 (1969)
14. Flemming, A., Hartig, O.: Quality Criteria for Linked Data sources, http://bit.ly/ld-quality
15. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Synthesis Lectures on the Semantic Web. Morgan & Claypool (2011)

16. Knight, S.A., Burn, J.: Developing a framework for assessing information quality on the World Wide Web. Informing Science 8 (2005)
17. Lee, T., Park, S., Huang, Z., Della Valle, E.: Toward Seoul Road Sign Management on the LarKC Platform. Posters and Demos of ISWC 2010 (2010)
18. McCallum, A., Wellner, B.: Object consolidation by graph partitioning with a conditionally-trained distance metric. In: Proceedings of the ACM Workshop on Data Cleaning, Record Linkage and Object Identification (2003)
19. Naumann, F., Rolker, C.: Assessment Methods for Information Quality Criteria. In: Klein, B.D., Rossin, D.F. (eds.) IQ. MIT (2000)
20. Newcombe, H.B.: Handbook of record linkage: methods for health and statistical studies, administration, and business. Oxford University Press, Inc. (1988)
21. Niu, X., Rong, S., Zhang, Y., Wang, H.: Zhishi.links results for OAEI. In: Shvaiko et al. [24] (2011)
22. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB J. 10(4), 334–350 (2001)
23. Redman, T.C.: Data Quality: The Field Guide. Digital Press (2001)
24. Shvaiko, P., Euzenat, J., Heath, T., Quix, C., Mao, M., Cruz, I.F. (eds.): Proceedings of the 6th International Workshop on Ontology Matching. CEUR-WS.org, vol. 814 (2011)
25. Siorpaes, K., Hepp, M.: Games with a Purpose for the Semantic Web. Intelligent Systems 23, 50–60 (2008)
26. Stadler, C., Lehmann, J., Höffner, K., Auer, S.: LinkedGeoData: A Core for a Web of Spatial Open Data. Semantic Web Journal (2011)
27. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and Maintaining Links on the Web of Data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 650–665. Springer, Heidelberg (2009)
28. von Ahn, L.: Games with a Purpose. IEEE Computer 39(6), 92–94 (2006)
29. von Ahn, L.: Human Computation. In: Alonso, G., Blakeley, J.A., Chen, A.L.P. (eds.) ICDE, pp. 1–2. IEEE (2008)
30. von Ahn, L., Dabbish, L.: Labeling images with a computer game. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2004, pp. 319–326. ACM, New York (2004)
31. von Ahn, L., Dabbish, L.: Designing games with a purpose. Commun. ACM 51(8), 58–67 (2008)
32. Winkler, W.E.: Overview of record linkage and current research directions. Technical report, Bureau of the Census (2006)

ourSpaces – Design and Deployment of a Semantic Virtual Research Environment

Peter Edwards, Edoardo Pignotti, Alan Eckhardt, Kapila Ponnamperuma, Chris Mellish, and Thomas Bouttaz

Computing Science, University of Aberdeen, Aberdeen AB24 5UA, UK
{p.edwards,e.pignotti,a.eckhardt,k.ponnamperuma,
c.mellish,t.bouttaz}@abdn.ac.uk

Abstract. In this paper we discuss our experience with the design, development and deployment of the *ourSpaces* Virtual Research Environment. *ourSpaces* makes use of Semantic Web technologies to create a platform to support multi-disciplinary research groups. This paper introduces the main semantic components of the system: a framework to capture the provenance of the research process, a collection of services to create and visualise metadata and a policy reasoning service. We also describe different approaches to support interaction between users and metadata within the VRE. We discuss the lessons learnt during the deployment process with three case study groups. Finally, we present our conclusions and future directions for exploration in terms of developing *ourSpaces* further.

Keywords: Provenance, Virtual Research Environment, Policies, NLG.

1 Introduction

Research challenges are becoming increasingly complex requiring researchers from different institutions and different disciplines to work together. At the same time, a range of information technologies have gradually been adopted by researchers to support the transfer of ideas, knowledge and resources, leading to the emergence of Web-based Virtual Research Environments (VREs) [1]. These have been proposed as one way to help researchers in all disciplines to manage the increasingly complex range of tasks involved in carrying out research. In the UK, the Joint Information Systems Committee (JISC) VRE programme[1] explored the virtual research environment collaborative landscape. Results from this programme concluded that one of the most important tasks for the academic community is to provide general frameworks that can be used to develop and host different VREs. Such frameworks should provide core services (such as authentication and rights management; repositories; project planning, collaboration and communication tools) and allow the development or easy integration of modules for specific uses. JISC also recognised that a major shift in research practices will occur through the formation of common taxonomies, data standards and metadata as researchers collaborate with others across disciplinary, institutional and national boundaries [1]. Semantic web technologies [2] are seen as crucial in this context in order to

[1] http://www.jisc.ac.uk/whatwedo/programmes/vre.aspx

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 50–65, 2012.

provide a common framework to allow the creation of intelligent applications and services which can be integrated with data resources, people and other objects in a VRE.

Some of the issues highlighted above have been explored by the PolicyGrid[2] project, a collaboration between human geographers and computer scientists as part of the UK Digital Social Research initiative. As part of this project we have developed *ourSpaces*[3], a semantic VRE which aims to provide a collaborative on-line environment for interdisciplinary academic research communities. Groups using *ourSpaces* work in socio-environmental and health-related domains and there are currently 183 registered users. A screenshot of the *ourSpaces* web interface in presented in Figure 1.

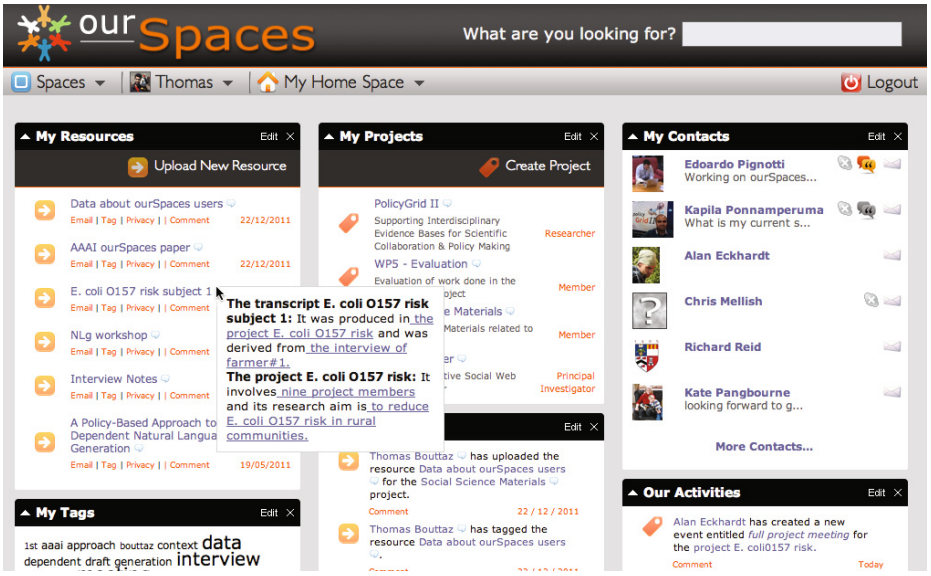


Fig. 1. A screenshot of the *ourSpaces* VRE

Managing provenance is one of the main aspects of the *ourSpaces* VRE which is required in order to make the context surrounding research artefacts more transparent. Understanding the provenance of scientific data is crucial [3] in order to understand and verify its authenticity and completeness. Provenance (also referred to as lineage or audit trail) captures the derivation history of an artefact, including the original sources, the intermediate products and the steps that were applied to produce that artefact. Within the environment, there is also a need to manage users and their behaviours so that they comply with certain policies. For example, a user may impose certain access constraints on digital artefacts that he/she owns, such as restricting an artefact to people within their social network. Semantic web technologies play an important role in supporting the management of provenance and policies. At the heart of *ourSpaces* is an extensible

[2] http://www.policygrid.org
[3] http://www.ourspaces.net

ontological framework describing different aspects of the provenance of the research process [4] and a reasoning service for provenance and policies [5].

Semantic Web technologies have already been applied to virtual research environments. For example, *myExperiment* [6] enables people to share digital objects associated with their research. The notion of *research objects* is used in *myExperiment* to provide a container for semantic aggregation of resources produced and consumed by common services. *myExperiment* uses ontologies to support the publication of such objects as RDF so they can be shared within and across organisational boundaries.

Semantic Web approaches have also been used in enterprise knowledge management tools [7]. For example, the IBM *WebSphere* Portal [8] uses ontologies to support different aspects of document management such as tagging and searching. All three approaches (including *ourSpaces*) employ Semantic Web technologies to provide a representational framework that can be used across different domain applications. However, the main difference between *ourSpaces* and other semantic environments is that it utilises policy reasoning to control the behaviour of users and services. This allows us to adapt our environment to meet domain-specific requirements without changing the logic behind services. We have also developed a number of general-purpose services for creating and visualising Semantic Web data.

In the remainder of this paper we discuss the role of Semantic Web technologies in the design, development and deployment of the *ourSpaces* system. In section 2 we describe the VRE architecture and its underlying components. In section 3 we present the user interfaces that we have developed in order to support interaction with semantic metadata. In section 4 we discuss the lessons learnt during the deployment of *ourSpaces* with our case-study communities. Finally, we present our conclusions and future directions for exploration in terms of developing *ourSpaces* further.

2 The *ourSpaces* Semantic Architecture

Based upon interactions with case study groups and communities, initial requirements for the *ourSpaces* VRE were identified. Even though the requirements for the VRE were clear in broad terms, the finer grain requirements relating to the system architecture and user interfaces were much less well defined due to the diversity of the case study groups. We therefore decided to continuously involve the users in the development of the VRE. This is discussed in more detail in section 4. The initial requirements are summarised below:

- It should be possible to describe and uniquely identify a range of entities: artefacts (digital and physical); processes (both computational services and human activities); people; organisational structures and membership; social networks.
- The system should incorporate online communication (e.g. instant messaging, blog entries, email) into the provenance record.
- It should be possible to define relationships (e.g. causal, social, organisational) between entities.
- It should be possible to define access control and documentation policies.

Following these requirements we have developed an architecture which is summarised in Figure 2. Underpinning *ourSpaces* are a number of repositories and services. Each

activity within the environment is enabled by a rich and pervasive RDF metadata infrastructure built upon a series of OWL ontologies (which are described more fully in section 2.1). Information is stored in metadata repositories (using the Sesame[4] triple store), databases (using MySql) and digital artefact repositories (using an NFS filesystem).

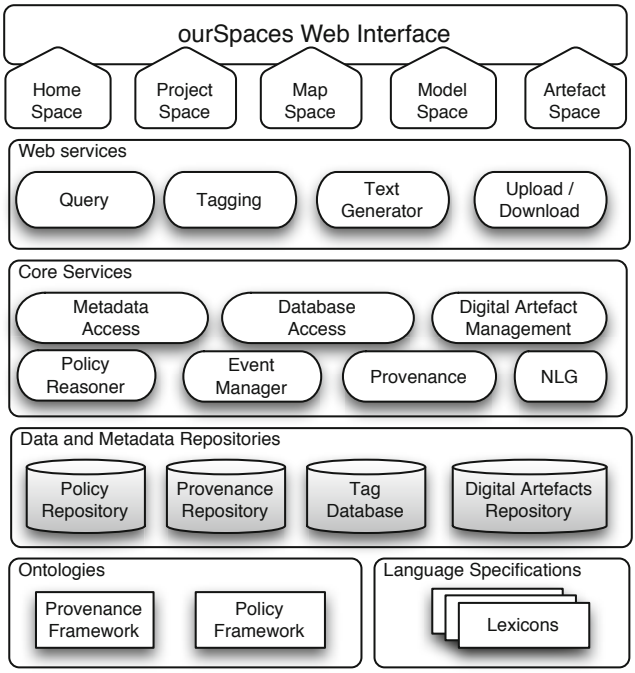


Fig. 2. The *ourSpaces* architecture

The *ourSpaces* architecture implements a number of core and Web services for creating, editing and querying data, metadata and digital artefacts. These include a natural language service to support browsing and querying data, a policy reasoning service and a service used to download and upload digital artefacts.

The *ourSpaces* user interface was developed using Web technologies such as Java Server Pages[5], JavaScript[6] and jQuery[7]. The interface is structured around the concept of a "space" (shown in Figure 2 - top), designed as a means to link, browse and share specific categories of data resources with other users. For example, the project space is used to manage a research project, the model space for handling simulation models, and the map space for browsing geospatial information.

[4] http://www.openrdf.org/
[5] http://www.oracle.com/technetwork/java/javaee/jsp/index.html
[6] http://www.ecma-international.org/publications/standards/Ecma-262.htm
[7] http://jquery.com/

In the remainder of this section we focus on two main components of the system where Semantic Web technologies were used: the provenance framework and the policy reasoning service.

2.1 Provenance Framework

We have designed and developed an extensible ontological framework for capturing the provenance of the research process based on the requirements highlighted in section 2. In order to describe and uniquely identify entities (such as artefacts, people, locations) and to make explicit relations between entities we followed the linked data principles [9].

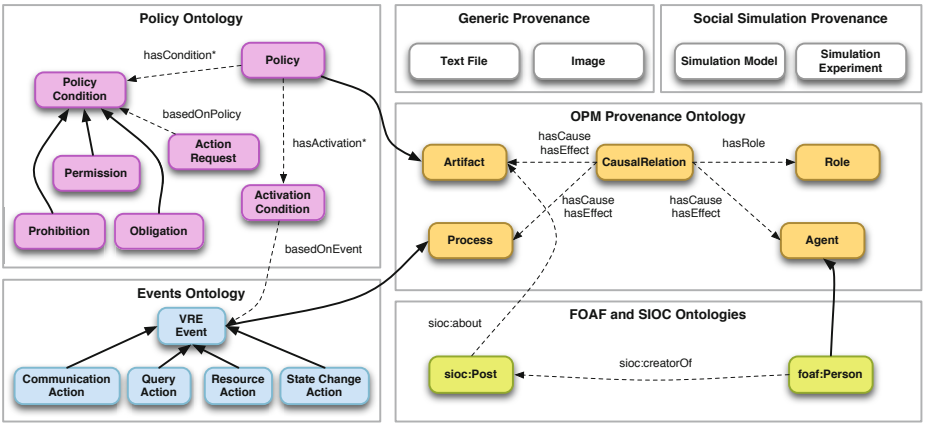


Fig. 3. *ourSpaces* ontological framework

At the heart of *ourSpaces* (and thus, our provenance framework) is an OWL representation of the Open Provenance Model [10]. This ontology defines the primary entities of OPM as well as the causal relationships that link them (see Figure 3, OPM Provenance Ontology). OPM is a generic solution and as a result, our framework supports additional domain-specific provenance ontologies that are created by extending the concepts defined in the OPM ontology with domain-specific classes. For example, in a social simulation domain ontology (see Figure 3, Social Simulation Provenance), one might have a `Simulation Model` as a type of artefact and a `Simulation Experiment` as a type of process. To date we have developed a number of domain-specific provenance ontologies describing aspects of Human Geography and Social Simulation. Using these ontologies it is possible, for example, to describe a physical research activity (e.g. an interview) as an `opm:Process`, and how such an activity causes an `opm:Artifact` to be generated (the interview notes).

Based on the requirements from our case study groups, the provenance framework should not only capture information regarding artefacts and processes, but must be able to situate these alongside people and their associated organisational structures. Friend-of-a-Friend[8] (FOAF) is an established RDF vocabulary for describing people

[8] `http://www.foaf-project.org/`

and their social networks and we have opted to utilise this within our framework; a `foaf:Profile` is thus a subclass of `opm:Agent`. Several FOAF profiles are visible in Figure 1, as contacts of the user (My Contacts). Organisational structures such as projects or employer institutions can also be defined, and users within *ourSpaces* may belong to several projects or groups. An event ontology (Figure 3 - lower left corner) was introduced to describe events taking place in the system and to allow the policy framework to operate (see section 2.2 for more details).

Another requirement was to capture the provenance of online communication within the social network. However, the current OPM specifications support limited information about the relationship between a person (`opm:Agent`) and the research process (`opm:Process`). As a result, we have integrated the social networking vocabulary SIOC[9] (Semantically-Interlinked Online Communities) within our provenance framework. The SIOC ontology is designed to enable the integration of online community information by providing a model to express user-generated content such as posting a message in a blog or posting a comment. Using this vocabulary, traditional artefact and process-driven provenance can be extended to incorporate social data. For example, a collaborator (defined with `foaf:worksWith`) could post a comment (`sioc:Post`) about some artefact (e.g. `prov:Paper`) uploaded by a colleague asking for some clarification about the method used to generate the data.

2.2 Policy Reasoning

Projects or individual users in *ourSpaces* may have different metadata requirements and access restrictions associated with their data. For example, a user may impose certain access constraints on digital artefacts that he/she owns, e.g. an artefact may only be accessible to users who are members of a particular project and who contributed towards creation of the artefact (i.e. were named as a co-author). As a result, there is a need for a framework to support reasoning about policies within the VRE. We have thus extended our provenance framework to define such policies as a combination of obligations, prohibitions or permissions.

We have combined the existing OWL binding of the Open Provenance Model with an OWL ontology (inspired by the work of Sensoy et al. [11]) defining the concepts introduced above. An extract of the provenance policy ontology is shown in Figure 3 (Policy Ontology). Moreover, we make use of the SPIN ontology[10] to support the use of the SPARQL query language to specify rules and logical constraints necessary to reason about policies. The SPIN ontology allows SPARQL queries to be represented in RDF and associated to classes in an ontology using two pre-defined description properties. `spin:constraint` can be used to define conditions that all members of a class must fulfil; `spin:rule` can be used to specify inference rules using SPARQL CONSTRUCT, DELETE and INSERT statements. An example of a `spin:rule` is presented in Figure 7. In our ontology a policy is a combination of `PolicyCondition` instances described by the property `hasCondition`*. Each condition can be defined as an Obligation, Prohibition or Permission depending on the nature of the policy. We

[9] `http://sioc-project.org/`
[10] `http://spinrdf.org/spin.html`

define a condition as a `spin:Construct` query describing its logic in the form of an *if-then* statement where *if* is represented by the WHERE block of the query and *then* by the CONSTRUCT block of the query (see Figure 7). Once processed by the SPIN reasoner a *spin:Construct* can assert a new `ActionRequest` instance which is constructed as part of the query, such as the `NLGRectractionRequest` in Figure 7. A policy in our ontology also has one or more `ActivationCondition` instances describing the activation condition of the policy via a `spin:Construct` query. As a result of an activation, the `spin:Construct` query asserts a new `PolicyActivation` instance. A `PolicyActivation` links a specific policy instance to the event that activated the policy, e.g. a resource action `UploadResource`.

When an activity is detected in the system, the event manager initiates a *policy session*. The *PolicyReasoner* checks if any of the policies stored in the policy repository can be activated by running the SPIN reasoner against the `spin:rule` instances associated with the policies and stores the outcome of the activation in the *policy session*. In order to reason about obligation, permission or prohibition conditions we require a reasoning mechanism able to check conditions over a provenance graph. This can be seen as a semantic matchmaking problem where a functional description of a condition is matched to a subset of a provenance graph. This is done by evaluating each condition defined as a `spin:rule`. For an obligation, conditions have to be met; for a prohibition, the condition cannot be met; and for a permission, the condition might (or might not) be met.

Using this approach in *ourSpaces* we were able to implement a policy for use by one of the project teams using the system. The policy specifies the kind of metadata required for artefacts that will eventually be archived to the UK social science data archive UKDA[11]. More specifically, the policy is created by the PI of the project and it is addressed to its members. The policy is activated when a person uploads an artefact. The required metadata vary for each artefact type, e.g. the policy activated by upload of a paper consists of three obligation conditions specifying that the title, author and date of publication are required.

3 Interaction with Semantic Web Data

Publishing data according to the linked data principles typically involves three main steps: choosing URIs and vocabularies, generating links and creating associated metadata [9]. Smith [12] states that it is challenging for non technical users to create and consume semantic metadata and this is considered to be a major issue while creating user interfaces. In this section we describe solutions integrated within the *ourSpaces* VRE to support the creation and visualisation of metadata.

3.1 Creating Semantic Web Data

We have developed a web interface to make creation of metadata by the users of the VRE as intuitive as possible, by allowing them to utilise a traditional web form and by automatically generating metadata were possible.

[11] `http://www.data-archive.ac.uk/`

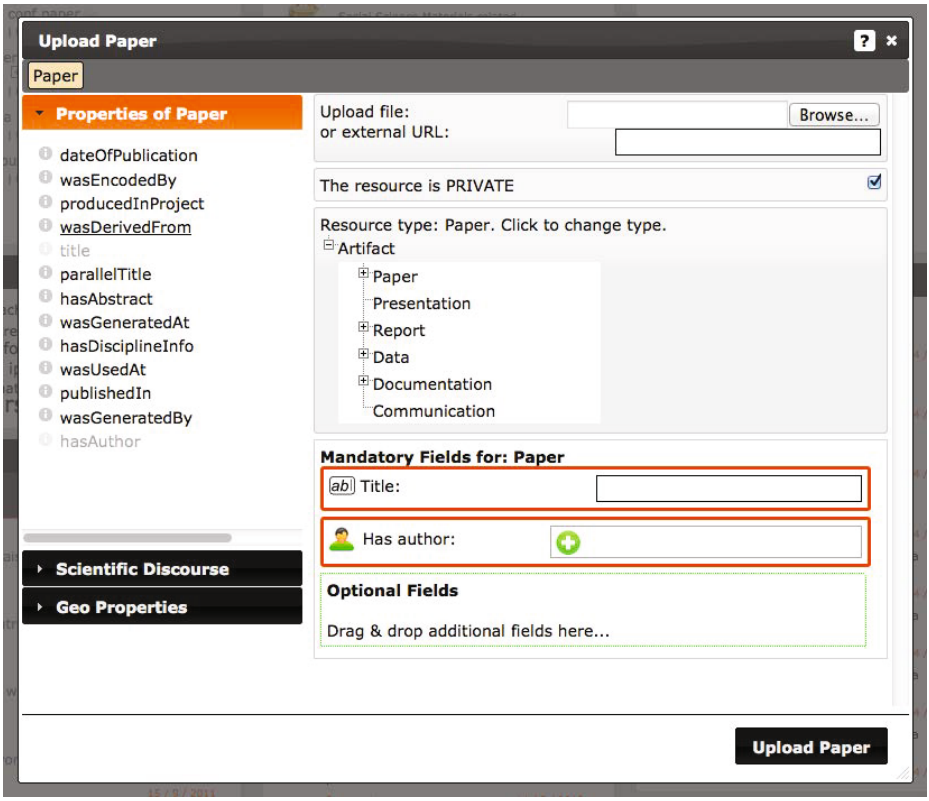


Fig. 4. A screenshot of the metadata creation form in *ourSpaces*

Consider the example where a user would like to upload into the VRE a new journal article he has published in order to share it with other researchers. By opening the upload form (see Figure 4), the user will be asked to specify the type of artefact he is uploading by selecting from a tree-like structure (see Figure 4-centre: *Resource type*). The tree is dynamically generated by processing the ontologies described in section 2. Once the user has selected an artefact type from the tree, the properties associated with it are displayed in the form (see Figure 4-left: *Properties of Paper*), by processing the appropriate ontology. Mandatory properties are automatically added depending on the cardinality defined in the ontology. Other properties might also be required depending on relevant policy activations. This can occur while opening the form or while selecting properties and entering values. The user is also able to select additional properties from a list. When inputting the values of a property, the upload form also guides the user as to what type of information should be entered by looking at the *rdf:range* axioms associated with the property. If the range of a property is an object (e.g. a `Person` for the `hasAuthor` property), the interface will use an autocomplete search functionality to help the user find an existing object from the repository.

We have also developed methods to support the creation of semantic links within communication items such as messages, comments and posts. For instance, when

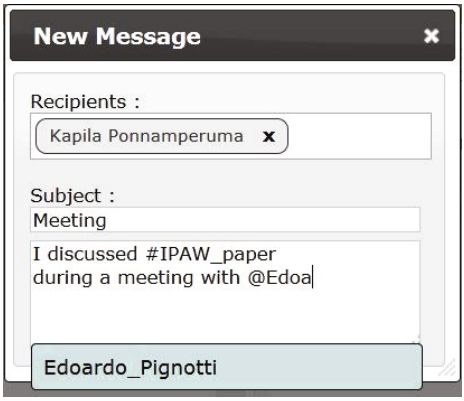


Fig. 5. Using # and @ tags

writing a message to a colleague, a user can refer to a person or an artefact in the system, by using @ (for people) or # (for artefacts) tags in combination with an autocomplete search function which returns instances from the repository. Examples of such links are illustrated in Figure 5.

3.2 Metadata Visualisation

In order to allow users to browse metadata, we have incorporated different metadata visualisation modalities including a natural language generation interface, a space-based interface, and a graph-based interface. The behaviour of these interfaces can be controlled by the policy framework in order to adapt to the specific preferences of a user or a project. We will now describe each of these visualisation modalities in turn.

Natural Language Generation Interface
We have developed a service enabling users to generate short textual descriptions of the resources stored in the *ourSpaces* repository. This service translates RDF statements into English sentences, based on the approach described by [13]. To generate the description of a particular RDF resource, this service queries the metadata repository with the ID of that resource to retrieve all related statements. A local model is then built from that list of statements, representing the information about the resource (see Figure 6 for an example of such a model).

This model is then used by the NLG service that converts its axioms into plain text, using the appropriate *language specification* files. These files (encoded in XML) describe how axioms should be translated in English. Each file represents a particular property in the ontology and contains linguistic information about how to structure the sentence corresponding to that property (e.g. syntactic category, verb, source and target). In the example shown in Figure 6, the *Transcript* artefact has a *producedInProject* property with a value of *E. coli O157 risk*. The language specification of *producedInProject* will indicate that this information must be rendered as: “The transcript was

produced in the project E. coli O157 risk". Those files use a dependency tree structure to represent the relationships between the different syntactic units of the sentence. This allows properties with similar syntactical structures to be aggregated together in the text. The final stage of linguistic realisation is carried out using the SimpleNLG realiser [14], which converts abstract representations of sentences into actual text using rules of grammar (morphology and syntax).

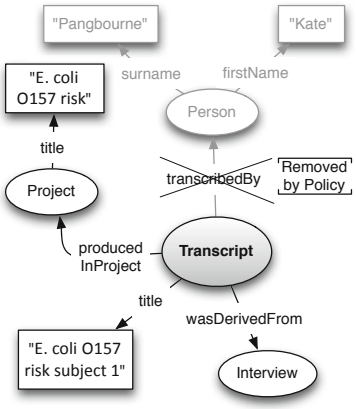


Fig. 6. Internal representation of a resource described by the NLG service

```
CONSTRUCT {
   _:b0 a pol:NLGRetractionRequest .
   _:b0 pol:requestAboutResource ?this .
   _:b0 pol:requireObject ?rmObject .
   _:b0 pol:requirePredicate pggen:transcribedBy .
   _:b0 pol:requireSubject ?this .
}
WHERE {
   ?this a pggen:Transcript .
   ?this pggen:transcribedBy ?rmObject .
   ?policy pol:basedOnEvent ?nlgAction .
   ?policy pol:activePolicy :NLGTranscriptPolicy .
   ?nlgAction a vre:NLGAction .
   ?nlgAction  opmv:Used ?this .
   ?nlgAction opmv:WasControlledBy ?person .
   NOT EXISTS {
      ?this pggen:producedInProject ?project .
      ?project project:hasMemberRole ?role .
      ?role project:roleOf ?person .
   } .
}
```

Fig. 7. Example of policy condition

To integrate this mechanism within *ourSpaces*, we developed a RESTful service that requests a textual description when a user hovers the mouse pointer on top of a resource name in the interface. The generated text is presented to the user in a popup window, who then has the option to further explore the repository by clicking anchors to related resources. Every time a user clicks on an anchor, the service is invoked with a new resource ID and the generated text is inserted in the same popup window. For example in Figure 1, the user generated the description of *the transcript E. coli 0157 risk subject 1* and then requested more information about a related resource (*the project E. coli 0157 risk*).

ourSpaces is designed to support collaboration within multidisciplinary research groups. Members of our case study groups have often stressed that people from different backgrounds tend to have different information presentation preferences. Empirical evidence also suggests [1] that there is a need to adapt information interfaces to users and their context. This can include information about the users themselves, the task they are currently performing including information about the project they are working on. To address this issue, we used the policy framework described in section 2.2 to choose between data presentation strategies (e.g. graphical or textual visualisation to explore the provenance graph), as well as controlling the content of the text generated by the NLG service ([15]). The latter is a kind of rule-based content determination ([16]).

For example, the principal investigator of a project might want to protect the identity of the person that transcribed an artefact from users that are not members of that project. This preference can be expressed by constructing a policy enforcing a rule similar to the one shown in Figure 7. This rule triggers an action request to remove the *transcribedBy* property, if the user visualising the description of a *Transcript* is not a member of the project where that artefact was produced. The corresponding nodes are removed from the model used by the NLG service, as shown in Figure 6. The use of this policy is illustrated in Figure 1, where the private information has been omitted from the textual description. The policy framework can also be used to assert information to the model used for NLG. By retrieving information deeper in the repository (i.e. metadata about related resources), a more complete description of an artefact can be generated. In this manner, the NLG service combined with the policy framework allows the system to generate descriptions aligned to the user's context and preferences.

Space and Graphical Visualisation Interfaces
As mentioned earlier in this paper, a "space" acts as a container to provide access to information about a specific resource or a category of resources. In order to generate a space a number of SPARQL queries are performed over the provenance repository, extracting relevant sections of the RDF graph. Different metadata visualisation modalities can be utilised within a space (table, graph-based and NLG-based) all of which may offer links to other resources. By exploring such links, the focus of the space changes thus providing the user with a tool to navigate the graph. To illustrate this, an example of an *Artefact Space* is presented in Figure 8 showing data about a *focus group* resource. The properties of the artefact are presented in a table (see Figure 8 - top left). The same information is also described using natural language (see Figure 8 - top right). Related artefacts are also presented. Figure 8 (bottom left) illustrates the graphical interface used to visualise metadata about the provenance of the artefact. In this example the *focus group* is highlighted and the immediate provenance properties and values are displayed. By clicking the + button, the user can expand the graph in order to explore additional provenance information. Moreover, by hovering the mouse pointer over processes or artefacts the user is presented with additional information which is rendered in plain text by the NLG service.

4 Lessons Learnt from Deploying the VRE

ourSpaces has been deployed for use with three interdisciplinary case study groups: a 'project' (all researchers working on the ESRC-NERC funded RELU programme's 'Reducing Escherichia coli O157 risk in rural communities' project); a research 'community' (all members and affiliates of the Aberdeen Centre for Environmental Sustainability - ACES) and a group of agent based social simulation modellers. Members of these groups have been encouraged to use *ourSpaces* and to contribute to its continued development by reflecting on how it has been used, and how they see that it might be developed to enable deeper research integration.

The VRE has been available on-line since September 2009. To date, there are 254 *foaf:profiles* defined in *ourSpaces* of which 183 are registered users. Non registered pro-

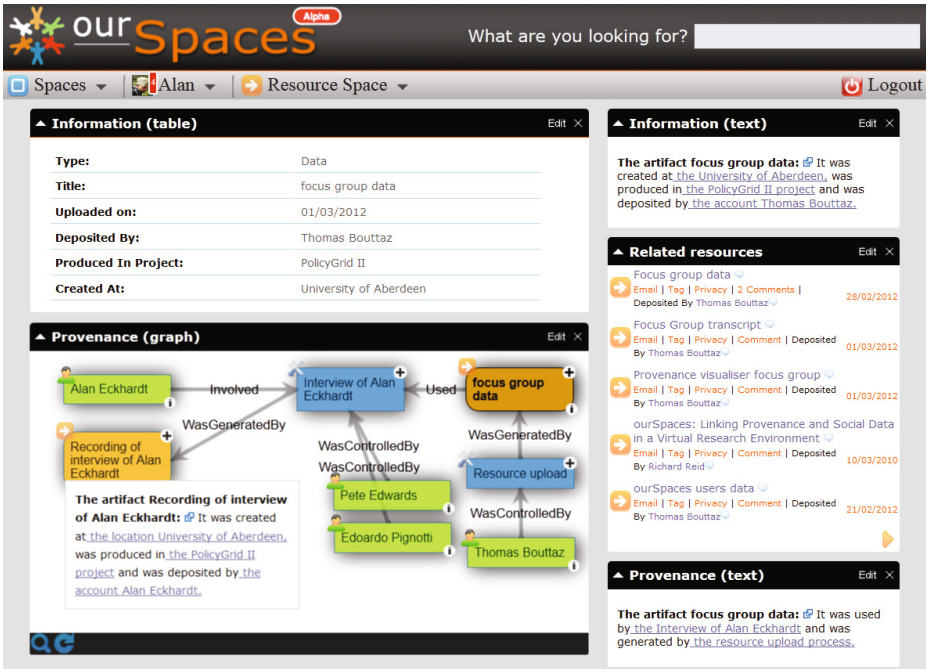


Fig. 8. A screenshot of the *Artefact Space*

files have been created by the system while specifying authors of documents. The social network in the VRE is composed of 204 links (*foaf:knows*) between user accounts. Users created 49 projects and sub-projects and 92% of the accounts in *ourSpaces* are members of at least one project. Users have also uploaded 435 research artefacts. The *ourSpaces* metadata repository contains 14680 triples describing 4388 distinct entities. To date, 63 distinct classes and 105 distinct properties are used to describe entities in the repository, utilising 33% of the classes and 40% of the properties defined in the supporting ontologies.

The two primary sources of data for our ongoing evaluation of the system are: a) the repository containing metadata about resources, people, events, projects, etc. and b) the mysql database containing user account information and system logs. We have also conducted interviews and focus groups with our case study groups in order to gather evidence on user perspective and feedback. In the remainder of this section we will use the outcome of these data gathering exercises in order to illustrate some of the lessons learnt during the deployment of the VRE.

4.1 Deployment Issues and Lessons Learnt

One of the most significant difficulties was the unwillingness of users to provide metadata about artefacts. In the early stages of *ourSpaces*, users were required to provide a great deal of metadata about research artefacts in order to guarantee a detailed metadata

record. The result was that few users went through the effort of providing such metadata and only a small number of artefacts were uploaded. Following feedback from users we adopted a more relaxed approach, where very few mandatory fields were required and the users themselves had the option to choose which metadata to add to the artefact. This resulted in more artefacts being uploaded at the cost of producing a much sparser metadata record. To illustrate this issue we now present some summary data collected from our metadata repository. As illustrated in section 3.1, the user is required to select the type of artefact that he/she is uploading and mandatory fields are displayed in the form depending on the class selected. Types of artefact are shown on a tree-like structure, where *Artefact* is the root class and more specialised types are presented up to two levels down the class hierarchy. From our analysis, 20% of the artefacts in *ourSpaces* have been associated with the root class, 71% with subclasses of *Artefact* and 9% with classes at the next level in the hierarchy.

We aimed to solve the problem of sparse metadata by allowing people (with the right authority) to define policies in *ourSpaces* to specify the mandatory information required when uploading a research artefact. In this way, the request for additional information originated with a person rather than the system, e.g. the principal investigator of a project. After introducing policies into one of the projects in *ourSpaces*, the average number of RDF triples used to describe each research artefact increased from 9 to 32. However, policies and particularly the SPIN reasoner require additional computational resources, resulting in a delay when a user is using a web form. The time taken by SPIN to process each policy depends on the precise nature of that policy. Some, such as logging in, do not require much data to be evaluated. Others, such as uploading an artefact, require a large provenance graph to be evaluated.

We have analysed the logs from the policy reasoning service in order to determine the performance of the reasoner. The hardware used for the deployment of the *ourSpaces* services and repositories consists of three Sun Fire X4100 M2 with two dual-core AMD Opertron 2218 processors and 32 Gb of memory. Based on 2895 runs of the reasoner logged by the system the average time to run a policy was 1993ms. Miller [17] and Card [18] argue that system response times of less than ten seconds do not compromise the user's attention on the current task. However for delays greater that one second, it is necessary to indicate to the user that the system is busy. Based on the result from the analysis of the logs we determined that time taken to reason about a policy was acceptable without compromising the overall performance of the system. However, we had to make use of the AJAX spinning wheel widget to inform the user that the system was performing a task. A similar analysis has also been conducted for the use of policies by the NLG service. In spite of the overhead associated with the use of the policy reasoner, text is generated and appears within 200ms.

Policy reasoning also presents a cost in terms of data storage. The inferences generated by policy reasoning could be stored in the RDF repository in order to provide additional provenance about user actions. However, it is not always necessary to do this in practice as recording very fine grained provenance may not always be useful. For example, policy reasoning is triggered every time the user makes a change in a field during the upload of an artefact but it is not necessary to record the provenance of each individual action. On the other hand, knowing that a user had twenty failed log-in attempts

with an incorrect password could indicate a potential security issue and is something worth registering for later inspection.

Based on the feedback from our users we have discovered that the graphical interface for visualising metadata (section 3.2) served a useful function as a means to validate the metadata uploaded via the form based interface. This was especially useful as the UKDA documentation policy required them to provide detailed information about the methods used for generating a research artefact. The graphical interface was also used by representatives of the UKDA to review the data (and metadata) uploaded by project members. Screen-grabs of the provenance graphs generated by our system have been used by the UKDA as part of their internal documentation describing the project archive.

Maintaining the *ourSpaces* VRE and introduction of new user requirements often requires changes to the underlying data structures. From the beginning, *ourSpaces* has used both an SQL database and an RDF store, with the SQL database used to log user activities for monitoring and security purposes. When the policy reasoning facilities were subsequently introduced, it was necessary to change the representation of user activities so that the reasoner could infer policy activations. As a result, the data structure describing activities had to be changed from a relational database to RDF. This has allowed us to describe system activities in terms of artefacts, processes and agents and to include them as part of the wider provenance graph.

A crucial part of maintenance of the system is to take into account the requirements of new user groups. When a new group joins *ourSpaces*, it is normal to expect that they might have their own way to describe research artefacts and processes. The *ourSpaces* provenance framework can be easily extended in order to accommodate new domain-specific provenance concepts. This issue was detected very early during the development of *ourSpaces* and we therefore designed the system in such a way that new domain ontologies could be integrated into the system without the need to change the source code. Our implementation of the policy framework also allows new policies to be integrated into the system without the need for alterations to the underlying code. Although we don't yet have a specific tool for designing policies, a standard ontology editor can be used to design the individual policy ontologies.

5 Conclusions

In this paper we have introduced the *ourSpaces* virtual research environment focusing on three elements which make use of Semantic Web technologies - a provenance representation framework, a collection of services for creating and visualising metadata and a policy reasoning service.

The ontological support in *ourSpaces* allows us to capture entities such as artefacts, people and processes and to include links between them. This 'linked data' makes certain aspects of information discovery and presentation possible within the *ourSpaces* environment. For example, the concept of a "space" (home, project, etc.) would not be effective unless it was possible to identify the entities linked to the resource featured in the space such as related resources, people involved, communication activities, etc. Linked data also allows components such as the natural language visualisation service to exploit this model to allow users to explore the provenance graph.

The policy and provenance framework provides a real benefit in terms of adaptability of the system. As was discussed in section 4, new policies and ontologies can be introduced by changing the configuration files without having to change the Java code. Moreover, the declarative nature of policies allows the introduction of new logic into the system even by users that are not familiar with the underlying VRE source code.

In future, we plan to implement a model space, which would enable users to run their own simulation experiments using standard simulation environments such as Repast[12] or Netlogo[13]. This space will enable users to explore the provenance associated with simulation models and to support the reproducibility of simulation experiments.

Future plans also include a personalised thesaurus service, which would accommodate the vocabulary differences of users and disciplines. This service could be used by the existing NLG and search services to deal with user or discipline specific terminology differences.

Acknowledgments. This work is supported by the UK Economic & Social Research Council (ESRC) under the Digital Social Research programme; award RES-149-25-1075.

References

1. Butterworth, A., Reimer, T.: Virtual research environment collaborative landscape study. Technical report, JISC (2010)
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American 284(5), 34–43 (2001)
3. Heinis, T., Alonso, G.: Efficient Lineage Tracking for Scientific Workflows. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD 2008), pp. 1007–1018. ACM (2008)
4. Pignotti, E., Edwards, P., Reid, R.: A Multi-faceted Provenance Solution for Science on the Web. In: McGuinness, D.L., Michaelis, J.R., Moreau, L. (eds.) IPAW 2010. LNCS, vol. 6378, pp. 295–297. Springer, Heidelberg (2010)
5. Pignotti, E., Edwards, P.: Using web services and policies within a social platform to support collaborative research. Working Notes of AAAI 2012 Stanford Spring Symposium on Intelligent Web Services Meet Social Computing (March 2012)
6. Roure, D.D., Goble, C., Stevens, R.: The design and realisation of the virtual research environment for social sharing of workflows. Future Generation Computer Systems 25(5), 561–567 (2009)
7. Aastrand, G., Celebi, R., Sauermann, L.: Using linked open data to bootstrap corporate knowledge management in the organik project. In: Proceedings of the 6th International Conference on Semantic Systems, I-SEMANTICS 2010, pp. 18:1–18:8. ACM, New York (2010)
8. Kreiser, A., Naurex, A., Bakalov, F.: A web 3.0 approach for improving tagging systems. In: Proceedings of the International Workshop on Web 3.0: Merging Semantic Web and Social Web (in Conjunction with the 20th International Conference on Hypertext and Hypermedia 2009), Torino, Italy, vol. 467 (June 2009)
9. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. International Journal on Semantic Web and Information Systems, IJSWIS (2009)

[12] `http://repast.sourceforge.net/`
[13] `http://ccl.northwestern.edu/netlogo/`

10. Moreau, L., Freire, J., Futrelle, J., McGrath, R.E., Myers, J., Paulson, P.: The Open Provenance Model: An Overview. In: Freire, J., Koop, D., Moreau, L. (eds.) IPAW 2008. LNCS, vol. 5272, pp. 323–326. Springer, Heidelberg (2008)
11. Şensoy, M., Norman, T.J., Vasconcelos, W.W., Sycara, K.: OWL-POLAR: Semantic Policies for Agent Reasoning. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 679–695. Springer, Heidelberg (2010)
12. Smith, D.A., Popov, I., schraefel, mc.: Data picking linked data: Enabling users to create faceted browsers. In: Web Science Conference 2010 (March 2010) Event Dates: 26-27
13. Hielkema, F.: Using Natural Language Generation to Provide Access to Semantic Metadata. PhD thesis, University of Aberdeen (2010)
14. Gatt, A., Reiter, E.: Simplenlg: a realisation engine for practical applications. In: Proceedings of the 12th European Workshop on Natural Language Generation, ENLG 2009, Stroudsburg, PA, USA, pp. 90–93. Association for Computational Linguistics (2009)
15. Bouttaz, T., Pignotti, E., Mellish, C., Edwards, P.: A policy-based approach to context dependent natural language generation. In: Proceedings of the 13th European Workshop on Natural Language Generation, Nancy, France, pp. 151–157. Association for Computational Linguistics (September 2011)
16. Reiter, E., Dale, R.: Building natural language generation systems. Cambridge University Press, New York (2000)
17. Miller, R.B.: Response time in man-computer conversational transactions. In: Proceedings of the Joint Computer Conference 1968, pp. 267–277. ACM, New York (1968)
18. Card, S.K., Robertson, G.G., Mackinlay, J.D.: The information visualizer, an information workspace. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Reaching Through Technology CHI 1991, pp. 181–186. ACM, New York (1991)

Embedded $\mathcal{EL}^+$ Reasoning on Programmable Logic Controllers*

Stephan Grimm, Michael Watzke, Thomas Hubauer, and Falco Cescolini

Siemens AG, Corporate Technology, Munich, Germany

Abstract. Many industrial use cases, such as machine diagnostics, can benefit from *embedded reasoning*, the task of running knowledge-based reasoning techniques on embedded controllers as widely used in industrial automation. However, due to the memory and CPU restrictions of embedded devices like programmable logic controllers (PLCs), state-of-the-art reasoning tools and methods cannot be easily migrated to industrial automation environments. In this paper, we describe an approach to porting lightweight OWL 2 EL reasoning to a PLC platform to run in an industrial automation environment. We report on initial runtime experiments carried out on a prototypical implementation of a PLC-based $\mathcal{EL}^+$-reasoner in the context of a use case about turbine diagnostics.

1 Motivation

Embedded controllers are extensively used in industrial environments for operating and monitoring technical machinery. They can be placed near the underlying machine's sensors and actuators to perform local computation tasks on-site with quick reaction times. Much of the computation required for process control or condition monitoring on industrial facilities is thus performed on computational devices that are embedded in the surrounding field in a decentralized manner, which is in contrast to the central computation performed on PCs in areas such as business applications. When embedded controllers run knowledge-based systems that apply the computational techniques of automated reasoning, we speak of *embedded reasoning*.

One particular use case for embedded reasoning is on-site diagnostics of industrial facilities based on diagnostic knowledge models for technical devices. There, embedded controllers can perform reasoning on sensor data in the context of diagnostic background knowledge to detect a machine's faulty behavior or to trace the causes thereof. Single diagnostic results reasoned over local device models can be combined to yield an overall diagnosis for the whole facility under investigation. An example scenario is reactive diagnostics for steam and gas turbines in electrical power plants, where early detection of anomalies in running a plant helps avoid costly downtimes and repair of turbine machinery.

In contrast to business applications that run on PCs, knowledge-based modeling and reasoning techniques, such as those used in the Semantic Web, are

* This research was funded in part by the German Federal Ministry of Education and Research under grant number 01IA11001.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 66–81, 2012.

not readily available in an automation environment. Embedded controllers are subject to tight limitations on computational power or memory size and often run proprietary operating systems for which no standard reasoning tools are available, as reported in [11]. There, it was also observed that standard implementations of state-of-the-art reasoning algorithms designed for a PC environment cannot easily be migrated to embedded controllers due to their hardware restrictions. Hence, there is the need for porting reasoning techniques to embedded environments.

In the area of industrial automation, programmable logic controllers (PLCs) are the most widespread type of embedded controller used for many real-time automation tasks. Unlike general purpose computers, PLCs are designed for control of industrial machinery and employ a processing scheme based on fixed length execution time cycles to meet realtime requirements. On the other hand, the Web Ontology Language (OWL) and its underlying reasoning services are well suited for representing structural diagnostic models of technical devices and their dependencies. Especially the OWL 2 profiles, which provide restricted language variants with faster computation, are well suited for use in resource-constrained embedded systems. Hence, it would be beneficial to port (light-weight) OWL 2 reasoning to PLCs in order to provide for on-site processing of knowledge models in the context of machine diagnostics and many other industrial use cases.

The implementation of state-of-the-art OWL reasoning on a PLC platform, however, has various difficulties. On the one hand, the typical methods used for OWL reasoning, such as tableaux or consequence-driven procedures, are designed to use dynamic data structures like tree models or concept mappings that are continuously expanded, whereas on PLCs only static memory management is available, and restricted to very limited size. Moreover, the dynamic runtime behavior of standard reasoning algorithms operating on a set of tableau- or consequence-like rules over an indeterminate number of iterations does not fit the strictly cyclic processing pattern that PLCs follow.

In this paper, we present an approach to porting consequence-driven ontology classification in the OWL 2 EL language profile, as e.g. described in [1,9], to PLC platforms in the context of an industrial diagnostics use case. We propose an architecture for using PLCs in combination with standard OWL tools hosted on a PC environment, and we show how to overcome the architectural discrepancies between standard implementation platforms for such procedures and the PLC world. In particular, we present the use of a compact and efficient axiom representation based on fixed length data structures, and we introduce an interruption-safe saturation mechanism that fits a PLC's cyclic processing paradigm. We also report on initial runtime experiments carried out on our prototypical implementation of a PLC-based embedded OWL 2 EL reasoner covering the description logic $\mathcal{EL}^+$ to show the feasibility of our approach. This also includes a formulation of a turbine diagnostic problem in terms of an $\mathcal{EL}^+$ ontology and the use of classification for deriving diagnoses. By this, we contribute to making Semantic Web reasoning technology available on industrial automation platforms dominated by the PLC paradigm.

2 Preliminaries

2.1 Programmable Logic Controllers

In the automation domain, *programmable logic controllers* (PLCs) are a flexible means for solving a control problem. In contrast to control wiring contactors and relays, a PLC uses a specific program to realize a control task. PLCs are the central part of today's automation solutions for control of machines or plants. They are networked to central and/or distributed I/O modules connected to sensors and actuators, display devices for monitoring and operation, as well as a programming device or SW development environment for programming and configuration [5].

In order to behave like hardwired logics consisting of contactors and relays, where logical operations are effectively executed in parallel, a PLC executes its sequential user program cyclically. Within each execution cycle, a PLC first stores a snapshot of its current input signals in a process image input table in the CPU memory. The CPU executes the user program step-by-step, one command at a time. During command execution, signal states are taken from the process image input table, processed and stored in the process image output table. At the end of an execution cycle, when the user program has finished, a PLC sets output signals according to their values in the process image output table. The shorter the cycle time of the PLC user program, the more frequently input signals are read and output signals are set. PLC commands can be clustered into binary logic operations (e.g. AND, OR), memory functions (e.g. bit assignment), move functions (e.g. register load/store from/to memory), counters and timers [4].

The work presented in this paper is based on Siemens' SIMATIC automation system. The SIMATIC automation system is a family of different products [5] built around the SIMATIC S7 controller. There are several PLC types available, addressing a range of performance and availability requirements. Usually, a power supply unit, CPU module and I/O modules are installed in a mounting rack and form a *station*. The SIMATIC DP distributed I/O system allows for input/output modules to be installed nearby a machine, connected to a PLC by means of a PROFIBUS network. SIMATIC HMI (Human Machine Interface) provides a number of different products for visualization, ranging from simple text displays to interactive touch screen panels. SIMATIC NET provides networking of all SIMATIC stations for data exchange, programming and configuration. OPC[1] is an interface standard for communication between several stations or to PCs. STEP 7 is the standard tool for configuration and programming of the user program in any of the available languages (Ladder Logic (LAD, a representation similar to relay logic diagrams), Function Block Diagram (FBD), Statement List (STL, an Assembler-like language) or Structured Control Language (SCL, a high-level language similar to Pascal)).

In addition to hardware-based PLCs, SIMATIC WinAC provides software that emulates a PLC on a standard Industrial PC (IPC) running Microsoft Windows.

[1] http://www.opcfoundation.org/

2.2 Ontologies and the $\mathcal{EL}^+$ Description Logic

In the area of knowledge-based systems and the Semantic Web, *ontologies* are the key artifact for representing and reasoning about knowledge of a specific domain, such as turbine machinery. Compared to subsymbolic approaches for capturing domain knowledge, ontologies allow for an explicit representation of domain concepts and their interrelations as well as for automated reasoning based on the clear semantics of logic. In this work, we build on the prominent *OWL Web Ontology Language* which is semantically founded on the description logic (DL) formalism [2]. In its second version, OWL comes with a number of so-called profiles that offer language variants with reduced expressivity and better computational properties. In particular, we use (a part of) OWL 2 EL [13], which is based on the tractable description logic $\mathcal{EL}^+$ [1]. Research has shown that the OWL 2 EL profile is especially well suited for representing diagnostic models, including the well-known SNOMED-CT[2] and GALEN[3] medical ontologies.

The basic building blocks for representing knowledge in $\mathcal{EL}^+$ are (atomic) *concepts*, such as *GasTurbine*, and *roles*, such as *hasComponent*. An $\mathcal{EL}^+$ *ontology* $\mathcal{O}$ is a set of concept inclusion axioms of the form $\mathit{GasTurbine} \sqsubseteq \mathit{Turbine}$ (stating that every *GasTurbine* is a *Turbine*) and role inclusion axioms of the form $\mathit{directlyControls} \circ \mathit{hasSubComponent} \sqsubseteq \mathit{controls}$ (stating that control is propagated over the partonomy of a system). The symbol $\sqsubseteq$ can be read as logical implication. The sets $\mathsf{N}_C^{\mathcal{O}}$ and $\mathsf{N}_r^{\mathcal{O}}$ denote the concept and role names that occur in an ontology $\mathcal{O}$ and form its *signature*. Complex concepts can be composed from simpler concepts and roles using the concept constructors $\sqcap$ and $\exists$. An example for a complex $\mathcal{EL}^+$ concept inclusion axiom is $\mathit{GasTurbine} \sqsubseteq \exists\, \mathit{hasComponent}\,.\,\mathit{Combustor}$, stating the fact that any gas turbine has a combustor as its component. For further details on $\mathcal{EL}^+$, see [2,13].

The main features that $\mathcal{EL}^+$ lacks compared to OWL 2 EL are concept disjointness by means of the bottom concept $\bot$, and nominals $\{o\}$, which allow expression of facts about particular instances of a concept. The main reasoning task in $\mathcal{EL}^+$ (and a central reasoning task for description logics in general) is called *classification*, it builds up a complete inferred subsumption hierarchy of all atomic concepts mentioned in an ontology $\mathcal{O}$. This can be understood as deriving deductively all concept subsumptions $A \sqsubseteq B$ that are a logical consequence of $\mathcal{O}$, expressed formally by $\mathcal{O} \models A \sqsubseteq B$, for $A, B \in \mathsf{N}_C^{\mathcal{O}}$

2.3 Consequence-Driven Reasoning in $\mathcal{EL}^+$

Classification of $\mathcal{EL}^+$ ontologies can be realized efficiently (i. e., in polynomial runtime) using a so-called consequence-driven reasoning approach, as described in [1] or [9]. Since our embedded reasoning method for PLCs also employs a consequence-driven approach, we give an introduction to consequence-driven reasoning for $\mathcal{EL}^+$, following the approach from [1], in this section.

[2] http://www.nlm.nih.gov/research/umls/Snomed/snomed_main.html
[3] http://www.openclinical.org/prj_galen.html

$$\textbf{(CR1)}\ \frac{C \sqsubseteq C_1}{C \sqsubseteq D}\ [C_1 \sqsubseteq D \in \mathcal{O}] \quad \textbf{(CR2)}\ \frac{C \sqsubseteq C_1 \qquad C \sqsubseteq C_2}{C \sqsubseteq D}\ [C_1 \sqcap C_2 \sqsubseteq D \in \mathcal{O}]$$

$$\textbf{(CR3)}\ \frac{C \sqsubseteq C_1}{C \sqsubseteq \exists r.D}\ [C_1 \sqsubseteq \exists r.D \in \mathcal{O}] \quad \textbf{(CR4)}\ \frac{C \sqsubseteq \exists r.C_1 \qquad C_1 \sqsubseteq C_2}{C \sqsubseteq D}\ [\exists r.C_2 \sqsubseteq D \in \mathcal{O}]$$

$$\textbf{(CR5)}\ \frac{C \sqsubseteq \exists r.D}{C \sqsubseteq \exists s.D}\ [r \sqsubseteq s \in \mathcal{O}] \quad \textbf{(CR6)}\ \frac{C \sqsubseteq \exists r_1.C_1 \qquad C_1 \sqsubseteq \exists r_2.D}{C \sqsubseteq \exists s.D}\ [r_1 \circ r_2 \sqsubseteq s \in \mathcal{O}]$$

Fig. 1. Completion rules for $\mathcal{EL}^+$

In a preprocessing step, the input ontology $\mathcal{O}$ is normalized into a semantically equivalent ontology $\|\mathcal{O}\|$ which contains only axioms of the form **(NF1)** $C_1 \sqsubseteq D$, **(NF2)** $C_1 \sqcap C_2 \sqsubseteq D$, **(NF3)** $C_1 \sqsubseteq \exists r.D$, **(NF4)** $\exists r.C_2 \sqsubseteq D$, **(NF5)** $r \sqsubseteq s$, and **(NF6)** $r_1 \circ r_2 \sqsubseteq s$ for atomic concepts C_1, C_2 and D, and roles $r, 2, r_1$, and r_2. Next, starting from the trivially true tautologies $C \sqsubseteq \top$ and $C \sqsubseteq C$ (for each concept name C in $\|\mathcal{O}\|$), the consequence-driven classification algorithm derives additional valid subsumptions that are also logical consequences of $\|\mathcal{O}\|$ based on the information stated explicitly in $\|\mathcal{O}\|$. This is done based on a set of so-called completion rules shown in Figure 1, which can be understood as follows: If the premise(s) above the horizontal line are known to be consequences of $\|\mathcal{O}\|$ and the axiom in square brackets is contained in the ontology $\|\mathcal{O}\|$, then the conclusion below the horizontal line is a valid consequence of $\|\mathcal{O}\|$ as well. Note that there is exactly one completion rule for each type of normalized axiom; the application of completion rule **(CR*i*)** is therefore guarded by the presence of an appropriate **(NF*i*)**-axiom in $\|\mathcal{O}\|$. When the completion process terminates (i. e. no more rules are applicable), all valid subsumptions have been derived.

The procedure outlined above is typically realized based on two mappings $S : \mathsf{N}_C^{\mathcal{O}} \mapsto \mathcal{P}(\mathsf{N}_C^{\mathcal{O}})$ and $R : \mathsf{N}_r^{\mathcal{O}} \mapsto \mathcal{P}(\mathsf{N}_C^{\mathcal{O}} \times \mathsf{N}_C^{\mathcal{O}})$ which make explicit the subsumptions derived so far[4]. For this, observe that the premises and conclusions of the completion rules only consist of two types of axioms, namely $C \sqsubseteq D$ and $C \sqsubseteq \exists r.D$. These axioms are represented in the mappings as follows: $C \sqsubseteq D$ corresponds to a mapping entry $D \in S(C)$, and $C \sqsubseteq \exists r.D$ is represented by $\{C, D\} \in R(r)$. Following the intuition given above, these mappings are initialized by $S(C) = \{C, \top\}$ for each concept name C in $\|\mathcal{O}\|$ and $R(r) = \emptyset$ for each role name r in $\|\mathcal{O}\|$. Testing the premises in the rules from Figure 1 then corresponds to lookups in the mappings S and R. Analogously, the conclusions correspond to the addition of elements to the respective mapping. The classification result of the original input ontology $\mathcal{O}$ can directly be read from S after termination.

The algorithm presented above has been implemented successfully in the CEL[5] reasoner. Naturally, such a practical implementation requires additional

[4] $\mathcal{P}(\cdot)$ denotes the powerset operator.

[5] http://lat.inf.tu-dresden.de/systems/cel/

considerations for minimizing memory usage and maximizing performance; in [3], an overview of the respective decisions made for the CEL system is given. Most recent developments for consequence-driven OWL 2 EL classification are reported in [9] for the ELK[6] reasoner system. One central optimization, the representation of entities using integer numbers, has also been employed for our embedded implementation.

2.4 Related Approaches to Embedded Reasoning

Although there are a variety of approaches for reasoning on embedded (or mobile) devices, we are not aware of any other research on implementing reasoning algorithms for PLCs or similarly constrained devices. The Pocket KRHyper system[7], for example, realizes DL reasoning via a hyper tableau calculus for first-order logic [12]. It is implemented as a Java 2 Mobile Edition[8] application and thereby overcomes any need to address hardware issues. On the downside, KRHyper can only be used on systems for which a Java Virtual Machine is available; PLCs and other automation systems do not fall into this category and their cycle-based paradigm makes an easy port unlikely. More closely related to the approach presented in this paper is [11], whose authors propose a RETE-based OWL 2 RL reasoner for an embedded system with a 400 MHz CPU and 64 MB of RAM that features a Linux operating system. Although standard DL reasoners such as Pellet[9] or FaCT++[10] cannot be used sensibly in this environment either, the availability of a standard operating system permits the authors to use standard programming paradigms. Interestingly, the authors also considered the possibility of using consequence-driven reasoning (namely CEL) on their platform; this was impractical due to the lack of an Allegro Common Lisp environment. In a very similar fashion, Bossam[11] uses the RETE algorithm to realize an OWL reasoner with a small memory footprint [8], but it requires a Java Virtual Machine just like the Pocket KRHyper system.

3 Industrial Application of Embedded $\mathcal{EL}^+$ Reasoning

In this section, we describe the use of embedded $\mathcal{EL}^+$ reasoning for industrial diagnostics, and we sketch a suitable architecture for such reasoning in automation environments.

3.1 Diagnostics of Technical Devices with Embedded Reasoning

The main motivation for our research on embedding reasoning on PLCs is industrial diagnostics. In particular, we consider a use case involving reactive diagnostic reasoning for steam and gas turbines. In the scenario considered here, a power

[6] http://code.google.com/p/elk-reasoner/
[7] http://mobilereasoner.sourceforge.net/
[8] http://www.oracle.com/technetwork/java/javame/index.html
[9] http://clarkparsia.com/pellet/
[10] http://owl.man.ac.uk/factplusplus/
[11] http://bossam.wordpress.com/

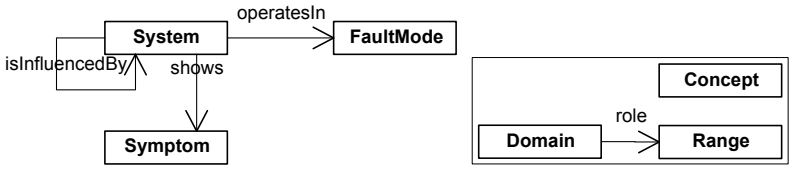


Fig. 2. A basic diagnostic model formalized in $\mathcal{EL}^+$ (c.f. [6])

generation command and control center (CCC) is responsible for a large number of plants scattered all over the globe. Each plant typically comprises one to three turbines which are equipped with several hundred sensors each, mounted to the numerous components of the turbine. All sensors typically provide measurements at a rate between 0.1 and 1.0 Hertz, sending their data to the CCC. Engineers at the CCC have the task of monitoring the turbines, identifying faults, and taking reactive measures for preventing subsequent damage.

To illustrate how diagnostic knowledge in the turbine domain can be represented using $\mathcal{EL}^+$ description logic axioms, consider the expert statement *"fan blade vibrations and fluctuations in the combustion chamber's temperature indicate a can flame failure"*. Based on a diagnostic meta model that relates the unknown *FaultMode* that a certain *System operatesIn* to the observable *Symptom*s it *shows* (see Figure 2), the above statement can be formalized as follows:

$$\begin{aligned} &\mathit{System} \sqcap \exists\, \mathit{isInfluencedBy}\,.\,(\mathit{Fan} \sqcap \exists\, \mathit{shows}\,.\,\mathit{Vibrations}) \sqcap \\ &\exists\, \mathit{isInfluencedBy}\,.\,(\mathit{CombChamber} \sqcap \exists\, \mathit{shows}\,.\,\mathit{TempFluctuations}) \\ &\qquad \sqsubseteq \exists\, \mathit{operatesIn}\,.\,\mathit{CanFlameFailure} \end{aligned}$$

To understand how classification can be employed for diagnostics in this case, assume that the compositional model of the turbine also contains the axioms $\mathit{Turbine} \sqsubseteq \exists\, \mathit{hasComponent}\,.\,\mathit{Fan}$ and $\mathit{Turbine} \sqsubseteq \exists\, \mathit{hasComponent}\,.\,\mathit{CombChamber}$ describing the turbine components, and the role inclusion axiom $\mathit{hasComponent} \sqsubseteq \mathit{isInfluencedBy}$ stating that subcomponents influence their supercomponents. This static knowledge about a turbine is complemented by additional axioms based on sensor measurements: If the vibration sensor mounted at the fan hub detects vibrations exceeding a threshold, a corresponding axiom $\mathit{Fan} \sqsubseteq \exists\, \mathit{shows}\,.\,\mathit{Vibrations}$ is added. Analogously, temperature fluctuations at the combustion chamber lead to the addition of the axiom $\mathit{CombChamber} \sqsubseteq \exists\, \mathit{shows}\,.\,\mathit{TempFluctuations}$. Once this knowledge about currently observed symptoms has been added, the resulting ontology entails the axiom $\mathit{Turbine} \sqsubseteq \exists\, \mathit{operatesIn}\,.\,\mathit{CanFlameFailure}$ (among others).[12] By triggering the classification process either regularly based on a timer or after every addition of a symptom, a diagnostic system can therefore determine the state of the turbine based on a formal model of the system and its

[12] Since $\mathcal{EL}^+$ does not support Aboxes, we must encode facts as terminological axioms. A more natural modelling approach using Abox axioms could be realized in $\mathcal{EL}^{++}$.

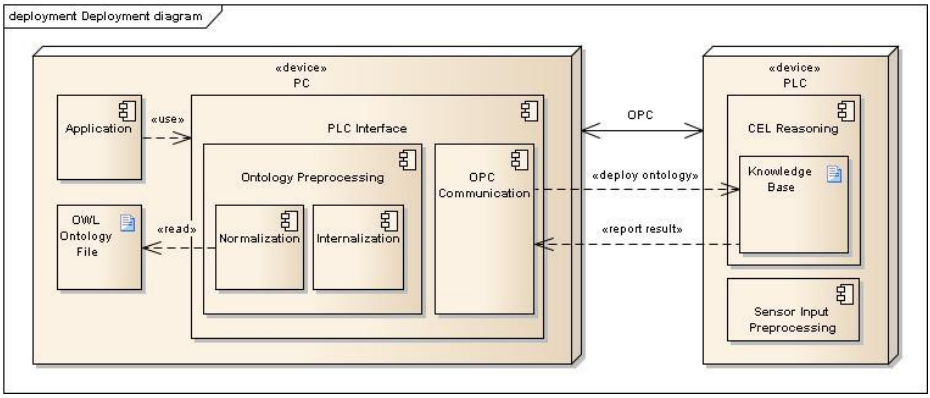


Fig. 3. An architecture for embedded reasoning on PLCs

diagnoses. If a faulty situation is detected, this can be signaled to the operator at the CCC, but also be used as input for other diagnostic components in a hierarchical setting.

Similar models are used in existing solutions such as [7,10], which typically assume that standard computer systems can be used for evaluating the models, e.g. by moving all data to a central store. To address the special restrictions posed by a PLC environment, we present a dedicated approach to embedded diagnostic reasoning, as follows.

3.2 An Architecture for Embedded $\mathcal{EL}^+$ Reasoning

Figure 3 shows an architecture for embedding $\mathcal{EL}^+$ reasoning in a PLC. Its core part is the *CEL Reasoning* component, running on a PLC, which implements consequence-driven $\mathcal{EL}^+$ reasoning based on the CEL approach from [1]. The *Knowledge Base* subcomponent contains both application-specific background knowledge as well as axioms reflecting current PLC sensor input.

The *CEL Reasoning* component is based on a very compact ontology representation suitable for resource-constrained embedded devices, such as a PLC, which are limited in computing power and memory. In order to be deployed on a PLC, any OWL ontology has to be converted to a compact ontology representation in a preprocessing step during development. For this purpose, a *PLC Interface* component running on a PC provides for ontology preprocessing as well as OPC-based communication between the PC and the PLC. The *Ontology Preprocessing* subcomponent reads a use case specific OWL ontology from a file. Next, its *Normalization* subcomponent normalizes ontology axioms to syntactically match the axiom types **(NFi)** used in the derivation rules of the CEL algorithm. In a subsequent ontology pre-processing step, the *Internalization* subcomponent maps ontology literals (concept names, role names) to corresponding integer numbers, which are used as indices in an array-based, compact

ontology representation on the PLC. Via OPC communication, the normalized and internalized ontology is finally deployed on the PLC.

During runtime, an application running on a PC, or alternatively a use case specific user program running on another superordinate PLC, may trigger the CEL Reasoning component on the PLC by sending an appropriate signal across the OPC communication link. Before the reasoning process, a *Sensor Input Processing* component of the PLC reads input signals from sensors, maps them to respective ontology axioms and adds them to the knowledge base. Accordingly, axioms that are no longer valid due to changes in sensor signals are being removed. When the reasoning process has been finished (indicated by an appropriate flag), the application or user program is reported relevant parts of the reasoning result, depending on the use case to be realized.

4 Consequence-Driven $\mathcal{EL}^+$ Reasoning on a PLC

In this section, we describe aspects of the implementation of consequence-driven $\mathcal{EL}^+$ reasoning in PLC environments, where restrictions preclude a direct implementation of the CEL algorithm as described in [1,3].

4.1 Difficulties with Reasoning Algorithms in PLC Environments

PLCs have very limited computing power and working memory capacity compared to contemporary desktop PCs. This, and their architectural design tailored to automation tasks, imposes various difficulties on the task of porting algorithms for efficient embedded OWL reasoning to a PLC environment.

One difficulty is the lack of dynamic data structures and memory allocation mechanisms. DL reasoning methods like tableaux or consequence-driven saturation procedures dynamically expand their data structures during the construction of tree-like models or derivation structures and partly also use backtracking to discard previously computed intermediate results. The CEL algorithm, in particular, operates on the mappings S and R, which are dynamically expanded by derived axioms during ontology classification. PLC platforms, however, only support static memory management based on a fixed size memory block scheme. Therefore, the consumption of PLC memory needs to follow a strategy of efficient memory layout based on upper bounds for the number of axioms that can potentially be derived in S and R, such that the very limited overall memory is not quickly exceeded by allocating sparsely populated blocks.

Another difficulty is rooted in the cyclic processing paradigm of PLCs. The repeated processing of a memory image of input signals within a fixed time slice is contrary to algorithms that expand their results non-deterministically within processing times that depend on intermediate results. The CEL algorithm, in particular, has two sources of dynamic expansion of results. Firstly, the application of the derivation rules **(CR*i*)** is triggered by both original axioms in the ontology and derived consequences in S and R, while the sequence of their application has an influence on the overall number of derivation steps required

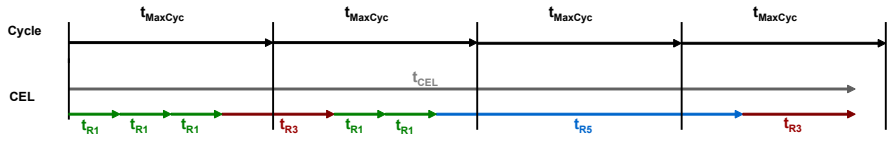


Fig. 4. Time cycle problems with PLCs

for classification. Hence, the number of rule applications is highly dynamic and cannot be determined in advance. Secondly, the time required for performing the various rules is also dependent on the previously computed consequences in S and R and differs from rule to rule. A rule can either be performed very quickly compared to the maximum cycle time of the PLC, or its runtime can even exceed this limit, in which case its interruption would lead to a PLC error state. Figure 4 depicts a cyclic view of the CEL algorithm, comparing the PLC's cyclic processing pattern (t_{MaxCyc}) to the overall classification time (t_{CEL}) composed of the runtimes for individual rule applications (t_{Ri}). The overall runtime t_{CEL} needs to fit into the cyclic time pattern given by t_{MaxCyc}, such that no application of a rule **(CR*i*)** with duration t_{Ri} exceeds the maximum cycle length. However, an intricate rule like **(CR*5*)** might not fit in a single cycle.

In the following sections, we describe our approaches to overcome these difficulties.

4.2 Compact Axiom Representation

The restricted memory available on PLCs requires a compact representation of axioms in the ontology $\|\mathcal{O}\|$ based on an integer encoding of concept and role names, which can then be used for efficient index-based memory array access. We achieve this by exploiting the fact that the normalized axioms in $\|\mathcal{O}\|$ are of one of the forms **(NFi)**, all of which require at most three class/role names as their parameters. Thus, we can encode any normalized axiom $\alpha \in \|\mathcal{O}\|$ as a four-tuple

$$\alpha = (\mathsf{T}, p_1, p_2, p_3),$$

where T encodes the type of normal form axiom (1-6) and the p_i are the integer numbers for concept and role names obtained by simply enumerating the elements in $\mathsf{N}_C^{\mathcal{O}}$ and $\mathsf{N}_r^{\mathcal{O}}$, e.g. in a lexicographic ordering with regard to their string name representation. Although the different positions of such an axiom tuple could be encoded with variable bit lengths depending on the number of concept and role names, we have chosen a representation that uses a two byte integer value per position for a total of eight bytes for an axiom, since such a fixed size encoding scheme provides for more efficient access. Based on this axiom encoding, we represent the ontology $\|\mathcal{O}\|$ on the PLC side as a fixed length array whose size of $8 \cdot \#\|\mathcal{O}\|$ bytes is determined during the preprocessing phase.

We represent the classification results stored in the mappings S and R as fixed length bit arrays $S[i][j]$ and $R[i][j][k]$, while i, j range over concept name and k

over role name index numbers. We have to assume that, in the worst case, all possible axioms of the forms $A \sqsubseteq B$ and $A \sqsubseteq \exists r.B$ are being derived, which provides the following upper bounds for the lengths of the arrays.

$$l_S = (\#\mathsf{N}_C^{\mathcal{O}})^2\,, \qquad l_R = (\#\mathsf{N}_C^{\mathcal{O}})^2 \cdot \#\mathsf{N}_r^{\mathcal{O}}$$

Thus, the overall memory required for representing the reasoning results in S and R is given by $\lceil (l_S + l_R)/8 \rceil$ in bytes. Efficient access to these arrays through the indices i, j, k also provides good performance for the frequent checks on S and R in the rules **(CRi)**.

While PC-based reasoners can optionally use such compact memory representations for optimization, they are necessary for porting reasoning algorithms to a PLC to handle the above mentioned memory issues.

4.3 Interruption-Safe $\mathcal{EL}^+$ Saturation

To address the difficulty about the PLC's cyclic processing paradigm, our objective is to safely fit the whole required computation time t_{CEL} for classification into the periodic cycles without running into error states by exceeding t_{MaxCyc} in a single rule application. To this end, we have implemented a time-out mechanism that uses a PLC's integrated interruption features for program-triggered termination of the current processing cycle. Figure 5 shows a timeline for the behavior of this interruption mechanism.

Shortly before the end of the cycle time t_{MaxCyc} is reached while processing a derivation rule, our mechanism initiates an artificial interrupt that saves the current state of rule processing to be reentered at the beginning of the next cycle. The state consists of the index variables i, j, k that serve to iterate over the structures S and R for checking a rule's applicability, as well as the currently processed axiom $\alpha \in \|\mathcal{O}\|$ and the type of rule **(CR*i*)** whose processing is to be reentered. In this way, we abstract from the cyclic processing constraints of a PLC, spreading the required overall calculation time t_{CEL} over an arbitrary sequence of cycles. In Section 5, we will show that the overhead for saving and reentering rule processing states is negligible for reasonable values of t_{MaxCyc}.

Notice that this interruption mechanism allows us to dynamically add diagnostic $\mathcal{EL}^+$-reasoning to any control program installed on a PLC whenever the

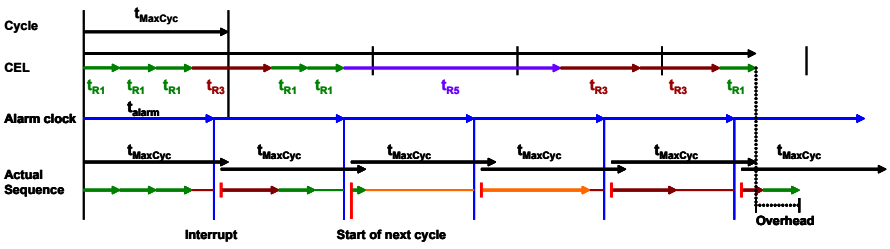


Fig. 5. Cycle time interruption mechanism

maximum cycle length determined by the control task is not fully exploited. The rest of t_{MaxCyc} not used for control tasks can then be utilized for diagnostic reasoning for many pre-installed PLCs in the automation field.

5 Evaluation

In this section, we report on first experimental results that we carried out to evaluate our embedded reasoning approach. For this purpose, we first describe our experimental setting, before we report on performance experiments.

5.1 Evaluation Setting

The hardware setting used for our evaluation consists of a SIMATIC IPC427C industrial PC with various sensors and a WinCC HMI interface attached to it via PROFINET. It is equipped with an Intel Core2 Duo U9300 1.2GHz processor and 956 MB RAM and runs Microsoft Windows XP SP2 as its operating system. During the SIMATIC WinAC RTX installation, the Ardence RTX 8.1 real time kernel is installed which adds real time capabilities to the OS. On this basis, the Software PLC SIMATIC WinLC (Windows Logic Controller) RTX v4.4.1 SP1 performs the tasks of a S7-300 or S7-400 PLC in our setting. The WinLC software provides the full functionality of an S7-300 or S7-400 PLC, although the processing time depends on the actual CPU used. It hosts our prototypical PLC-based $\mathcal{EL}^+$-reasoner implemented in SCL following the architecture and implementation features described in the sections 3 and 4 . For OPC communication we use the Siemens OPC Server v7.0 and a Java-based PC-client.

As test data we have used a set of $\mathcal{EL}^+$ ontologies that stem from the turbine diagnostics use case described in Section 3. They are listed in Table 1 with their number of concepts, roles and axioms. The base ontology $\mathcal{O}_{tur}$ is a local diagnostic model that was used to capture the causal relationships between symptoms and faults for certain parts of turbine machinery in the $\mathcal{EL}^+$ formalism. Since in $\mathcal{O}_{tur}$ not all the $\mathcal{EL}^+$ constructs are used, however, we have produced a modified version $\mathcal{O}^+_{tur}$, which contains some additional axioms to cover the full expressivity of $\mathcal{EL}^+$ (the missing construct was conjunction $\sqcap$). As a result, classification of $\mathcal{O}^+_{tur}$ triggers all the derivation rules **(CRi)** at least once and thus comprises a suitable test data ontology for our system. To scale to larger test data, we extended the ontologies $\mathcal{O}_{tur}$ and $\mathcal{O}^+_{tur}$ to multiplied versions that contain multiple renamed copies of the original axioms, indicated by a factor in their name (e.g., $\mathcal{O}_{tur10}$ contains 10 copies of the original axiom set in $\mathcal{O}_{tur}$). The largest ontologies $\mathcal{O}_{tur22}$ and $\mathcal{O}^+_{tur16}$ have been chosen such that their classification uses up all of the PLC's total memory of 4MB. In this way we get runtime results for the case of maximal memory usage as an upper bound for answer times.

5.2 Performance Experiments

Correctness Tests. To ensure correctness of our implementation, we performed a back-to-back test of our PLC-based $\mathcal{EL}^+$ reasoner compared to a standard

Table 1. Detailed overview of ontologies, their memory consumption and runtimes

	Ontology				memory in kByte				runtime in ms	
	$\mathcal{O}$	$\#\mathsf{N}_C^{\mathcal{O}}$	$\#\mathsf{N}_r^{\mathcal{O}}$	$\#\|\mathcal{O}\|$	$\|\mathcal{O}\|$	S	R	total %	t_{CEL}	t_{total}
1	$\mathcal{O}_{tur}$	28	4	49	0.38	0.10	0.38	0.02%	29	250
2	$\mathcal{O}_{tur5}$	136	20	245	1.91	2.26	45.16	1.18%	773	641
3	$\mathcal{O}_{tur10}$	271	40	490	3.83	8.97	358.60	9.02%	1774	2156
4	$\mathcal{O}_{tur22}$	595	88	1078	8.42	43.22	3803.00	94.00%	8848	9141
5	$\mathcal{O}_{tur}^{+}$	34	7	77	0.60	0.14	0.99	0.03%	113	375
6	$\mathcal{O}_{tur3}^{+}$	100	21	231	1.80	1.22	25.63	0.68%	976	1188
7	$\mathcal{O}_{tur5}^{+}$	166	35	385	3.00	3.36	117.73	2.99%	2815	2937
8	$\mathcal{O}_{tur16}^{+}$	529	112	1232	9.63	34.16	3825.95	94.36%	29586	29781

reasoner available on a PC platform. For this purpose, we used the Java-based JCEL[13] reasoner as a reference system in order to compare the classification hierarchy output by our system through the structure S to that produced by JCEL. We noticed no differences in the output of the two systems for our turbine diagnostics ontologies or for some $\mathcal{EL}^+$ ontologies available on the web. Although this is not a 100% test, we argue that these tests strongly support the correctness of our implementation, especially since the ontology $\mathcal{O}_{tur}^{+}$ is modified such that it covers all features of $\mathcal{EL}^+$ and triggers all the derivation rules (**CR1**) - (**CR6**), which ensures a certain coverage of the underlying formalism's constructs.

Runtime Performance and Scalability Test. Runtime and scalability performance requirements for embedded reasoning very much depend on the particular use case in question. For the diagnostics use case that we consider here, neither the ontologies get typically very large as any single PLC does only have to reason over the local diagnostic model for its surrounding machinery, nor the answer times for retrieving diagnoses need to be in real time due to the offline nature of the diagnostic task. Therefore, we varied the ontology size in our experiments in the lower ranges (compared to typical benchmark ontologies) and we accepted answer times for retrieving diagnoses within seconds or even minutes.

Notice that a direct comparison of classification times with those of PC-based systems, such as JCEL, does not provide useful insights on performance issues for PLC-based reasoning, since today's standard PCs are much more powerful than PLCs in terms of CPU and memory. Instead, we are interested in the memory consumption and runtime behavior of our PLC-based reasoning approach in the light of the requirements of our diagnostics use case. In particular, we are interested in bringing embedded reasoning onto existing PLC hardware pre-installed in the automation field alongside the control tasks that these PLCs already run. To this end, we report on investigations about memory consumption, optimal cycle length and answer times for diagnoses in the following.

Memory Consumption. The memory required for classifying an ontology $\mathcal{O}$ in the PLC is given by the size and signature of $\mathcal{O}$, while $\mathsf{N}_C^{\mathcal{O}}$ and $\mathsf{N}_r^{\mathcal{O}}$ determine

[13] http://jcel.sourceforge.net/

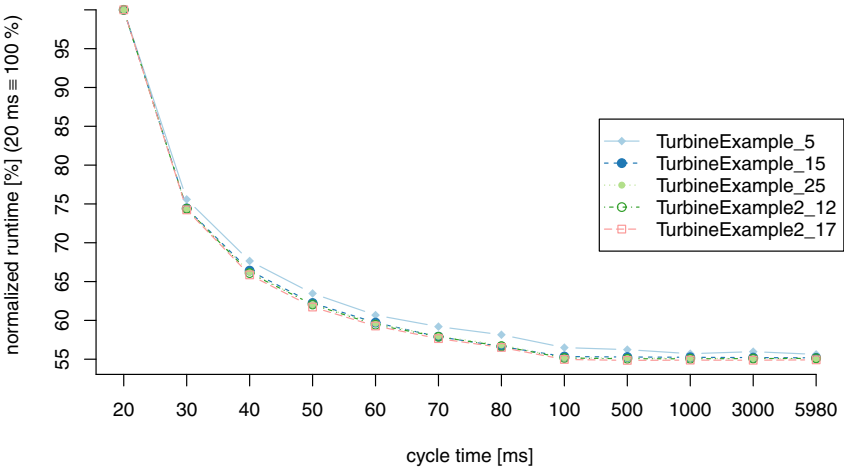


Fig. 6. Cycle Time Evaluation

the size of the data arrays for the mappings S and R. Table 1 shows the sizes of these memory components for all the test ontologies used. It also shows the percentage of the PLC's total memory used for classification. It can be seen that most of the memory used is reserved for the classification results stored in S and R, while the representation of the ontologies' axioms occupy only a small amount, for all but the smallest ontologies. Classification of the single original and modified diagnostic models $\mathcal{O}_{tur}$ and $\mathcal{O}^+_{tur}$ requires only modest memory, while for their multiplied versions a polynomial increase can be observed due to the respective growth of the arrays for S and R. Only for the largest artificially increased models of over a thousand axioms does memory consumption reach critical values. For the expected sizes of local diagnostic models, it should thus be feasible to run diagnostic reasoning on a pre-installed PLC in addition to the automation tasks it already runs, without consuming too much of its memory.

Cycle Time Optimization. User programs performed on pre-installed PLCs often do not require the possible maximum cycle time allowed by the process to be controlled but run much faster. By exploiting such potential process control idle times, additional tasks like diagnostic reasoning can be run within a PLC's execution cycle at no cost of critical CPU power if the cycle length for reasoning is chosen small enough to fit the idle time.

To this end, we investigated the overhead that comes with small execution cycle times for $\mathcal{EL}^+$ reasoning in our approach, where any interruption of rule execution requires the saving and reentering of the current execution state. The diagram in Figure 6 shows the relative classification times in relation to different cycle lengths for all the test ontologies. (The longest classification time corresponds to 100%.) It can be seen that the overhead for rule interruption is only significant for very small cycle lengths less than about 100ms. Beyond

this threshold the overhead becomes negligible for all the test ontologies considered. This suggests that from a cycle time of 100ms or greater, PLC-based $\mathcal{EL}^+$-reasoning runs in an optimal mode without wasting a significant amount of execution time due to interruption handling. Since this is a rather low number, we are hopeful to encounter many cases for adding $\mathcal{EL}^+$-based diagnostic reasoning to pre-installed PLCs, exploiting potential process control idle times.

Runtime Performance. As for runtime measurement, Table 1 shows the time for actual classification on the PLC (t_{CEL}) and total time (t_{total}) including OPC communication between the PLC and PC (ontology deployment + result reporting). Classification times for the original diagnostic model $\mathcal{O}_{tur}$ as well as for the modified $\mathcal{O}^+_{tur}$ are within milliseconds and are thus rather fast. Also the multiplied models of up to five or even ten copies are being classified within a few seconds. For the larger samples $\mathcal{O}_{tur22}$ and $\mathcal{O}^+_{tur16}$, classification requires several seconds up to half a minute. Since these two ontologies already fill the maximum memory of the PLC, we can say that the answer times in our approach lie within feasible bounds for the task of offline diagnostics. The OPC-based communication causes a significant relative overhead for small ontologies and becomes negligible for larger models.

6 Conclusion

In this paper, we have presented an approach to porting OWL 2 EL reasoning to PLCs, a type of embedded controller prevalent in industrial automation. We have described how deficiencies about the differing paradigms of PLCs and PC-hosted reasoning algorithms can be overcome by a compact memory representation strategy and an interruption-safe variant of the CEL [1] classification procedure suitable for PLCs. Furthermore, we have reported on promising initial experimental results carried out on a prototypical implementation of a PLC-based $\mathcal{EL}^+$ reasoner in the context of a use case about diagnostics for turbine machinery, in which we prove the feasibility of our approach in terms of memory consumption and answer times for retrieving diagnoses.

A particular finding we have made here is that attempts of porting reasoning procedures to strongly restricted embedded devices could greatly benefit from avoiding too much dynamics and flexibility in the data structures used, ideally keeping them as static as possible. In this work, we could successfully employ low-degree polynomial upper bounds on the maximum size of derivation results that reside in memory at a single time. For tableau-style algorithms, whose tree-like models appear to be more difficult to handle, the separate calculation of single tableau branches at once goes in this direction.

To the best of our knowledge, this is the first endeavor of bringing description logic reasoning to the PLC-dominated industrial automation field. Building on this enabling step of providing a platform for embedded reasoning, we intend to utilize this platform in our use case of turbine diagnostics in forthcoming field tests and experiments to have an impact on the maintenance of power plants in the energy sector. Next to machine diagnostics, we see many other potential

use cases for embedded reasoning in industry, such as component verification in industrial engineering or the support of condition monitoring and control tasks, that benefit from exploiting an explicit representation of expert knowledge models brought close to industrial machinery.

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: Proc. of IJCAI 2005, pp. 364–369. Professional Book Center (2005)
2. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook. Cambridge University Press (2003)
3. Baader, F., Lutz, C., Suntisrivaraporn, B.: Efficient reasoning in EL+. In: Proc. of DL 2006. CEUR Workshop Proceedings, vol. 189. CEUR-WS.org (2006)
4. Berger, H.: Automating with STEP 7 in STL and SCL: programmable controllers SIMATIC S7-300/400. John Wiley & Sons (2007)
5. Berger, H.: Automating with SIMATIC: Controllers, Software, Programming, Data Communication Operator Control and Process Monitoring. Publicis (2009)
6. Hubauer, T.M., Grimm, S., Lamparter, S., Roshchin, M.: A diagnostics framework based on abductive description logic reasoning. In: Proc. of ICIT 2012, March 19-21. IEEE Computer Society (2012)
7. Hubauer, T.M., Legat, C., Seitz, C.: Empowering Adaptive Manufacturing with Interactive Diagnostics: A Multi-agent Approach. In: Demazeau, Y., et al. (eds.) PAAMS 2011. AISC, vol. 88, pp. 47–56. Springer, Heidelberg (2011)
8. Jang, M., Sohn, J.-C.: Bossam: An Extended Rule Engine for OWL Inferencing. In: Antoniou, G., Boley, H. (eds.) RuleML 2004. LNCS, vol. 3323, pp. 128–138. Springer, Heidelberg (2004)
9. Kazakov, Y., Krötzsch, M., Simančík, F.: Concurrent Classification of $\mathcal{EL}$ Ontologies. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 305–320. Springer, Heidelberg (2011)
10. Legat, C., Hubauer, T.M., Seitz, C.: Integrated diagnosis for adaptive service-oriented manufacturing control with autonomous products. In: Proc. of IESM 2011, pp. 1363–1372. International Institute for Innovation, Industrial Engineering and Entrepreneurship (I^4e^2) (2011)
11. Seitz, C., Schönfelder, R.: Rule-Based OWL Reasoning for Specific Embedded Devices. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 237–252. Springer, Heidelberg (2011)
12. Sinner, A., Kleemann, T.: KRHyper - In Your Pocket. In: Nieuwenhuis, R. (ed.) CADE 2005. LNCS (LNAI), vol. 3632, pp. 452–457. Springer, Heidelberg (2005)
13. W3C, O.W.L.: Working Group. OWL 2 Web Ontology Language: Profiles. W3C Recommendation, October 27 (2009)

Experiences with Modeling Composite Phenotypes in the SKELETOME Project

Tudor Groza[1], Andreas Zankl[2], and Jane Hunter[1]

[1] School of ITEE, The University of Queensland, Australia
tudor.groza@uq.edu.au, jane@itee.uq.edu.au
[2] Bone Dysplasia Research Group
UQ Centre for Clinical Research (UQCCR)
The University of Queensland, Australia
a.zankl@uq.edu.au

Abstract. Semantic annotation of patient data in the skeletal dysplasia domain (e.g., clinical summaries) is a challenging process due to the structural and lexical differences existing between the terms used to describe radiographic findings. In this paper we propose an ontology aimed at representing the intrinsic structure of such radiographic findings in a standard manner, in order to bridge the different lexical variations of the actual terms. Furthermore, we describe and evaluate an algorithm capable of mapping concepts of this ontology to exact or broader terms in the main phenotype ontology used in the bone dysplasia domain.

1 Introduction

Skeletal dysplasias represent a group of rare genetic disorders affecting the skeletal development. Patients with such disorders suffer from complex medical issues that can be grouped into three categories: (i) clinical findings, i.e., pains in limbs; (ii) radiographic findings, i.e., bilateral arachnodactyly; (iii) genetic findings, i.e., deletion mutation in FGFR3. In a previous paper [1], we have introduced the SKELETOME project that has developed a community-driven knowledge curation platform for this domain, able to capture and integrate such clinical, radiographic and genetic findings. The underlying foundation of the platform is an ontology-driven knowledge engineering cycle introduced to bridge the current knowledge about the domain and the continuously growing pool of patient cases. The cycle has two phases: (1) semantic annotation – bridging knowledge to cases – and (2) collaborative diagnosis, collaborative knowledge curation and evolution – from cases to knowledge. The semantic annotation process relies on clinical and radiographic findings grounded in Human Phenotype Ontology (HPO) [2] concepts – one of the only phenotype ontologies for rare disorders.

In this paper we focus on issues associated with this semantic annotation process, and more precisely on representing, in a standard manner, radiographic findings present in X-Ray descriptions and clinical summaries. At the same time, since the phenotype knowledge in SKELETOME is modeled only via HPO concepts, we aim to map where possible, instances of this standard representation to

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 82–97, 2012.

terms existing in HPO. The root of our problem lies in the structural and lexical differences that exist between the terms describing such radiographic findings.

There currently are two major "*vocabularies*" [1] used within the community: (i) a generic one, and (ii) the International Skeletal Dysplasia Registry (ISDR) vocabulary. The generic vocabulary, used by the vast majority of clinicians, consists of an unstructured and virtually unlimited set of terms representing associations between qualities and anatomical entities – entries in free text clinical summaries. The terms used to describe a patient case are subject to the personal style of the clinician documenting the case, and hence may take different granularities and lexical groundings. For example, a clinician may use the term *bowed tibial shaft*, while another may use the term *angulation of the tibial diaphysis* to denote, in practice, the same thing. On the opposite side, the ISDR vocabulary, used at a much smaller scale only by ISDR, has a fixed and hierarchical set of around 270 terms representing anatomical parts, each having associated, in average, 5 to 10 qualities (hence a total of around 2,000 terms). Patient case findings will always have assigned terms from this set and each term will have the exact same structure and lexical grounding in all cases. For example, the corresponding entry for the two terms mentioned above would be: *Tibia – Diaphysis – Abnormality: Angulated*. The differences between the different lexical groundings of terms make the semantic annotation process very challenging.

In addition to the issues listed above, while HPO is, to date, the most comprehensive phenotype ontology for rare disorders, unfortunately, it is far from being complete. As a result, in order to provide a proper context for radiographic findings found in patient cases, we do not only require a mapping to existing HPO terms – where these exist, but also a mapping to the most appropriate parent within HPO – for those that don't have an exact match.

The contribution brought by this paper is two-fold: (i) we describe an ontology, the Phenotype Fragment Ontology (PFO), aimed at providing a standard structure for radiographic findings, independently of the actual lexical groundings, and (ii) we propose an algorithm that maps concepts modeled with this ontology to HPO terms by considering both exact and broader matches.

The goal of PFO is to capture the inner structure of radiographic findings by enabling the construction of complex phenotypes via combinations of anatomical entities (i.e., *Diaphysis* – partOf – *Tibia*) and qualities (i.e., *Bowed*). In practice, PFO provides a meta-model for phenotypes where the actual concepts (i.e., anatomical entities and qualities) are defined via well-known and widely adopted ontologies in the biomedical domain, such as the Foundational Model of Anatomy (FMA) [3] and the Phenotype and Trait Ontology (PATO) [4]. The granularity proposed by PFO does not only provide a solution to the issues discussed in this paper, but also enables a fine-grained exploration of the phenotype space in the bone dysplasia domain. This, in turn, enables the exploration of commonalities

[1] Throughout the paper, we use the term *vocabulary* to denote the structural and lexical commonalities that group a set of terms used to describe radiographic findings. From a semiotic and medical perspective the community uses a single set of terms.

between disorders based on the anatomical localization of phenotypes and the development of anatomical localization - oriented decision support methods.

Starting from concepts represented using PFO, we have developed a mechanism that maps them to exact or broader HPO terms. Since this mapping is part of the semantic annotation process in SKELETOME, the user plays a central role by validating the mapping results. Our algorithm provides a ranked list of candidate HPO terms, of which the top 5 are being presented to the user. Consequently, our focus has been on achieving a high precision.

Ontology matching (OM) has been a very active research area during the last decade. Systems like Falcon [5], RiMOM [6], SAMBO [7] or DSSim [8] have achieved impressive results during several OM challenges (see [9] for a comprehensive overview on OM). Apart from the lexical similarity performed by these systems (which is dependent on the lexical groundings), the overall mapping process we require is different. Hence, we were unable to directly use or compare against any of them. Ontology matching assumes the mapping of concepts from one ontology to corresponding concepts in another ontology. In our case, PFO does not provide actual concepts that could be directly mapped. It only provides the scaffolding onto which concepts can be created, while the actual semantics is provided by terms from FMA and PATO, used to compose PFO concepts [2].

The remainder of the paper is organized as follows. Section 2 provides a brief overview of HPO and discusses the motivation behind PFO. Section 3 describes PFO and its associated engineering process. In Section 4 we detail the mapping algorithm, and before concluding in Section 6, we discuss some experimental results and the shortfalls of our approach in Section 5.

2 Background and Motivation

2.1 Human Phenotype Ontology

HPO has been developed to provide a controlled vocabulary for phenotypic features encountered, in principle, in hereditary diseases listed in the Online Mendelian Inheritance in Man (OMIM) database [3]. The ontology, currently comprising around 9,900 concepts, describes three main streams [4]: (i) Mode of Inheritance, (ii) Onset and clinical course, and (iii) Phenotypic abnormalities.

Phenotypic abnormalities (the concepts of interest in our study) represent more than 95% of the ontology and are organized in a hierarchical manner (via class–subclass relations). This hierarchy is, in principle, based on the main anatomical systems, such as, the nervous system (`HP_0000707` – *Abnormality of the nervous system*) or the skeletal system (`HP_0000924` – *Abnormality of the*

[2] Mapping HPO terms to concepts from other phenotype ontologies has been previously discussed in the literature. However this also falls under Ontology Matching since the goal is to match `Cardiomegaly` from HPO, for example, to `Enlarged heart` from the Mammalian Phenotype Ontology.

[3] `http://www.omim.org/`

[4] Please note that all experiments discussed in this paper have been conducted on the HPO version from 31 May 2012.

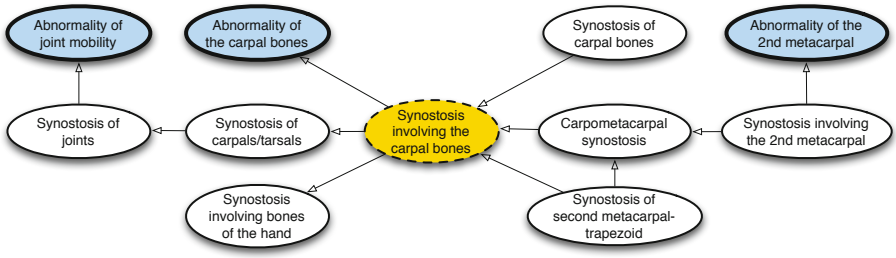


Fig. 1. A snapshot of the HPO structure (arrows denote class–subclass relations)

skeletal system). Each concept has a label and may have a definition and exact or related synonyms.

There are two aspects that raise challenges when using HPO: multiple inheritance and an overall inconsistency of the lexical representations of the concepts. The latter refers to the inconsistency in the inner lexical structure of the terms (i.e., *Synostosis of joints* vs. *Carpometacarpal synostosis*), as well as to the reuse of the same conceptual lexical grounding in multiple terms (e.g., `HP_0005048` – *Synostosis of carpal bones* has listed *Fusion of carpal bones* as synonym, while `HP_0009702` – *Synostosis involving the carpal bones* has listed as synonyms *Fused carpal bones*). Finally, a third issue is the use of generic lexical representations as synonyms for very specific concepts (e.g., `HP_0010239` – *Aplasia of the middle phalanges of the hand* has listed as synonym *Absent middle phalanges*). All these issues make the mapping process more complex and affect its resulting precision.

The multiple inheritance, on the other hand, requires one to adopt a goal-oriented interpretation of the hierarchy, if relying on the subclass relations between terms. Fig. 1 depicts an example of multiple inheritance extracted from HPO. This shows how at different levels in the hierarchy one may find concepts that are subclasses of both concepts defined based on the anatomical localization of the abnormality (i.e., *Abnormality of joint mobility*), as well as concepts defined based on the type of the abnormality (i.e., *Synostosis of carpals/tarsals*). In addition, the structure also contains relations that do not follow any of the two directions, e.g., the subclass relation between *Synostosis of second metacarpal-trapezoid* and *Synostosis involving the carpal bones*. In our case, these aspects have influenced the design of the mapping algorithm described in Section 4. More concretely, when defining broader matches between PFO and HPO concepts we have considered a correct match to be the most specific HPO ancestor that is defined based on the anatomical localization. For example, in this interpretation, *Synostosis involving the 2nd metacarpal* would have the closest broader match *Abnormality of the 2nd metacarpal*.

2.2 Analysis of Radiographic Findings

In order to gain a deeper understanding in the inner structure of the radiographic findings we have collected a list of 675 random findings from patient cases listed

Table 1. Coverage of our radiographic findings in HPO

Category	Total terms	Exact HPO match	Broader HPO match	No HPO match
Simple	387	237 (61%)	136 (35%)	14 (4%)
Composite anatomy	156	66 (42%)	56 (36%)	34 (22%)

in the European Skeletal Dysplasia Network registry and from a widely adopted text book in the bone dysplasia domain – Spranger et al., *Bone dysplasia: an atlas of genetic disorders of skeletal development* [5]. All items in this list can be categorised under the generic vocabulary introduced in Section 1. The nature of the ISDR vocabulary provided us with access to all possible terms in it, hence there was no need for a collection process. Also, the analysis focuses only on the generic vocabulary since this is the one to introduce the major issues previously described. The size of the list represents around a fifth of the total number of findings present under the HPO *Abnormality of the skeletal system* (3,744 sub-concepts), which is of particular interest for the skeletal dysplasia domain.

The analysis of the list of radiographic findings has revealed that they can be grouped, based on their inner structure, into three categories: (i) simple findings, i.e., associations between a single anatomical entity and qualities – *flat skull* (387 findings); (ii) composite anatomy findings, i.e., associations of composed anatomic entities and qualities – *bifid distal phalanx of the thumb* (156 findings); and (iii) composite phenotypes, i.e., conjunctions of findings from the previous two categories – *curved femora with rounded distal epiphyses* (132 findings).

As a next step, we investigated to what extent is HPO able to cover these findings, with a focus only on the first two categories as findings in the last category are covered by combining existing findings in the first two. Consequently, we have manually mapped the findings in the list to HPO terms. Table 1 summarises the mapping results. In the "simple" category 237 findings (61%) had an exact match, 136 findings (35%) could be associated with a broader match (considering the interpretation for broader matches given in the previous section), while the rest of 14 (4%) could not be mapped. Two reasons made the mapping impossible for this last set: the findings represented *normal* states (i.e., *normal pelvis*), which have no correspondence in HPO (since it models only abnormal findings) or the findings were too generic and their interpretation was subject to a particular context (i.e., *gracile bones* – in the context of a particular X-Ray, *bones* could refer to, for example, *phalanges*). Similarly, in the "composite anatomy" category 66 findings (42%) had an exact match, 56 (36%) a broader match, and 34 (22%) we were unable to map.

The exact match manual mappings have then been used as ground truth in an attempt to automatically map the findings to HPO terms. In this experiment, we've used the NCBO Annotator [10] and three well-known string similarity measures: Levenstein, Needleman-Wunch and Smith-Waterman (the last two are

[5] The list can be downloaded from: `http://tiny.cc/grtifw`

Table 2. String similarity measures performance on the radiographic findings

Similarity	P@1	P@2	P@3	P@4	P@5
Category: Simple					
Levenstein	**0.55**	0.29	0.21	0.16	0.13
N-W	0.5	0.27	0.19	0.14	0.11
S-W	0.54	0.29	0.2	0.15	0.13
Category: Composite anatomy					
Levenstein	**0.41**	0.25	0.17	0.13	0.11
N-W	0.38	0.21	0.14	0.11	0.09
S-W	0.36	0.24	0.17	0.14	0.12

Table 3. NCBO Annotator annotation performance on the radiographic findings

Category	Precision	Recall	F1 score
Simple	0.79	0.37	0.5
Composite anatomy	0.57	0.2	0.3

the reference algorithms used to align gene/protein sequences in Bioinformatics). In order to ensure a fair comparison, in the case of the similarity measures, the 1-to-1 mapping (finding – HPO term) has been realized by computing the similarity of the finding against the label and all synonyms of the HPO term. The highest similarity score was returned as the final similarity between the two. We've evaluated the resulting performances by looking at precision at k (P@k) with k=1, 2, 3, 4, 5 as presented in Table 2. In the case of the NCBO Annotator, since the annotation results are dichotomous (i.e., a match is either found or not) we have calculated the standard precision, recall and F1 (results are listed in Table 3).

The result of this experiment shows (if it was necessary) that the mapping process is very sensitive to the lexical representation of the findings. As an additional remark, the example provided in the introduction (i.e., *bowed tibial shaft* vs. *angulation of the tibial diaphysis*) is a typical case that can make the difference between a mapping hit and a miss. Overall, the NCBO Annotator had a very good precision at the expense of the recall, while among the similarity measures the best performance has been achieved by the Levenstein distance in both categories of findings. The two results are obviously not directly comparable, but they did provide us with a good overview of what we can achieve, with respect to our goal, with off-the-shelf solutions.

Consequently, we saw the need to create a standard format for these findings, abstracting from the actual lexical representation (i.e., capturing the *object* as opposed to the *symbol*, from a semiotical perspective), and for which we can design a generic mapping mechanism. This standard structure is provided by our Phenotype Fragment Ontology (PFO), described next. Another obvious solution

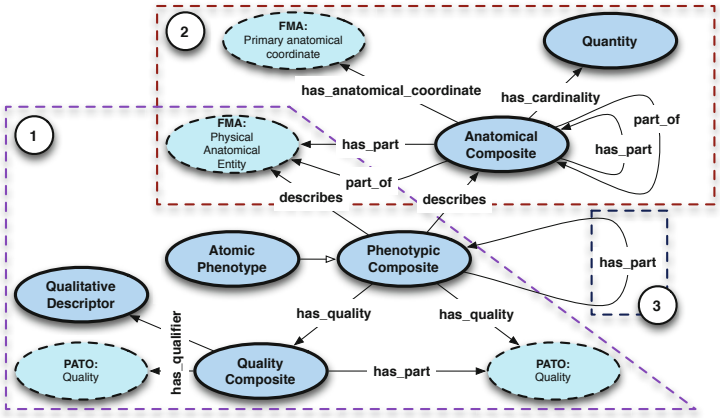


Fig. 2. Snapshot of the main part of the Phenotype Fragment Ontology

would have been to design a direct mapping algorithm between findings and HPO terms, sensitive to the actual terms yet, probably, achieving a fairly high accuracy. However, this would have served solely its design purpose, i.e., the direct mapping. PFO, on the other hand, opens several new research paths by enabling the exploration of radiographic findings at a level never achieved before.

3 The Phenotype Fragment Ontology

The Phenotype Fragment Ontology (PFO) has two main goals: (i) to provide a standard representation for radiographic findings, based on their intrinsic structure, and (ii) to enable the creation of the corresponding concepts by re-using concepts from widely adopted ontologies, i.e., FMA and PATO.

Fig. 2 depicts the core part of PFO [6] together with the design steps. The central concept of the ontology, the `Phenotypic Composite`, carries a bridging role between the anatomical side of the findings and the qualities they bear. Starting from this central concept, the design of PFO has followed closely the result of the experiment detailed in Section 2. As a first step (no. 1 in the figure) we have added support for modeling simple findings, as associations of `FMA: Physical Anatomical Entity` (via the *describes* relation) and qualities (via the *has_quality* relation), which can either be a `PATO: Quality` or a `Quality Composite`. The latter can then be further expressed as an association between an explicit quality and a qualifier. Most qualities and qualifiers are directly reused from PATO, however, where this is not possible, PFO introduces its own concepts. Secondly, we have introduced the `Anatomical Composite` concept to enable the modeling of the second category of findings. This, may have (*has_part*) or be a *part of* a `FMA: Physical Anatomical Entity` and may have attached

[6] http://purl.org/skeletome/phenotype

an anatomical coordinate (the figure only presents one such type of anatomical coordinate, but the ontology contains several) or a cardinality. Finally (no. 3 in the figure), in order to capture the third category (i.e., composite phenotypes per se), we have added the *has_part* self-relation on `Phenotypic Composite`.

Returning to the example introduced in Section 1, it can be observed that, independently of the lexical grounding (*bowed tibial shaft* or *angulation of the tibial diaphysis*), the concept denoting the abstract radiographic finding will have the following structure: a `Phenotype Composite` that

- *describes* an `Anatomical Composite`, which *has_part* `FMA: Diaphysis`, and is *part_of* `FMA: Tibia`, and
- *has_quality* `PATO: PATO_0000406` (*Bowed*).

At the same time, this structure maps perfectly onto the structure of the ISDR concepts. Similarly, *Flat skull* is modeled as: `Phenotype Composite` *describes* `FMA: Skull` and *has_quality* `PATO: PATO_0000407` (*flat*). For conciseness we have not included the logical definitions of these concepts. However, the definitions of all concepts discussed in the previous section, from the simple and composite anatomy findings categories, plus some examples from the composite phenotypes category, can be found at `http://purl.org/skeletome/spo`. These definitions have been also used for the evaluation described in Section 5.

While lightweight, PFO enables the definition of standard representations for very rich radiographic findings. In addition, it brings a series of side advantages, such as: (i) a **standard lexicon**, since the lexical grounding of the concepts will always be provided by the ontologies that underpin its definition; (ii) explicit modeling of **cardinality**, which is important from a clinical perspective; (iii) the possibility of modeling **normal** phenotypes, again important from a clinical and decisional perspective; and (iv) the scaffolding for decomposing **atomic elements**, i.e., monolithic terms that do not reveal in their lexical representation the association anatomical localization – quality; for example, *macrocephaly*, which can be represented as `FMA: Head` – `PATO: PATO_0000586` (*Large*)

On the negative side, relying on external ontologies introduces a series of issues, one of which is the lack of concepts to represent certain anatomical parts or qualities (the rest are discussed in Section 5). For example, FMA has no corresponding concept for *Müllerian duct*, while PATO does not cover most of the "metaphoric" qualities, such as *angel-shaped* or *cloverleaf* – which we had to introduce in PFO.

4 Mapping PFO Concepts to HPO Terms

As mentioned in Section 1 our second goal is to map radiographic findings modelled as PFO concepts to exact or broader terms in HPO. Consequently, we have developed a mapping algorithm that ranks all HPO terms according to their similarity to a given PFO concept (described in Section 4.1). This algorithm has been evaluated for exact matching in Section 5. For finding broader HPO terms, we've added a series of extra steps to the general algorithm, as detailed in Section 4.4.

4.1 General Mapping Algorithm

Algorithm 1 lists the general mapping algorithm, while snippets 2, 3 and 4 present some specific methods used in it. In order to get a better understanding of the algorithm we will consider as example, the 1-to-1 mapping of `HP_0009611` (*Notched terminal thumb phalanx*) to `Bifid_distal_phalanx_of_the_thumb` [7].

Lexicon Creation and Tokenization. A PFO concept has two main elements, i.e., the anatomical part and the quality, each of which may be associated with an extra set of elements, i.e., an anatomical coordinate and qualifiers – as described in Section 3. For each of these four elements, we generate the corresponding lexicons using the concepts that underpin their definition. Each lexicon comprises the label and all the synonyms of the given concept. Subsequently, all entries in the lexicons are tokenized. The same procedure is also employed on the HPO concept. In our example, the PFO concept has two anatomical entities (`FMA: Phalanx of finger` and `FMA: Thumb`), one anatomical coordinate (`Distal`) and one quality (`Bifid`). Hence, the result of this step is:

- $LexAnat_C\_TOKENS$ = { [Phalanx of finger ← (phalanx, of, finger), Hand phalanx ← (hand, phalanx), ...], [Thumb ← (thumb), First digit of hand ← (first, digit, of, hand)] }
- $LexCoord_C\_TOKENS$ = { [Distal ← (distal), Terminal ← (terminal)]}
- $LexQual_C\_TOKENS$ = { [Bifid ← (bifid), Forked ← (forked)]}

Similarly, the result for the HPO concept is:

- HPO_C_TOKENS = { [Notched terminal thumb phalanx ← (notched, terminal, thumb, phalanx), Bifid distal phalanx of thumb ← (bifid, distal, phalanx, of, thumb]), ...] }

Similarity Matrix and Traces Computation. For each entry in each lexicon of the PFO concept (i.e., Anat, Coord, Qual and Qualif) we compute a similarity matrix and associated traces against each entry in the lexicon of the HPO concept. As a remark, we use the term *trace* to denote the maximal diagonal in a similarity matrix and not the usual trace that can be computed only in squared matrices. Section 4.2 details this process and exemplifies it for *Phalanx of finger* vs. *Bifid distal phalanx of thumb*. For each entry in each lexicon of the PFO concept this step results in a list of associations (HPO lexicon entry – max trace) for both the full and the optimal length of the entry. As an example, *Phalanx of finger* will have the following (partial) result:

- Full length traces: {(notched terminal thumb phalanx – 0.33), Length: 3; (bifid distal phalanx of thumb – 0.66), Length: 3}
- Optimal length traces: {(notched terminal thumb phalanx – 0.59), Length: 1 (only *phalanx* is used); (bifid distal phalanx of thumb – 0.66), Length: 3}

[7] `http://purl.org/skeletome/spo#Bifid_distal_phalanx_of_the_thumb`

Algorithm 1. General mapping algorithm

Require: PFO_C

$PFO_C = \{Anat_C, Coord_C, Qual_C, Qualif_C\}$
$LEX_C = \{LexAnat_C, LexCoord_C, LexQual_C, LexQualif_C\}$,
where $LexX_C = \{label, syn_1, \ldots, syn_n\}$ of X_C and X ∈ {Anat, Coord, Qual, Qualif}

// Tokenization of all lexical groundings of each entry in a particular lexicon
for all $LexX_C \in LEX_C$ **do**
    $LexX_C\_TOKENS = \{LexX_C\_Tokens_i = [t_1, t_2, \ldots, t_N], i = [1, N]\}$
    where N = No. synonyms + 1 (the label) and $X \in \{Anat, Coord, Qual, Qualif\}$
end for

for all $HPO_C \in HPO$ **do**
    // Tokenization of all lexical groundings of the HPO concept
    $HPO_C\_TOKENS = \{HPO_C\_ENTRY_i = [t_1, t_2, \ldots, t_N], i = [1, N]\}$
    for all $X \in \{Anat, Coord, Qual, Qualif\}$ **do**
        for i:= 1 to N **do**
            Consider $LexX_C\_TOKENS_i$ // E.g., $LexAnat_C\_TOKENS_i$
            $SimX_i = similarity\_matrix\_and\_traces(LexX_C\_TOKENS_i, HPO_C\_TOKENS)$
            $OptSimX_i = length\_based\_optimal\_traces(SimX_i, HPO_C\_TOKENS)$
        end for
    end for

    SimAnat = $\{SimAnat_N, \ldots, SimAnat_1\}$
    FullSimAnat = full_anat_traces(HPO_C_TOKENS, SimAnat)

    $HPO_C\_SIM_i$ = aggregated_similarity(OptSimX, FullSimAnat), X ∈ {Coord, Qual, Qualif}
    // See Section 4.3 for the aggregated similarity computation
    $HPO_C\_SIM = max\|HPO_C\_SIM_i\|$
end for

Algorithm 2. Similarity matrix and traces

Require: $LexX_C\_TOKENS_i, HPO_C\_TOKENS$

TRACES = {}
for j:= 1 to N **do**
    SIM_MAT_j = similarity_matrix($LexX_C\_TOKENS_i, HPO_C\_ENTRY_j$)
    // See Section 4.2 for the similarity matrix and traces computation

    Trace($HPO_C\_TOKENS_j$) = compute_traces(SIM_MAT_j)
    // Trace($HPO_C\_TOKENS_j$) = {Value, Length, Start_Index}

    TRACES = TRACES ∪ $\{HPO_C\_ENTRY_j \rightarrow Trace(HPO_C\_TOKENS_j)\}$
end for

return TRACES

Length-Based Optimal Traces. For each entry in each lexicon of the PFO concept, this step reduces the list of associations produced by the previous one by choosing the maximal trace for a particular length. Continuing the example above, for *Phalanx of finger* this step will produce:

- Length:3 ← (bifid distal phalanx of thumb – 0.66)
- Length:1 ← (notched terminal thumb phalanx – 0.59)

Full Anatomy Traces. Until this point each element of the PFO concept has been considered individually, including the different anatomical parts. In our example, we have calculated the length-based optimal traces for both *Phalanx of finger* (and the rest of its lexicon entries), as well as for *Thumb* (and the rest of its lexicon entries). This step reunites all anatomical parts by looking for the optimal full anatomy trace for each HPO lexicon entry. This is done by averaging the individual anatomical traces for a particular HPO lexicon entry and then choosing the trace with the highest score. For example, if for the HPO lexicon entry *bifid distal phalanx of thumb* we have the following:

- 1: (Phalanx of finger – 0.59), (Thumb – 0.99) → 0.79
- 2: (Hand phalanx – 0.34), (Thumb – 0.99) → 0.66

this step will choose option 1 has being the optimal full anatomy trace.

Similarity Aggregration. Taking the length based optimal traces and the full anatomy traces produced above, this step computes the final similarity as described in Section 4.3.

Algorithm 3. Length based optimal traces

Require: $SimX_i, HPO_C\_TOKENS$
1: LENGTH_BASED_OPT = {}
2: **for all** $HPO_C\_ENTRY_j \in SimX_i$ **do**
3: Trace ∈ LENGTH_BASED_OPT
4: **if** Trace.Length == $Trace(HPO_C\_TOKENS_j.Length$ **then**
5: LENGTH_BASED_OPT = LENGTH_BASED_OPT ∪ $max\|Trace(HPO_C\_TOKENS_j.Value, Trace.Value\|$
6: **end if**
7: **end for**
8: return LENGTH_BASED_OPT

Algorithm 4. Full anatomy traces

Require: HPO_C_TOKENS, SimAnat
1: FULL_SIM = {}
2: **for all** $HPO_C\_ENTRY_j$ **do**
3: OPT_FULL_SIM = $max\|\frac{1}{N} * \sum_{i=1}^{N} SimAnat_{ij}\|$
4: FULL_SIM = FULL_SIM ∪ $\{HPO_C\_ENTRY_j \rightarrow OPT\_FULL\_SIM\}$
5: **end for**
6: return FULL_SIM

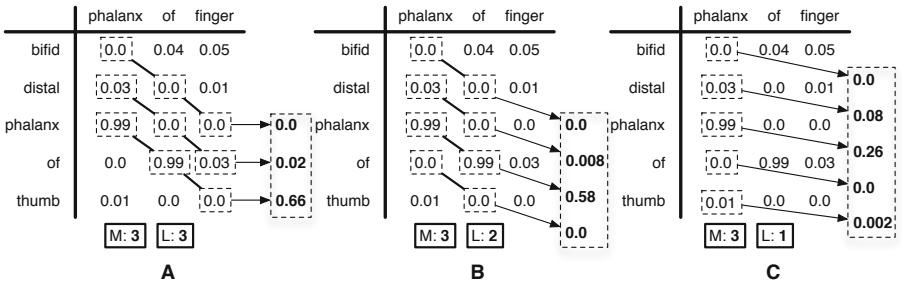


Fig. 3. Example of similarity matrix and traces computation for L = 3, 2, 1

4.2 Similarity Matrix and Traces Computation

The similarity matrix and traces computation is always performed on pair of tokenized lexicon entries, e.g., a lexicon entry for an anatomical part such as *Phalanx of finger* and a HPO concept lexicon entry, such as *bifid distal phalanx of thumb*. The goal of this step is to find the segment in the HPO concept lexicon entry that best matches the lexicon entry of, in our example, the anatomical part for different lengths of this last lexicon entry. We start by creating a M x N similarity matrix, where M is the length of the anatomical part and N is the length of the HPO lexicon entry. Fig. 3 depicts a full example of this step using the above pair of lexicon entries. The values in the similarity matrix are given by the following string similarity metric:

$$sim(s_1, s_2) = w_1 * sim_1(s_1, s_2) + w_2 * sim_2(s_1, s_2) + w_3 * sim_3(s_1, s_2) \quad (1)$$

where sim_1, sim_2 and sim_3 are defined below and correspond to the normalized longest common subsequence (LCS) and the normalized maximal consecutive longest common subsequence (MCLCS) starting at 1 (i.e., with the first character) and respectively at n (i.e., starting anywhere in the string). The weights w_1, w_2 and w_3 have been set empirically to 0.3, 0.6 and 0.1, because in the case of anatomical parts their prefix and root provide a higher accuracy in similarity matching (e.g.., *tibia*, *tibiae*, *tibial*).

$$sim_1 = NLCS(s_1, s_2) = \frac{length(LCS(s_1, s_2))^2}{length(s_1) * length(s_2)} \quad (2)$$

$$sim_2 = NMCLCS_1(s_1, s_2) = \frac{length(MCLCS_1(s_1, s_2))^2}{length(s_1) * length(s_2)} \quad (3)$$

$$sim_3 = NMCLCS_n(s_1, s_2) = \frac{length(MCLCS_n(s_1, s_2))^2}{length(s_1) * length(s_2)} \quad (4)$$

The trace of the similarity matrix is the multiplication of the arithmetic mean of the matrix diagonal for a given length L $\leq$ M with a penalty factor, as per Eq. 5. The penalty factor (the first component of Eq. 5) is a monotonically decreasing

function that penalises the trace for all the non-stop word tokens omitted from the initial lexicon entry (i.e., M - L). For example, part C of Fig. 3 shows the trace computation for L = 1, i.e., the token *Phalanx*. In this case, the penalty factor depends on M - L = 1 (*finger*), since the token *of* is a stop word.

$$Trace(SimMat) = (e^{-\frac{M-L}{M}} - \frac{M-L}{M*e}) * \frac{\sum_{i=1}^{L} SimMat_{ii}}{L} \quad (5)$$

4.3 Similarity Aggregation

The overall similarity is the paired aggregation of the similarities of each of the four elements of the PFO concept, i.e., Anat – Coord and Qual – Qualif. We consider them in pairs because the anatomical coordinate is an extension of the anatomical part, and hence it is directly dependent on it, while the qualifier is an extension of the quality. Eq. 6 shows the overall similarity, with sim_{A-C} (i.e., the joint Anat – Coord similarity) being expressed in Eq. 7 and sim_{Q-Q} (i.e., the joint Qual – Qualif similarity) being expressed in Eq. 8. The two components of the overall similarity are: (i) a penalty factor depending on t_L – the number of HPO tokens left out from the similarity computation, and is the same as in the similarity matrix and trace calculation; and (ii) the aggregation of the two above mentioned similarities that gives more weight to the anatomical similarity.

The individual Anat – Coord similarity is computed by raising the multiplication of the final Coord trace with the arithmetic mean of all Anat traces to the power of e – the higher the multiplication score → the higher the similarity. Finally, the individual Qual – Qualif similarity is the arithmetic mean of the Qualif trace and the arithmetic mean of all Qual traces.

$$sim(PFO, HPO) = (e^{-t_L} - \frac{t_L}{e}) * \frac{6 * sim_{A-C} * sim_{Q-Q}}{2 * (2 * sim_{A-C} + sim_{Q-Q})} \quad (6)$$

$$sim_{A-C} = (\frac{TraceCoord}{N} * \sum_{i=1}^{N} TraceAnat_i)^e \quad (7)$$

$$sim_{Q-Q} = \frac{1}{2} * (TraceQualif + \sum_{i=1}^{N} TraceQual_i) \quad (8)$$

4.4 Broader Mapping Algorithm

The broader mapping algorithm extends the general one with two more steps. Firstly, it generates the ranked list of similarities on all HPO concepts using the general mapping algorithm and retains only those candidates that have the maximum similarity. Secondly, for each pair of candidates in the filtered list, it looks for the lowest common ancestor (LCA) from HPO and maps the LCA to the list of corresponding candidates. It then computes the standard deviation of the sizes of the candidates list associated with each LCA and retains only

Table 4. Evaluation results of the mapping process

Category	P@1	P@2	P@3	P@4	P@5
Exact match					
Simple	**0.85**	0.49	0.35	0.28	0.24
Composite anatomy	**0.75**	0.48	0.39	0.32	0.28
Broader match					
Simple	**0.91**	0.50	0.37	0.30	0.25
Composite Anatomy	**0.72**	0.46	0.36	0.31	0.27

those LCAs that have their corresponding list size greater than the standard deviation. Finally, these LCAs are used as input for another general mapping run, against the original PFO concept, however, this time by using only the anatomical and anatomical coordinates similarities. This last mapping is driven by the interpretation of the HPO hierarchy we have introduced in Section 2 in which the classification is done based on the anatomical localization, and hence the quality of the broader concept should be broader than the quality of the PFO concept under scrutiny.

5 Experimental Results

We have evaluated both mapping algorithms on all the concepts in the simple and composite anatomy category mentioned in Section 2. As already mentioned, their logical definitions can be found at `http://purl.org/skeletome/spo`. Similarly to the experiment described in Section 2, and following the goal set in Section 1, we have looked at precision at k (P@k) with k=1, 2, 3, 4, 5 (see Table 4).

The exact mapping of simple findings has achieved a maximal P@1 of 0.85, while the best exact matching on composite anatomy findings has been 0.75 at P@1. In both experiments (i.e., exact and broader) the composite anatomy has achieved lower precision results due of increased number of false positives it may introduce for each of the composing anatomical parts. A careful analysis of the missed mappings has revealed the following aspects:

1. Most of the failed mappings, especially in the composite anatomy category, are due to the HPO inconsistencies at the lexical representation level (as mentioned in Section 2). More concretely, three aspects have caused issues: (1) inconsistencies in using proper quality terms – i.e., using terms such as *hypoplastic* and *short / small*, or *aplastic* and *absent* in an alternative manner, although such terms have a clear individual semantics; (2) ambiguous quality definitions – i.e., *hypoplasia / small* as a quality of a finding; (3) the presence of generic synonyms in specific terms – i.e., *Absent middle phalanges* listed as synonym of *Aplasia of the middle phalanges of the hand*.

2. The logical definition of certain PFO Anatomical Composites introduces noise. For example, a simplified logical definition of `Middle phalanges`, without

considering cardinality and anatomical coordinate, is the union of `FMA: Phalanx of finger` and `FMA: Phalanx of toe`. The mapping process uses both underpinning FMA concepts for lexicon generation and hence creates an entire series of false positives. A slightly different example is `Pubic rami`, which is an union of `FMA: Inferior pubic ramus` and `FMA: Superior pubic ramus`.

3. Some of the FMA synonyms are too verbose and thus lead to false positives. For example, `FMA: Diaphysis` has listed *Body of long bone* as synonym. While *Diaphysis* is a fairly unique term and achieves a low similarity score against almost every other anatomical part (based on our function), our optimal trace calculation method may choose its synonym as a better pick, due to the high similarity values it produces (even if it is always penalised).

The broader mapping has been only partly affected by the above issues, since some of them are discarded when looking for the lowest common ancestor, and has achieved a maximal P@1 of 0.91 for the simple and 0.72 for the composite anatomy category. However, here the issues have shifted towards the second challenge mentioned in Section 2, i.e., the structure of HPO. Firstly, the multiple inheritance aspect of HPO influences the computation of the lowest common ancestor (LCA). Consequently, this may lead to the pruning of an entire set of relevant HPO terms in the advantage of a LCA which represented by a set of false positives. Subsequently, this is used in the final similarity computation where it achieves a very low score (see example in Section 2). Secondly, the distribution of certain phenotypes is heavily skewed and produces an almost linear branch of the corresponding abnormality. As a result, the algorithm picks concepts that are more specific than required. For example, `Retarded_ossification_of_the_femoral_neck` will be associated with the broader concept `HP_0006429` (*Broad femoral neck*) instead of `HP_0003367` (*Abnormality of the femoral neck* – the super class of `HP_0006429`), because it has a larger number of relevant children than relevant siblings.

Unfortunately, most of the issues listed above cannot be cleanly fixed from an algorithmic perspective without introducing specific work-arounds. On the HPO side, we will work together with the HPO maintainers on addressing the aspects we have mentioned. On our side, we will focus on making the algorithm more robust, in order to deal especially with the verbosity of the FMA synonyms.

6 Conclusion

In this paper we have reported on our efforts in creating a standard representation for radiographic findings in the skeletal dysplasia domain and mapping concepts modeled in this representation to Human Phenotype Ontology terms . We have shown that our Phenotype Fragment Ontology provides a flexible metamodel that bridges the diverse lexical groundings of the radiographic findings by using widely adopted ontologies to underpin the actual concepts definition. Subsequently, we have described an exact and a broader matching algorithm able to efficiently map PFO concepts to HPO terms. From an application perspective, the SKELETOME platform now uses as part of its semantic annotation

process both HPO terms, and PFO instances, in particular where appropriate HPO terms are not available. Future work will focus on extending PFO to cater for a series of particular cases, such as combinations of atomic phenotypes, e.g., *Acromesomelic brachymelia*.

Acknowledgments. The work presented in this paper is supported by the Australian Research Council (ARC) under the Discovery Early Career Researcher Award (DECRA) – DE120100508 and the Linkage grant SKELETOME – LP100100156. The experiments detailed in the paper have used material from "*Bone dysplasia: an atlas of genetic disorders of skeletal development*" by Jurgen W. Spranger, Paula W. Brill, Andrew K. Poznanski (2002) – By permission of Oxford University Press, Inc.

References

1. Groza, T., Zankl, A., Li, Y.-F., Hunter, J.: Using Semantic Web Technologies to Build a Community-Driven Knowledge Curation Platform for the Skeletal Dysplasia Domain. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 81–96. Springer, Heidelberg (2011)
2. Robinson, P.N., Kohler, S., Bauer, S., Seelow, D., Horn, D., Mundlos, S.: The Human Phenotype Ontology: A Tool for Annotating and Analyzing Human Hereditary Disease. The American Journal of Human Genetics 83(5), 610–615 (2008)
3. Rosse, C., Mejino, J.L.V.: A reference ontology for biomedical informatics: the Foundational Model of Anatomy. J. of Biomed. Inf. 36(6), 478–500 (2003)
4. Gkoutos, G.V., et al.: Entity/Quality-Based Logical Definitions for the Human Skeletal Phenome using PATO. In: Proceedings of the 31st Annual International Conference of the IEEE EMBS, Minneapolis, Minnesota, USA, pp. 7069–7072 (2009)
5. Hu, W., Qu, Y., Cheng, G.: Matching large ontologies: A divide-and-conquer approach. IEEE Data and Knowledge Engineering 67(1), 140–160 (2008)
6. Li, J., et al.: RiMOM: A dynamic multi-strategy ontology alignment framework. IEEE Data and Knowledge Engineering 21(8), 1218–1232 (2009)
7. Lambrix, P., Tan, H.: SAMBO – A system for aligning and merging biomedical ontologies. Journal of Web Semantics 4(1), 196–206 (2006)
8. Nagy, M., Vargas-Vera, M.: Multi-Agent Ontology Mapping Framework for the Semantic Web. IEEE Transactions on Systems, Man, and Cybernetics – Part A 41(4), 693–704 (2011)
9. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Transactions on Knowledge and Data Engineering (2012)
10. Jonquet, C., et al.: The open biomedical annotator. In: Proc. of the 2010 AMIA Summit of Translational Bioinformatics, San Francisco, CA, US, pp. 56–60 (2010)

Toward an Ecosystem of LOD in the Field: LOD Content Generation and Its Consuming Service

Takahiro Kawamura[1,2] and Akihiko Ohsuga[2]

[1] Corporate Research & Development Center, Toshiba Corp.
[2] Graduate School of Information Systems,
University of Electro-Communications, Japan

Abstract. This paper proposes to apply semantic technologies in a new domain, Field research. It is said that if "raw data" is openly available on the Web, it will be used by other people to do wonderful things. But, it would be better to show a use case together with that data, especially in the dawn of LOD. Therefore, we are proceeding with both of LOD content generation and its application for a specific domain. The application addresses an issue of information retrieval in the field, and the mechanism of LOD generation from the Web might be applied to the other domain. Firstly, we demonstrate the use of our mobile application, which searches a plant fitting the environmental conditions obtained by the smartphone's sensors. Then, we introduce our approach of the LOD generation, and present an evaluation showing its practical effectiveness.

Keywords: Sensor, LOD, AR, Plant, Field.

1 Introduction

Semantic search is intrinsically suited for information retrieval in the field, where a trial-and-error approach to search is difficult because input is less convenient and the network tends to be slower than in the case of desktop search. It is burdensome for users in the field to research something while changing keywords and looking through a list of the results repeatedly. Therefore, search with SPARQL, which can specify the necessary semantics, would be useful in the field. Moreover, exploitation of mobile and facility sensors is now prevailing, but applications are still vague although sensor information is overflowing. Thus, LOD can serve as an intermediary interpreting the semantics of the users and their environmental information obtained by the sensors and connecting it to the collective intelligence on the net. LOD and SPARQL have the tremendous potential in the field. However, to build ecosystem of LOD used in the field, it requires at least the actual LOD content for the field, and its appealing application which consumes that LOD. In the talk of Tim Berners-Lee at TED2009 and 2010 [1], it is intended that someone publishes data, and then the other one

[1] `http://www.ted.com/talks/tim_berners_lee_on_the_ next_web.html`

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 98–113, 2012.

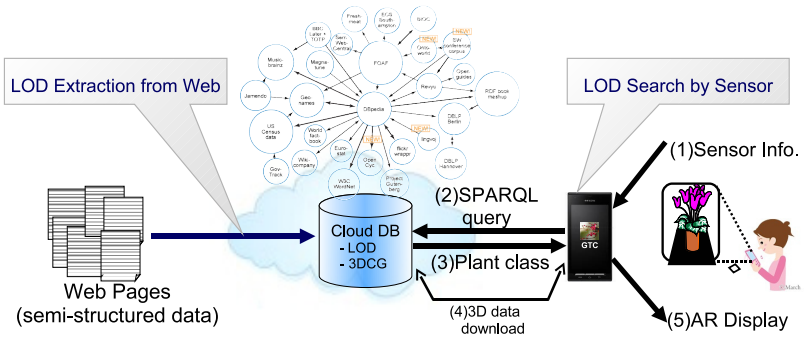


Fig. 1. Proposed architecture of LOD use

will use it, and create application mashups. But it would be better to show a use case with the data, in which the linking data is semantically utilized. Therefore, we would like to propose both of a mechanism of LOD content generation and its concrete application in this paper.

The remainder of this paper is organized as follows. Section 2 outlines related work, and then firstly we introduce a field application of LOD, where LOD is searched based on the sensor information on a smartphone in section 3. Next, section 4 describes a mechanism of LOD content generation, where LOD is extracted from the Web. Finally, section 5 presents conclusions and identifies future issues.

2 Related Work

First, we introduce two researches regarding architecture using sensors and semantics, and its application. The first one is Semantic Sensor Network (SSN), in which sensor data is annotated with semantic metadata mainly to support environmental monitoring and decision-making. SemSorGrid4Env [1] is applying it to flood emergency response planning. Our architecture (Fig. 1) is similar to SSN. However, instead of searching and reasoning within the collected semantic sensor data, we assume the existence of LOD on the net, to which the sensor data is connected. In that sence, SENSEI[2] had almost the same purpose to integrate the physical with the digital world. But the project mainly addressed the scalability issue and the definition of services interfaces, and then LOD content was limited to a few types of data like geospatial.

The second one is about social sensor research, which integrates the existing social networking services and physical-presence awareness like RFID and twitter with GPS data to encourage users' collaboration and communication. Live Social Semantics (LSS) [3] applied it to some conferences and suggested new interests for the users. It resembles our architecture in that face-to-face contact events based on RFID are connected to the social information on the net. However, from the difference in its objective, which is a social or field support, the information flow

is opposite. In our architecture, the sensor (client) side requests the LOD on the net, although in LSS the social information (DB) collects the sensor data.

Next, we introduce a research regarding LOD content generation. There are several ways and their combinations to generate LOD content. The first one is that an expert writes about a particular theme, e.g. data of Open Government. Also, there is a way to generate LOD as well as creating the content using CMS (Content Management Systems). The second and third ones are user participatory creation, e.g. DBPedia[4] and Freebase[5], and crowdsourcing, e.g. use of Amazon Mechanical Turk. Both of them use the power of the masses (or collective intelligence), but are classified according to the presence of business contract. The fourth one is conversion from the existing structured data like table, CSV and RDB using XLWrap[6] and OntoAccess[7], e.g. Life Science data, and then the last way we think is the (semi-)automatic generation of LOD from the Web. In the recent conferences, researches on the (semi-)automatic generation seems small in number, compared to LOD utilization under the premises that large-scale datasets have been provided, though there are many researches to build an ontology from the Web. But, one of them is NELL (Never-Ending Language Learner) presented by T. Mitchell at ISWC09[8] and more details at AAAI10[9], which is a semantic machine learning system using the existing ontologies, where several learning methods are combined to reduce extraction errors. Our generation method has been greatly inspired by NELL. The detailed description will be shown in section 4.

3 Field Application of LOD

3.1 Problem Statement

Home gardens and green interiors have been receiving increased attention owing to the rise of environmental consciousness and growing interest in macrobiotics. However, the cultivation of greenery in a restricted urban space is not necessarily a simple matter. In particular, as the need to select greenery to fit the space is a challenge for those without gardening expertise, overgrowth or extinction may occur. In regard to both interior and exterior greenery, it is important to achieve an aesthetic balance between the greenery and the surroundings, but it is difficult for amateurs to imagine the future form of the mature greenery. Even if the user checks images of mature greenery in gardening books, there will inevitably be a gap between the reality and the user's imagination. To solve these problems, the user may engage the services of a professional gardening advisor, but this involves cost and may not be readily available. Therefore, we considered it would be helpful if an 'agent' service offering gardening expertise were available on the user's mobile device. In this section, we describe our development of Green-Thumb Camera, which recommends a plant to fit the user's environmental conditions (sunlight, temperature, etc.) by using a smartphone's sensors. Moreover, by displaying its mature form as 3DCG using AR (augmented reality) techniques, the user can visually check if the plant matches the user's

surroundings. Thus, a user without gardening expertise is able to select a plant to fit the space and achieve aesthetic balance with the surroundings.

The AR in this paper refers to annotation of computational information to suit human perception, in particular, overlapping of 3DCG with real images. This technique's development dates back to the 1990s, but lately it has been attracting growing attention, primarily because of its suitability for recent mobile devices. AR on mobile devices realizes the fusion of reality and computational information everywhere. Research[10] on AR for mobile devices was conducted in the 1990s, but it did not attract public attention because "mobile" computers and sensors were big and hard to carry, and the network was slow.

Plant recommendation involves at least two problems. One problem concerns plant selection in accordance with several environmental conditions of the planting space. There are more than 300,000 plant species on the Earth , and around 4,000 plant species exist in Japan. Also, their growth conditions involve a number of factors such as sunlight, temperature, humidity, soil (chemical nutrition, physical structure), wind and their chronological changes. Therefore, we have incorporated the essence of precision farming[11], in which those factors are carefully observed and analyzed, and crop yields are maximized through optimized cultivation. In our research, firstly, using the sensors on the smartphone, we determine the environmental factors listed in Table 1, which we consider to be the major factors, and then try to select a plant based on those factors. Other factors, notably watering and fertilizing, are assumed to be sufficient. We intend to incorporate other factors in the near future [2].

Another problem concerns visualization of the future grown form. As well as the need to achieve aesthetic balance for both interior and exterior greenery, overgrowth is an issue. In fact, some kinds of plant cannot be easily exterminated. Typical examples of feral plants are vines such as Sicyos angulatus, which is designated as an invasive alien species in Japan, and Papaver dubium, which has a bright orange flower and is now massively propagating in Tokyo. Therefore, we propose visualization of the grown form by AR to check it in advance.

3.2 Plant Recommendation Service

This section explains the service that we propose.

Service Flow of Plant Recommendation. Firstly, the user puts an AR marker (described later) at the place where he/she wants to grow a plant , and then taps an Android application, Green-Thumb Camera (GTC), and pushes a start button. If the user looks at the marker through a camera view on the GTC App (Fig. 1), the app (1) obtains the environmental factors, such as sunlight, location and temperature from the sensor information (2) searches on LOD Cloud DB with SPARQL, and (3) receives some Plant classes that fit the environment.

[2] A bioscience researcher whom we consulted confirmed that the factors listed in Table 1 are sufficient to serve as the basis for plant recommendation to a considerable extent.

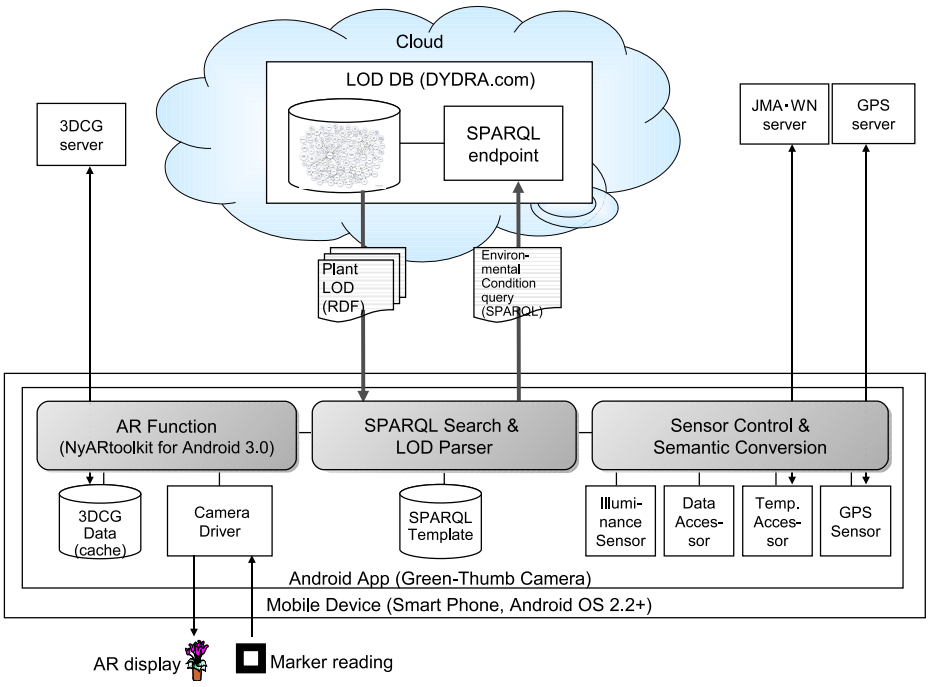


Fig. 2. Green-Thumb Camera

Then, the app (4) downloads 3DCG data for the plants, if necessary (the data once downloaded is stored in the local SD card), (5) overlays the 3DCG on the marker in the camera view. It also shows two tickers, one for the plant name and description below, and another for the retrieved sensor information on the top. Fig. 3 is an example displaying "Begonia". It is difficult to find a plant which can survive in a shade garden, so that we can find that the service is helpful. If the user does not like the displayed plant, he/she can check the next possible plant by clicking 'prev' or 'next' button, or flicking the camera view. Furthermore, if the user clicks a center button, GTC shows a grown form of the plant. In this way, the user would be able to find a plant that fits the environmental conditions and blends in with the surroundings. Fig. 2 shows the overview of this service.

Semantic Conversion from Sensor Information to Environmental Factors. This section describes the environmental factors, and how we convert raw data of the sensors to them. Table 1 shows the factors considered in this paper.

Sunlight
: This factor indicates the illuminance suitable for growing each plant and has several levels such as shade, light shade, sunny, and full sun [12,13]. To determine the current sunlight, we used a built-in illuminance sensor on the smartphone. After the application boots up, if the user brings the smart-

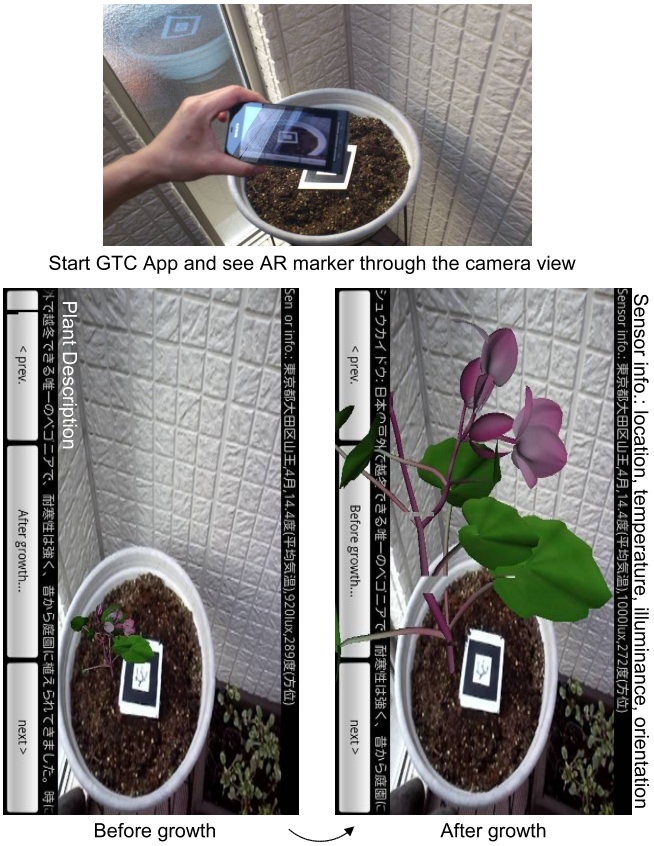


Fig. 3. Example of a plant display (left: before growth, right: after growth)

phone to the space where he/she envisages putting the plant and pushes the start button on the screen, the sunlight at the space is measured, and classified as the above levels. If it is less than 300 lux, it is deemed to be a shade area. If it is more than 300 lux but less than 3000 lux, it is deemed to be light shade, and If it is more than 3000 lux but less than 10000 lux, it is deemed to be sunny. Then, if more than 10000 lux, it is deemed to be a full sun area. In the case that the sunlight taken by the sensor fits that for the plant, it is deemed suitable.

Temperature

This factor indicates the range (min, max) of suitable temperature for a plant. The lower and the upper limits of the range are determined by reference to the sites as well as to the sunlight. To get the temperature, we referred to past monthly average temperatures for each prefecture from the Japan Meteorological Agency(JMA) [14] , using the current month and area (described below), instead of the current temperature. The temperature for

Table 1. Environmental factors

Factor	Description
Sunlight	minimum and maximum illuminance
Temperature	minimum and maximum temperature
Planting Season	optimum period of planting
Planting Area	possible area of planting

indoor plants from November to February is the average winter indoor temperature for each prefecture from WEATHERNEWS INC.(WN)[15]. In the case that the temperature taken by the sensor is within the range of the plant, it is deemed suitable.

Planting Season

The planting season means a suitable period (start, end) for starting to grow a plant (planting or sowing). The periods are set on a monthly basis according to some gardening sites[16,17]. To get the current month, we simply used the Calendar class provided by the Android OS. However, the season is affected by the geographical location (described below). Therefore, it is set one month later in the south area, and one month earlier in the north area. In the northernmost area, it is set two months earlier. , because the periods are given mainly for Tokyo (middle of Japan) on most websites. If the current month is in the planting season for the plant, it is deemed suitable.

Planting Area

The planting area means a suitable area for growing a plant. It is set by provincial area according to a reference book used by professional gardeners[12]. To get the current area, we used the GPS function on the smartphone. Then, we classified the current location (latitude, longitude) for the 47 prefectures in Japan, and determined the provincial area. If the current location is in the area for the plant, it is deemed suitable.

Recommendation Mechanism. In this section, we describe how a plant is recommended based on the above factors.

As a recommendation mechanism, we firstly tried to formulate a function on the basis of multivariate analysis, but gave it up because priority factors differ depending on the plant. Next, we created a decision tree per plant because the reasons for recommendation are relatively easily analyzed from the tree structure , and then we evaluated the recommendation accuracy[18]D However, this approach obviously poses a difficulty in terms of scaling up since manual creation of training data is costly. Therefore, we prepared Plant LOD based on collective intelligence on the net and adopted an approach of selecting a plant by querying with SPARQL.

There are several DBs of plants targeting such fields as gene analysis and medical applications. However, their diverse usages make it practically impossible to unify the schemas. Furthermore, there are lots of gardening sites for hobbyists,

and the practical experience they describe would also be useful. Therefore, instead of a Plant DB with a static schema, we adopted the approach of virtually organizing them using LOD on the cloud. Thus, we developed a semi-automatic generation system for Plant LOD, and combined the collected data with the DBPedia. The details are described in the next section.

The SPARQL query includes the above-mentioned environmental factors obtained from the sensors in the FILTER evaluation, and is set to return the top three plants in the reverse order of the planting difficulty within the types of Plant class.

It should be noted that SPARQL 1.0 does not have a conditional branching statement such as IF-THEN or CASE-WHEN in SQL. Thus, certain restrictions are difficult to express, such as whether the current month is within the planting season or not. Different conditional expressions are required for two cases such as *March* to *July* and *October* to *March*. Of course, we can express such a restriction using logical-or(||) and logical-and(&&) in FILTER evaluation, or UNION keyword in WHERE clause. But, it would be a redundant expression in some cases (see below, where ?start, ?end, and MNT mean the start month, the end month, and the current month respectively). On the other hand, SPARQL 1.1 draft [19] includes IF as Functional Forms. So we expect the early fix of 1.1 specification and dissemination of its implementation.

```
SELECT distinct ...
WHERE{
...
FILTER(
...
&&
# Planting Season
( ( (xsd:integer(?start) <= MNT) && (MNT <= xsd:integer(?end)) ) ||
  ( (xsd:integer(?start) >= xsd:integer(?end)) &&
    (xsd:integer(?start) <= MNT) && (MNT <= 12) ) ||
  ( (xsd:integer(?start) >= xsd:integer(?end)) &&
    ( 1 <= MNT) && (MNT <= xsd:integer(?end)) ) )
&&
..
)
ORDER BY ASC (xsd:integer(?difficulty))
LIMIT 3
```

Listing 1.1. SPARQL query

AR Interface. This application requires a smartphone running Google Android OS 2.2+ and equipped with a camera, GPS, and a built-in illuminance sensor. For the AR function, we used NyARToolkit for Android [20], which is an AR library for the Android OS using a marker. It firstly detects the predefined marker (Fig. 4) in the camera view, recognizes its three-dimensional position and attitude, and then displays 3DCGs in Metasequoia format on the marker. The 3DCG can quickly change its size and tilt according to the marker's position and attitude through the camera. We have already prepared 90 kinds of plant 3DCG data for recommendation.

Fig. 4. AR marker (6cm × 6cm)

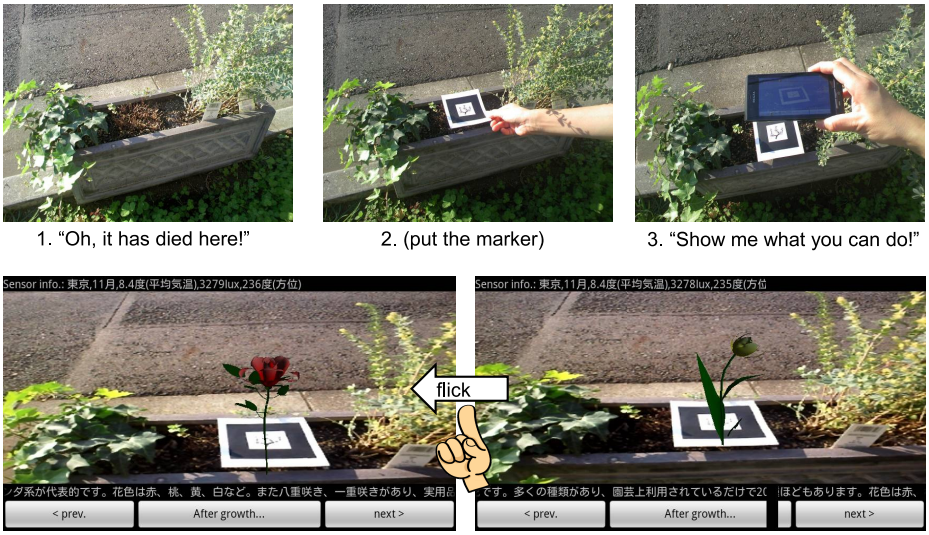


Fig. 5. Result of plant recommendation

3.3 Experimental Result of LOD Application

Fig. 5 shows an experimental result of the plant recommendation. The test environment was as follows: Tokyo, November, 3000+ lux, approx. 10 °C. If the user puts the marker at a place where he/she envisages putting a plant, and sees it through the camera, the GTC App reads the marker and gets the environmental factors such as sunlight, location, and temperature. Then, it overlays 3DCG of a recommended plant on the marker in the camera view. If the user views the marker from different angles and distances through the camera, it dynamically changes the 3DCG as if it were the real thing. Also, by flicking the camera view, the next plant in the order of recommendation is displayed.

In the figure, 3DCG of a rose and a tulip are displayed as a result. Those are typical candidates for planting in this season in Tokyo, and we confirmed the recommendation is working correctly. The GTC App is now open to the public, so anyone can download and try to use it [3]. In the near future, we are

[3] `http://www.ohsuga.is.uec.ac.jp/~kawamura/gtc.html` (in Japanese).

planning to conduct some evaluation by a group of potential users to determine it's effectiveness.

4 LOD Content Generation

4.1 Overview of Plant LOD

The Plant LOD (Fig. 6) is RDF data, in which each plant is an instance of "Plant" class of DBpedia ontology. DBpedia has already defined 10,000+ plants as types of the Plant class and its subclasses such as "FloweringPlant", "Moss" and "Fern". In addition, we created 90 plants mainly for species native to Japan. Each plant of the Plant class has almost 300 Properties, but most of them are inherited from "Thing", "Species" and "Eukaryote". So we added 19 properties to represent necessary attributes for plant cultivation, some of which correspond to { Japanese name, English name, country of origin, description, sunlight, temperature (min), temperature (max), planting season (start), planting season (end), blooming season (start), blooming season (end), watering amount, annual grass (true or false), related website, image URL, 3DCG URL, planting area, planting difficulty }. { name, country of origin, description, sunlight, temperature, planting season, blooming season, watering amount, planting difficulty }. Fig. 6 illustrates the overall architecture of the Plant LOD , where prefixes **gtc:** and **gtcprop:** mean newly created instances and properties. The Plant LOD is now stored in a cloud DB, DYDRA[4] and a SPARQL endpoint is offered to the public.

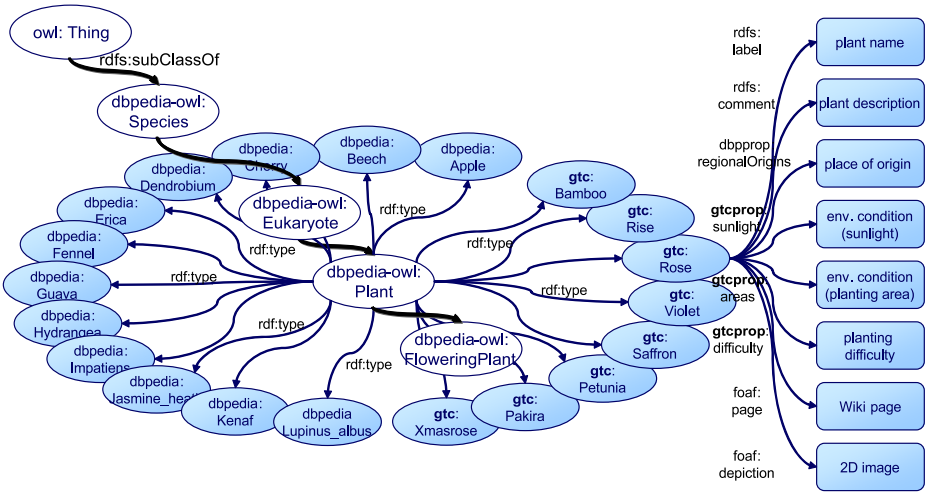


Fig. 6. Overview of Plant LOD

[4] http://dydra.com/

4.2 Semi-automatic Generation of LOD

In order to collect the necessary plant information from the Web and correlate it to the DBPedia, we developed a semi-automatic mechanism to grow the existing LOD, which involves a boot strapping method [21] based on ONTOMO [22] and a dependency parsing based on WOM Scouter [23]. But the plant names can be easily collected from a list on any gardening site and we have already defined the necessary properties based on our service requirements. Therefore, what we would like to collect in this case is the value of the property for each plant.

Process of our LOD generation is as follows (Fig. 7). First of all, we make a keyword list, which includes an instance name (plant name) and a logical disjunction of property names such as Rosemary ("Japanese name" OR "English name" OR "country of origin" OR ...), and then search on Google , and receive the result list, which includes more than 100 web pages. Next, we crawl the pages except for pdf files and also check Google PageRank for each page.

As the boot strapping method, we first extract specific patterns of DOM tree from a web page based on some keys, which are the property names (and their synonyms), and then we apply that patterns to other web pages to extract the values of the other properties. This method is mainly used for extraction of $(property, value)$ pairs from structured part of a page such as tables and lists (Fig. 8 left).

However, we found there are many (amateur) gardening sites that explain the nature of the plant only in plain text. Therefore, we developed an extraction method using the dependency parsing, because a triple $< plantname, property, value >$ corresponds to $< subject, verb, object >$ in some cases. It first follows modification relations in a sentence from a seed term, which is the plant name or the property name (and their synonyms), and then extract the triple, or a triple $< -, property, value >$ in the case of no subject in the sentence ('-' is replaced with the plant name in the keyword list later). See Fig. 8 right.

We combine all the property values obtained above, and filter it if it matches to co-occurrence strings with the corresponding property names, where a set of the co-occurrence string are prepared in advance, e.g. the propery "temperature" obviously co-occurs with a string °C. Then, we form some clusters of the identical property values for each property name based on LCS (Longest Common Substring). Furthermore, for correction of errors, which may be an error of extraction and/or of the information source, we sum up the PageRanks of the source pages for each cluster to determine the best possible property value and the second-best. Finally, after a user determines a correct value from the proposed ones, CSV and RDF files are generated to each plant.

In either way, the key or seed of the boot strapping and the dependency parsing are retrieved from our predefined schema of Plant LOD, that is, the instance name and the property name. It is our idea to put flesh on the bones of the existing LOD like the DBPedia in order to correlate to it.

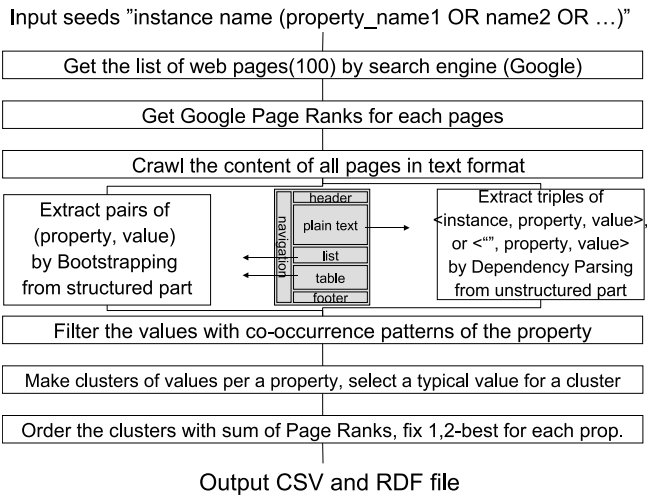


Fig. 7. Process of LOD content generation

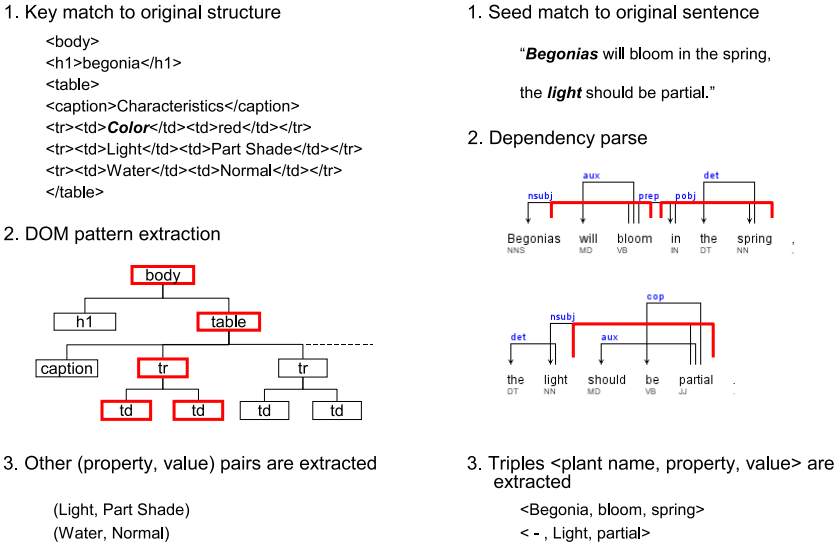


Fig. 8. Examples of boot strapping and dependency parsing

4.3 Evaluation of LOD Generation

Extraction Accuracy. We applied this LOD generation mechanism to extract the values of the 13 properties for the 90 plants that we added. The result shown in Table 2 includes an average presicion and recall of the best possible value (1-best) obtained through the whole process, the boot strapping method only, and

Table 2. Extraction accuracy

Accuracy(%)	1-best	2-best	1-best (bootstrap only)	1-best (dependency only)
Precision	85.2	97.4	88.6	85.2
Recall	76.9	87.2	46.2	76.9
Amount Ratio (%)	–	–	10.8	89.2

the dependency parsing only, and then these of the second possible value (2-best). It should be noted that we collected 100 web pages for each plant, but some reasons such as DOM parse errors and difference of file types reduced the amount to about 60%. In terms of determining the seasons (start month and end month), if the extracted period is subsumed by the correct period, and the gap between the start (end) months is within 1 month, then it is regarded as correct. Also, in terms of the temperature, if the gap is within 3 °C, it is regarded as correct. Properties like description, which are not clear whether it is true or not, are out of scope of this evaluation. If there are more than two clusters whose sum of the PageRanks are the same, we regarded them all as the first position. The accuracy is calculated in units of the cluster instead of each extracted value. That is, in the case of 1-best, a cluster which has the biggest PageRank corresponds to an answer for the property. In the case of 2-best, two clusters are compared with the correct value, and if either one of the two answers is correct, then it is regarded as correct (thus, it is slightly different than average precision).

$$N - best\ precision = \frac{1}{|D_q|} \sum_{1 \leq k \leq N} r_k$$

,where $|Dq|$ is is the number of correct answers for question q, and r_k is an indicator function equaling 1 if the item at rank k is correct, zero otherwise. The boot strapping method only and the dependency parsing only mean to form the clusters out of the values extracted only by the boot strapping and the dependency parsing, respectively. A cluster consists of the extracted values for a property, which seem identical accroding to LCS, but the number of the values in a cluster may vary from more than 10 to 1. Finally, if there are various theories as to the correct value for a property, we selected the most dominant one.

The best possible values (1-best) achieved an average precision of 85% and an average recall of 77%. But, the 2-best achieved an average precision of 97% and an average recall of 87%. So if we are permitted to show a binary choice to the user, it would be possible to present a correct answer in them in many cases. The accuracy of the automatic generation would not be 100% after all, and then a human checking is necessary at any step. Therefore, the binary choice would be a realistic option.

In detail, the boot strapping collects smaller amounts of values (11%), so the recall is substantially lower (46%) than the dependency parsing, but the precision is higher (89%). It is because data written in the tables can be correctly extracted, but lacks diversity of properties. Semantic drift of the values extracted by generic

patterns, which is a well-known problem in the boot strapping method, rarely happened here, because target sources are at most top 100 pages of the Google result, and the values are sorted by the PageRank at the end.

On the other hand, the dependency parsing collects large amount of values (89%), but it is a mixture of wheat and chaff. But, the total accuracy is affected by the dependency parsing, because the biggest cluster of the PageRank is composed mainly of the values extracted by the dependency parsing. So we will consider to put some weight on the values extracted by the boot strapping.

In addition, the accuracy varies by kind of the plant and the properties. The popularity of a plant affects the quantity and quality of information sources (web pages). Also, a specific property like "temperature (max)" is a minor information and the absolute number of descriptions are small.

Extraction Approach. In comparison with the above-mentioned NELL, our mechanism is the same as the NELL, as far as it is "Ontology-driven", "Macro-reading" which means that the input is a large text collection and the desired output is a large collection of facts, using "Machine learning methods". Recently, there have been several researches to extract information from semi-structured textual documents on the Web, which are combining NLP (Natural Language Processing) mechanisms or tools, and then using semantic resources like fine-grained ontologies. Among them, NERD (Named Entity Recognition and Disambiguation) framework[24] has also proposed an RDF/OWL-based NLP Interchange Format (NIF) and an API to unify various tools for a qualitative comparison. Both of the NELL and our mechanism are those of such researches.

However, instead of CPL (Coupled Pattern Learner) in NELL, we used a morphological analysis and a dependency parsing. Moreover, clustering of the values using LCS and the PageRank are also our own methods. But, a key strategic difference is a target domain of LOD generation. The NELL is targeting the world, so the granularity of the properties is big and the number of them is limited. For example, "agricultural product growing in state or province" is a barely fine-grain property in the NELL, but only 10 instances have that property, and also the number of all the properties is 5% of the total extraction. On the other hand, by restricting the domain of interest, the plant in this case, it is possible for our mechanism to construct the set of the co-occurrence string with the predefined property name. This simple heuristics effectively filter out candidates for the property values, thus raise the accuracy of the extraction and keep the variety of the properties together with the above methods. It obviously restricts applicability of the proposed technique, but is practically effective to generate useful LOD content.

Furthermore, the purpose of our mechanism is to grow the existing LOD in a specific domain, so that we extracted the correlating values according to the predefined schema from the Web. NELL also has the schema for broader domain than ours. In contrast, LODifier[25] has recently proposed a translation of textual information in its entirety into a structural RDF in open-domain scenarios with no predefined schema. In the future, we could use that technique as a pre-process of the whole document before matching the schema.

5 Conclusion and Future Work

Under the vision that LOD is suited for information retrieval in the field, we propose a mechanism of LOD content generation for a domain and its application to indicate the possible benefit of field use.

In the near future, we would like to apply this architecture of environmental sensing → semantic conversion → LOD Cloud (← collective intelligence) to more serious problems that would benefit from greater IT support. Now we are considering the provision of support for greening business, which addresses environmental concerns, and for agri-business in regard to the growing food problem. Also, we are planning to apply this architecture in the other fields than the plant. For example, by using a built-in GPS, and acceleration and orientation sensor of the smartphone, it is possible to understand users' outdoor behaviors, especially situation of their movements like walk, bus and train. Therefore, if traffic situation and accident data are provided as LOGD (Linking Open Government Data), we can mashup an application which shows safe navigation and precautions for the elderly people moving outside.

References

1. Gray, A.J.G., García-Castro, R., Kyzirakos, K., Karpathiotakis, M., Calbimonte, J.-P., Page, K., Sadler, J., Frazer, A., Galpin, I., Fernandes, A.A.A., Paton, N.W., Corcho, O., Koubarakis, M., De Roure, D., Martinez, K., Gómez-Pérez, A.: A Semantically Enabled Service Architecture for Mashups over Streaming and Stored Data. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part II. LNCS, vol. 6644, pp. 300–314. Springer, Heidelberg (2011)
2. Presser, M., Barnaghi, P.M., Eurich, M., Villalonga, C.: The SENSEI project. IEEE Communications Magazine 47(4) (2009)
3. Szomszor, M., Cattuto, C., Van den Broeck, W., Barrat, A., Alani, H.: Semantics, Sensors, and the Social Web: The Live Social Semantics Experiments. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part II. LNCS, vol. 6089, pp. 196–210. Springer, Heidelberg (2010)
4. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: A Nucleus for a Web of Open Data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
5. Freebase, http://www.freebase.com/.
6. Langegger, A., Wöß, W.: XLWrap – Querying and Integrating Arbitrary Spreadsheets with SPARQL. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 359–374. Springer, Heidelberg (2009)
7. Hert, M., Ghezzi, G., Würsch, M., Gall, H.C.: How to "Make a Bridge to the New Town" Using OntoAccess. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 112–127. Springer, Heidelberg (2011)

8. Mitchell, T.M., Betteridge, J., Carlson, A., Hruschka, E., Wang, R.: Populating the Semantic Web by Macro-reading Internet Text. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 998–1002. Springer, Heidelberg (2009)
9. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka Jr., E.R., Mitchell, T.M.: Toward an Architecture for Never-Ending Language Learning. In: Proc. of Conference on Artificial Intelligence, AAAI (2010)
10. Nagao, K.: Agent Augmented Reality: Agents Integrate the Real World with Cyberspace. In: Ishida, C.T. (ed.) Community Computing: Collaboration over Global Information Networks (1998)
11. Srinivasan, A.: Handbook of precision agriculture: principles and applications. Routledge (2006)
12. NEO-GREEN SPACE DESIGN. Seibundo Shinkosha Publishing (1996)
13. engeinavi (in Japanese), http://www.engeinavi.jp/db/
14. Japan Meteorological Agency, http://www.jma.go.jp/jma/indexe.html
15. weathernews, http://weathernews.com/?language=en
16. MAKIMO PLANT (in Japanese), http://www.makimo-plant.com/modules/maintenance/index.php
17. Angyo: The Village of Garden Plants, http://www.jurian.or.jp/en/index.html
18. Kawamura, T., Mishiro, N., Ohsuga, A.: Green-Thumb Phone: Development of AR-based Plant Recommendation Service on Smart Phone. In: Proc. of International Conference on Advanced Computing and Applications, ACOMP (2011)
19. W3C: SPARQL 1.1 Query Language Working Draft, May 12 (2011), http://www.w3.org/TR/2011/WD-sparql11-query-20110512/
20. NyARToolkit for Android, http://sourceforge.jp/projects/nyartoolkit-and/
21. Brin, S.: Extracting Patterns and Relations from the World Wide Web. In: Atzeni, P., Mendelzon, A.O., Mecca, G. (eds.) WebDB 1998. LNCS, vol. 1590, pp. 172–183. Springer, Heidelberg (1999)
22. Kawamura, T., Shin, I., Nakagawa, H., Nakayama, K., Tahara, Y., Ohsuga, A.: ONTOMO: Web Service for Ontology Building - Evaluation of Ontology Recommendation using Named Entity Extraction. In: Proc. of IADIS International Conference WWW/INTERNET 2010, ICWI (2010)
23. Kawamura, T., Nagano, S., Inaba, M., Mizoguchi, Y.: Mobile Service for Reputation Extraction from Weblogs - Public Experiment and Evaluation. In: Proc. of Twenty-Second Conference on Artificial Intelligence, AAAI (2007)
24. Rizzo, G., Troncy, R., Hellmann, S., Bruemmer, M.: NERD meets NIF: Lifting NLP Extraction Results to the Linked Data Cloud. In: Proc. of 5th Workshop on Linked Data on the Web, LDOW (2012)
25. Augenstein, I., Padó, S., Rudolph, S.: LODifier: Generating linked data from unstructured text. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 210–224. Springer, Heidelberg (2012)

Applying Semantic Web Technologies for Diagnosing Road Traffic Congestions

Freddy Lécué, Anika Schumann, and Marco Luca Sbodio

IBM Research, Smarter Cities Technology Centre
Damastown Industrial Estate, Dublin, Ireland
{firstname.lastname}@ie.ibm.com

Abstract. Diagnosis, or the method to connect causes to its effects, is an important reasoning task for obtaining insight on cities and reaching the concept of sustainable and smarter cities that is envisioned nowadays. This paper, focusing on transportation and its road traffic, presents how road traffic congestions can be detected and diagnosed in quasi real-time. We adapt pure Artificial Intelligence diagnosis techniques to fully exploit knowledge which is captured through relevant semantics-augmented stream and static data from various domains. Our prototype of semantic-aware diagnosis of road traffic congestions, experimented in Dublin Ireland, works efficiently with large, heterogeneous information sources and delivers value-added services to citizens and city managers in quasi real-time.

1 Introduction

Consider the case of city planning in anticipation of a large event that requires city-wide mobilization of urban resources - a key Republic of Ireland World Cup qualifier match in Croke Park, for example. By integrating and correlating partial observations from multiple data sources, we could infer that the unseasonable inclement weather, coupled with 83,000 people descending on one area in Dublin to watch a mid-week match on a normal working day, coupled with a lack of public parking, led to traffic chaos that was widely reported in the media, driving strong negative sentiment towards the handling of such events. Whilst such an analysis is a useful forensic tool for understanding *what went wrong* and *what were the causes* **after the event**, it is important to compute causes of such unexpected situations **in quasi real-time**. This ensures that city managers have a solid understanding of the issues that lead to an unexpected situation, and can then take appropriate corrective actions.

Even if traffic congestion can be easily detected, visualized and analyzed [1] through stream data and optimization mechanisms using existing data mining [2] and machine learning approaches [3], (i) *explaining* their causes, (ii) *predicting* their impact and (iii) *recommending* alternative solutions are more complex and challenging problems, mainly due to the lack of information, its correlation and interpretation. This work focuses on the former problem, also known as *diagnosis* i.e., identification of the nature and cause of road traffic congestion.

What could be the cause of a motorway traffic congestion? Is it broken traffic light, an accident, a large concert, some road works, a brief stall, a temporarily overcrowded

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 114–130, 2012.

highway entrance or exit? The latter are potential causes of road congestions which could happen in a city traffic. Unfortunately, it is not always straightforward to obtain clear and descriptive explanations on their reasons, especially in quasi real-time situations. Understanding potential causes is important for informing interested parties, for instance, car drivers and public authorities, in quasi real time. This is important not only for providing explanations to drivers who are sitting in bumper-to-bumper traffic, but also for ensuring that public authorities will take optimal decisions and appropriate actions (e.g., rerouting or changing traffic light strategy in case of an accident or a broken traffic light) in time, especially in case of emergency.

How do large events such as a concert could impact traffic conditions? Shall we expect delays? Is re-routing appropriate? Such questions remain open because (i) relevant data sets (e.g., road works, city events), (ii) their interlinking (e.g., road works and city events connected to the same city area) and (iii) historical traffic conditions (e.g., road works and congestion in Canal street on July 24th, 2010) are not fully and jointly exploited. Pure AI diagnosis approaches focus on point (iii) for inferring the cause-effect relationships while semantic web technologies tackle (i) and (ii) for integrating heterogeneous and large data. This work extends the scope of pure AI diagnosis approaches to compute accurate diagnoses for situations where cause-effect relationships have not been established before. The list of potential heterogeneous sources of effects (road traffic congestion) and their causes (e.g., road weather conditions, events) that we consider in our scenario are listed in Table 1. A large part of data is provided by DCC (Dublin City Council) through *dublinked.ie*[1] agreement, and hosted at IBM.

We applied semantic web technologies for integrating heterogeneous data and then enabling advanced analytics. We exploit static and stream data (stream for short) from the road traffic data by encoding their semantics using existing LOD vocabularies (*Semantic Data* column of Table 1) and ontologies we developed for missing concepts. Road congestions are captured by correlating Dublin City Bus streams with latter lightweight ontologies. Diagnosis results are retrieved and interpreted by exploiting semantic representation of historical data and infrastructure data such as road network and bus lines. The quasi real-time cause-effect analysis is then reported back to the users.

The remainder of this paper is organized as follows. In Section 2 we present how road congestions are captured. Section 3 presents our semantics-augmented approach for diagnosing road congestions. Section 4 presents details about the prototype implementation and reports some experiment results regarding its applicability and scalability. Section 5 briefly comments on related work. Section 6 draws some conclusions and talks about possible future directions.

2 Detecting Road Traffic Congestion Using Semantics of Stream

The model we consider to represent static background knowledge and semantics of stream data (a.k.a. evolving knowledge over time) is provided by an ontology. Dynamic knowledge is then captured by reasoning on these ontology-augmented data descriptions. We focus on W3C standard OWL 2 to represent such ontologies since its logic

[1] http://dublinked.ie/

(DL) offer good reasoning support for most of its expressive profile. This section[2] reviews (i) OWL 2 EL and its DL $\mathcal{EL}^{++}$ as a formal knowledge representation language to define (ii) ontology stream and infer (iii) road congestions. Fig.1 positions the reviewed elements in relation to our challenge: *diagnosing road congestions*.

Table 1. (Incomplete) Overview of Traffic Scenario Data sets (Dublin City Dependant)

	Data Source	Description	Format Type	Temporal Frequency (s)	Historic (mm/yyyy)	Size Estimation per day (GBytes)	Data Provider
Source of Effects	Dublin Bus	Vehicle activity (GPS location, line number, delay, stop flag)	SIRI[a] XML-based	20	11/2010	4-6	(Private) DCC
Source of Causes	Wunderground for Dublin	Real-time weather information	CSV	[5, 600] (depending on stations)	01/1996	[0.050, 1.5] (depending on stations)	(Public) Wunder-ground[b]
	Road Weather Condition (54 stations)		CSV	600	11/2010	0.1	(Public) NRA[c]
	Road Works and Maintenance		CSV	3600	11/2010	0.01	(Public) Dublinked[d]
	Events in Dublin	Events with small attendance	XML	Not considered	11/2011	0.001	(Public) Eventbrite[e]
		Events with large attendance			11/2011	0.05	(Public) Eventful[f]
Semantic Data	DBPedia	Structured facts extracted from wikipedia	RDF	No	No	3.5×10^6 concepts	(Public) DBPedia[g]
	Dublin City Roads (listing of type, junctions, GPS coordinate)		RDF	No	No	0.1	(Public) Linked-geodata[h]

[a] SIRI (Service Interface for Real Time Information) is a standard for exchanging real-time information about public transport services and vehicles - `http://siri.org.uk`

[b] `http://www.wunderground.com/weather/api` - http://www.wunderground.com/history/airport/EIDW/2012/5/28/DailyHistory.html?format=1

[c] NRA - National Roads Authority `http://www.nratraffic.ie/weather`

[d] http://dublinked.ie - Sample: http://www.dublinked.ie/datastore/metadata064.php

[e] https://www.eventbrite.com/api

[f] http://api.eventful.com

[g] http://dbpedia.org

[h] http://linkedgeodata.org

2.1 Background: OWL 2 EL and Its $\mathcal{EL}^{++}$ Description Logics

The selection of the OWL 2 EL profile has been guided by (i) the expressivity which was required to model semantics of data in our application domain (Table 1) and (ii) the complexity of its underlying reasoning e.g., subsumption in OWL 2 EL is in PTIME [4]. The DL $\mathcal{EL}^{++}$ [5] is the logic underpinning OWL 2 EL and the basis of many more expressive DL. For the sake of readability we illustrate semantic representation and reasoning using DL formalism.

[2] Semantic representations are illustrated in DL to keep descriptions as concise as possible.

A signature Σ, defined by $(\mathcal{CN}, \mathcal{RN}, \mathcal{IN})$, consists of 3 disjoint sets of (i) atomic concepts $\mathcal{CN}$, (ii) atomic roles $\mathcal{RN}$, and (iii) individuals $\mathcal{IN}$. Given a signature, the top concept $\top$, the bottom concept $\bot$, an atomic concept A, an individual a, an atomic role r, $\mathcal{EL}^{++}$ concept expressions C and D can be composed with constructs: $\top \mid \bot \mid A \mid C \sqcap D \mid \exists r.C \mid \{a\}$. We slightly abuse the notion of atomic concepts to include $\top$, $\bot$ and nominals [6] i.e., individuals appearing in concept definitions of form $\{a\}$.

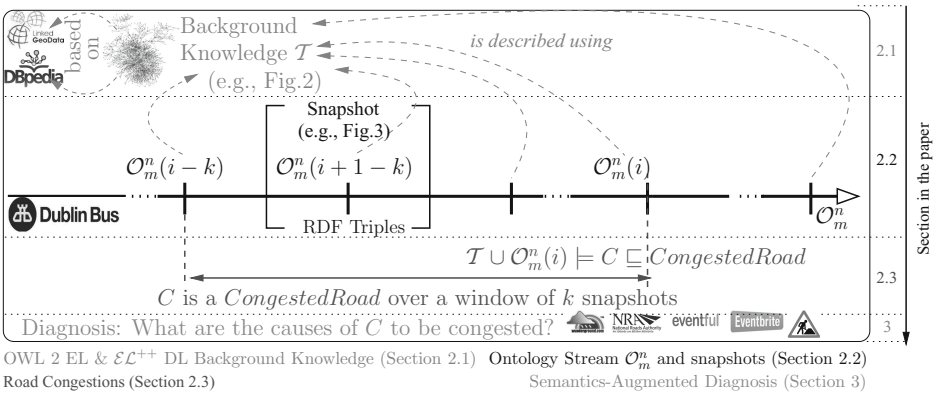


Fig. 1. Road Congestions Diagnosis through Background Knowledge and Ontology Stream

The particular DL-based ontology $\mathcal{O} \doteq< \mathcal{T}, \mathcal{A} >$, is composed of a TBox $\mathcal{T}$ and ABox $\mathcal{A}$. A TBox is a set of concept and role axioms.

$$Bus \sqsubseteq \exists id.BusID \sqcap \exists loc.GPSLocation \quad (1)$$
$$Road \sqsubseteq \exists id.RoadID \sqcap \exists in.GPSSetPoint \quad (2)$$
$$\exists id.BusID \sqcap \exists loc.(GPSLocation \sqcap \exists in.GPSSetPoint) \sqcap \exists id.RoadID \sqsubseteq Road \sqcap \exists with.Bus \quad (3)$$
$$Road \sqcap \exists with.(Bus \sqcap \exists congested.High) \sqsubseteq CongestedRoad \quad (4)$$
$$Bus \sqcap \exists delayed.High \sqsubseteq DelayedBus \quad (5)$$
$$\{r_1\} \sqsubseteq Road \sqcap \exists id.\{Canal\} \sqcap \exists roadPoint.\{(53.33, -6.27), (53.33, -6.28), (53.33, -6.29)\} \quad (6)$$

Fig. 2. Sample of an $\mathcal{EL}^{++}$ TBox $\mathcal{T}$ (GCI (6) is an internalized ABox axiom)

Example 1 ***($\mathcal{EL}^{++}$ DL Concept)***
A $CongestedRoad$ is a concept of a road with at least a congested bus (Fig.2).

$\mathcal{EL}^{++}$ supports General Concept Inclusion axioms (GCIs, e.g. $C \sqsubseteq D$ with C is subsumee and D subsumer) and role inclusion axioms (RIs, e.g., $r \sqsubseteq s$, $r_1 \circ \cdots \circ r_n \sqsubseteq s$). An ABox is a set of concept assertion axioms e.g., $a : C$, role assertion axioms e.g., $(a; b) : r$, and individual in/equality axioms e.g., $a \neq b$ or $a = b$.

We internalize ABox axioms into ($\rightsquigarrow$) TBox axioms so completion-based algorithms [5] can be applied to classify both axioms and entail subsumption. Thus TBox reasoning (subsumption, satisfiablility) can be performed on internalized ABox axioms.

$$a : C \rightsquigarrow \{a\} \sqsubseteq C \qquad (a, b) : r \rightsquigarrow \{a\} \sqsubseteq \exists r.\{b\}$$
$$a \doteq b \rightsquigarrow \{a\} \equiv \{b\} \qquad a \neq b \rightsquigarrow \{a\} \sqcap \{b\} \sqsubseteq \bot$$

Besides considering internalized ABox, we assume that $\mathcal{EL}^{++}$ TBox is normalized, and all subsumption closures are pre-computed [5]. We use the term background knowledge [7] to refer to such TBoxes.

2.2 Ontology Stream

An ontology stream [8] is considered as a sequence of ontologies (Definition 1) where knowledge is captured through its dynamic and evolutive versions.

Definition 1 ***(Ontology Stream)***
An ontology stream $\mathcal{O}_m^n$ from point of time m to point of time n is a sequence of ontologies $(\mathcal{O}_m^n(m), \mathcal{O}_m^n(m+1), \cdots, \mathcal{O}_m^n(n))$ where $m, n \in \mathbb{N}$ and $m < n$.

$\mathcal{O}_m^n(i)$ is a snapshot of an ontology stream (stream for short) $\mathcal{O}_m^n$ at point of time i, referring to a set of axioms in $\mathcal{L}$. A transition from $\mathcal{O}_m^n(i)$ to $\mathcal{O}_m^n(i+1)$ is an update.

$$\mathcal{O}_0^9(6) : \{bus31\} \sqsubseteq \exists id.\{dub31\} \sqcap \exists loc.\{(53.33, -6.27)\} \quad (7)$$
$$: \{bus31\} \sqsubseteq \exists congested.High \quad (8)$$

$$\mathcal{O}_0^9(7) : \{bus31\} \sqsubseteq \exists id.\{dub31\} \sqcap \exists loc.\{(53.33, -6.28)\} \quad (9)$$
$$: \{bus31\} \sqsubseteq \exists delayed.High \sqcap \exists congested.High \quad (10)$$

Fig. 3. Stream Snapshots: $\mathcal{O}_0^9(6)$ and $\mathcal{O}_0^9(7)$.

Example 2 ***(Ontology Stream)***
Fig.3 illustrates a partial ontology stream $\mathcal{O}_0^9$ along $\mathcal{O}_0^9(6)$ and $\mathcal{O}_0^9(7)$. Knowledge of snapshots is captured by GCIs e.g., $\{bus31\}$ is both delayed and congested in $\mathcal{O}_0^9(7)$.

2.3 Road Congestions

Road congestions (4) are derived by first capturing dynamic knowledge from the stream ontology, where the latter is then interpreted using background knowledge. Following (4), updating the definition of road congestions is straightforward.

Example 3 ***(Road Congestions)***
Road $\{r_1\}$ is a $CongestedRoad$ in $\mathcal{O}_0^9(7)$ with respect to GCIs (1-4), (6), (9-10).

In the following we will focus on diagnosing k-invariant road congestions i.e., congestions which remain persistent over a sequence of k snapshots. The diagnosis result is then extracted from this k-window i.e., all causes should occur in this window. Therefore, snapshots to be explored for diagnosis are pre-determined. In case of overlapping snapshots, the latter are considered once to avoid duplicate information, and all k snapshots are considered without any distinction.

Example 4 ***(k-Invariant Road Congestions)***
$\{bus31\} \sqsubseteq CongestedRoad$ *is a* 2*-invariant road congestion from* $\mathcal{O}_0^9(6)$ *to* $\mathcal{O}_0^9(7)$.

3 Semantics-Augmented Diagnosis

Diagnosis [9] is the task of explaining anomalies (e.g., congested roads) given a flow of observations. Interpreted in the context of the *Semantic* and *Stream Web*, anomalies are k-invariant road congestions and observations are captured from background knowledge $\mathcal{T}$ (e.g., any bus is conducted on roads) and dynamic knowledge of $\mathcal{O}_m^n$ (e.g., a bus is in a heavy traffic and a sport event is active in some snapshots). Our approach (Fig.4) elaborates an off-line diagnoser (Section 3.1) which aims at capturing *historical observations* over a *window timeframe* of k and their *explanations*. In other words diagnosis of historical anomalies is captured by the off-line diagnoser e.g., Canal street was congested in 2012, May 1^{st} at 6:00pm because of a concert event in Aviva stadium and road works in Bath avenue. Then quasi real-time diagnosis (Section 3.2) consists in retrieving "*similar*" causes (e.g., roads with heavy traffic of same duration) with "*similar*" conditions (e.g., close sport event) which have appeared in the past, and then reporting back their explanation through an interpretation in quasi real-time conditions.

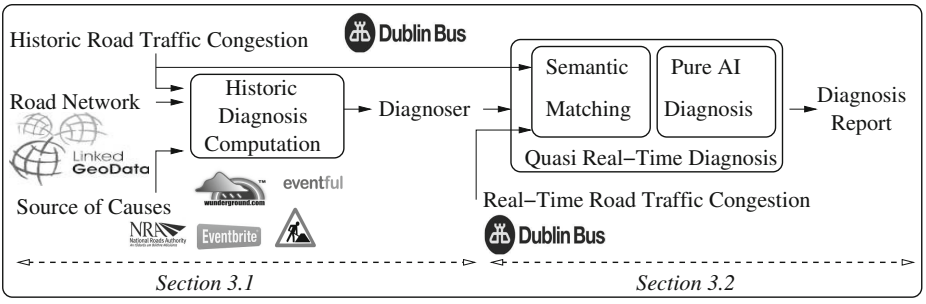


Fig. 4. Overview of the Semantics-Augmented Diagnosis Approach

3.1 Historic Diagnosis Computation

The traffic congestion diagnoser compiles off-line all historic diagnosis information into a deterministic finite state machine. The latter state machine is retrieved with respect to all RDF-augmented events, road works, road and weather conditions where a subset of them are connected to historic traffic congestions and the probability with which they have indeed caused it. Our traffic congestion diagnoser is strongly inspired from the Dublin City road network (using linkgeodata.org and complementary information[3]) to properly connect roads and to consider congestion propagation.

Illustrated in Fig.5, a diagnoser is defined by its states which are road intersections or car park locations that are associated to nearby events/road works. The latter association is done using their GPS geolocation and following the haversine formula [10] to

[3] http://www.dublinked.ie/datastore/metadata125.php

evaluate distances. The transitions of the diagnoser correspond to roads and each road is labeled by its historic diagnosis information where the latter is available for each snapshot of the day. Every two-way road, as bidirectional road in a city, corresponds to two roads in the diagnoser.

Example 5 Traffic Jam Diagnoser (where causes are defined as events) *Let 3 car park locations:* $\{l_1\}$, $\{l_2\}$, $\{l_3\}$ *used by people driving to events* $\{e_1\}$, $\{e_2\}$, $\{e_3\}$. *The transition labels of diagnoser in Fig.5 show the historic diagnosis of a snapshot* s. *The label* $(\{e_3\}, 0.6)$ *of road* $\{r_7\}$ *indicates that its cause to be congested (*C *stands for DL concept* $CongestedRoad$*) is event* $\{e_3\}$ *with a probability of* 0.6. *The probability is computed by retrieving from all historic records the probabilities:*

- *that* $\{r_7\}$ *was congested at snapshot* s *when* $\{e_3\}$ *took place , i.e.:*

$$p((\{r_7\} \sqsubseteq C)^s | \{e_3\}) := \frac{\textit{number of days with } \{e_3\} \textit{ and } \{r_7\} \textit{ being congested at } s}{\textit{number of days where } \{e_3\} \textit{ took place}}$$

- *that* $\{r_7\}$ *was congested at snapshot* s *when* $\{e_3\}$ *did not take place , i.e.:*

$$p((\{r_7\} \sqsubseteq C)^s | E \setminus \{e_3\}) := \frac{\textit{number of days without } \{e_3\} \textit{ and } \{r_7\} \textit{ being congested at } s}{\textit{number of days where } \{e_3\} \textit{ did not take place}}$$

$\{r_7\}$ *was congested on* 50% *of the days where* $\{e_3\}$ *took place and on* 20% *of the days where* $\{e_3\}$ *did not take place. Thus,* 20% *of the congestions on* $\{r_7\}$ *at snapshot* s *cannot be connected to city events while* 30% *of the congestions are caused by* $\{e_3\}$. *Thus, once we detect that* $\{r_7\}$ *is indeed congested we obtain that with a probability of* 0.6 *it was congested because of the upcoming event* $\{e_3\}$. *The events* e_1, e_2 *have no impact on the traffic situation of* r_7 *because* r_7 *is a "cul de sac". Formally:*

$$p^s_{r_7} := \frac{p((\{r_7\} \sqsubseteq C)^s | \{e_3\}) - p((\{r_7\} \sqsubseteq C)^s | \langle E \setminus \{e_3\}\rangle)}{p((\{r_7\} \sqsubseteq C)^s | \{e_3\})} \tag{11}$$

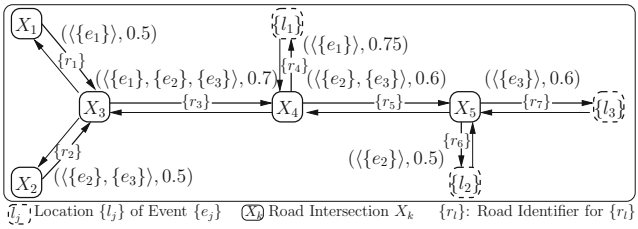


Fig. 5. A Traffic Jam Diagnoser (unlabeled roads have no causes of congestion)

We generalized the approach in a way that the traffic congestion diagnoser could handle road works, road weather and conditions as part of causes of a congested road. From a diagnosis point of view these causes are handled exactly like events $\{e_3\}$ in Example 5. From a semantic point of view this required the semantic description of road works, road weather and conditions and a corresponding matching function for determining their dis/similarity (Section 3.2) at various point of time.

3.2 Quasi Real-Time Diagnosis

As motivated in Section 1, pure AI diagnosis approaches [9,11] are not able to retrieve any diagnosis result of quasi real-time conditions (e.g., events or road works for which we do not have any historical records) if the latter do not exactly match at least one of the existing historical conditions. So how to compute diagnosis information with respect to new conditions? We tackle this problem by means of existing semantic techniques and define a matching function $Sim_{\mathcal{T}}$ for matching new DL concept-based C_n conditions and historic conditions C_h. Conditions, defined along city events, road works, weather and road conditions, are all represented using existing vocabularies such as DBpedia, SKOS[4], Talis[5], basic geo vocabulary[6] and internal IBM ontologies for handling basic generalization/specialization of new and historic conditions. For instance road works are represented through road asset type, work description and spread while events are represented through capacity, ownership and categories. The $Sim_{\mathcal{T}}$ function is based on the matchmaking functions introduced by [12] and [13]:

- **Exact.** If C_n and C_h are equivalent concepts: $\mathcal{T} \models C_n \equiv C_h$.
- **PlugIn.** If C_n is sub-concept of C_h: $\mathcal{T} \models C_n \sqsubseteq C_h$.
- **Subsume.** If C_n is super-concept of C_h: $\mathcal{T} \models C_h \sqsubseteq C_n$.
- **Intersection.** If the intersection of C_n and C_h is satisfiable: $\mathcal{T} \not\models C_n \sqcap C_h \sqsubseteq \bot$.

All conditions C_n are matched against every C_h using the latter function so the *"similarity"* of quasi real-time and historic conditions can be evaluated. Every pair of quasi real-time and historic conditions (C_n, C_h) is then ordered based on partial ordering of matching types $Sim_{\mathcal{T}}$. The most appropriate (or semantically similar) historic conditions is then used to simulate quasi real-time conditions using the diagnoser.

In more details the quasi real-time diagnosis process is based on a traffic congestion diagnoser where quasi real-time conditions (i.e., today's event, road works and weather conditions) are approximated by historic condition using $Sim_{\mathcal{T}}$. This diagnoser is computed at the beginning of the day such that its computation time does not impact quasi real-time diagnosis. During the day, once we then detect a traffic congestion on a road at snapshot s, we use the diagnoser to look up the historic diagnosis information that explains the traffic congestion i.e. to retrieve the transition label that corresponds to the congested road at s. This information might contain conditions that do not happen today but that were only considered because of their $Sim_{\mathcal{T}}$-semantic similarity to historic conditions. In such a case we use semantic techniques for computing a diagnosis report that interprets the traffic congestion in the context of quasi real-time events.

Definition 2 ***(Diagnosis Report)***
Let $\mathcal{L}$ be a DL, $\mathcal{T}$ be a set of axioms in $\mathcal{L}$. Let C_h, C_n, C and $\{r\}$ be four concepts in $\mathcal{L}$ such that C_h and C_n are respectively historical and new conditions. Let (C_h, p_h) be historical conditions of $\{r\}$ to be congested ($\{r\} \sqsubseteq C$) with a probability p_h in $[0, 1]$. A diagnosis report $\langle \mathcal{L}, \mathcal{T}, (C_h, p_h), C_n, \{r\}, C \rangle$ of GCI $\{r\} \sqsubseteq C$ consists in finding a pair $\langle R, p \circ p_h \rangle$ where R is a concept in $\mathcal{L}$ explaining the difference between C_h and C_n, and $\circ$ is an ordering function, which positions the probability p of C_n wrt. p_h.

[4] http://www.w3.org/2004/02/skos/core
[5] http://schemas.talis.com/2005/address/schema
[6] http://www.w3.org/2003/01/geo/wgs84_pos

The diagnosis report $\langle \mathcal{L}, \mathcal{T}, (C_h, p_h), C_n, \{r\}, C \rangle$ is constructed by exploiting concept *abduction* [14] $C_n \backslash C_h$, defined by $\{B \in \mathcal{L} \mid \mathcal{T} \models C_h \sqcap B \sqsubseteq C_n\}$ between new conditions C_n and historic conditions C_h. This description $C_n \backslash C_h$ represents what is underspecified in C_h in order to completely satisfy C_n in $\mathcal{T}$. As the solution of an abduction problem is not unique, we consider the most general description B. Besides retrieving similar conditions for diagnosing anomalies, concept abduction is used to report back the impact of considering non exact matching, so the estimated probability can be justified. The ordering function between p and p_h is defined based on the subsumption relation between C_n and C_h. Computing a diagnosis report is a PTIME problem due to the PTIME complexity of abduction [14] and subsumption [5] in $\mathcal{EL}^{++}$.

Example 6 ***(Diagnosis Report)***
Let $(\{e_1\}, 0.5)$ *in Fig.5 be the diagnosis result (through historical conditions) for a past traffic congestion on road* $\{r_1\}$ *and* $\{e'_1\}$ *a new event such that* $Sim_{\mathcal{T}}(\{e'_1\}, \{e_1\})$ *is PlugIn. In case of a congested road* $\{r_1\}$, $\{e_1\}$ *is provided as diagnosis. In addition* (12) *captures the impact of considering a PlugIn matching between* $\{e'_1\}$ *and* $\{e_1\}$.

$$\langle \{Event \sqcap \exists attendee.LargeType, Event \sqcap \exists attendee.Youth\},\ p \geq 0.5 \rangle \qquad (12)$$

The diagnosis result reflects the real-time condition expect that `LargeType` *audience of the event has been over-generalized during diagnosis (if* $LargeType$ *is defined to be subsumed by* $SmallType$*), and that its characteristic of* $Youth$ *audience has not been considered by* $\{e_1\}$. *Since more people attend event in* $\{e'_1\}$ *than it was the case for* $\{e_1\}$ *we infer* $p \geq 0.5$.

4 Validation

This section reports (i) our context of experimentation, (ii) details of the prototype and (iii) a computational-based evaluation of our approach for testing its performance in real world scenarios. The main objective was to diagnose congested roads (4) using various semantics-augmented real live streams and static data in Table 1.

Table 2. SIRI Data Fragment (Headers are not part of the SIRI data but here for clarity)

timestamp	line reference	direction	journey pattterns reference	in congestion	latitude	longitude	delay	block reference	vehicle	point number	at stop
1322352000000	41	0	041B0002	0	-6.3026	53.4877	-182	41018	33427	3874	1
1322352000000	27	0	271003	0	-6.28932	53.3357	0	27101	33286	2190	0
1322352000000		0	null	0	-6.25923	53.3448	0	4001	33380	7226	0
1322352000000	41	0	410001	0	-6.2375	53.466	0	41013	33631	7348	0
...	...	...	...	...	...	...	...	...	...	...	...

4.1 Context: Dublin City

(**i**) The **Dublin Bus Stream** is encoded according to the SIRI standard (footnote *a* in Table 1), and the real-time stream is persisted into CSV file. Each file represents one day of SIRI data i.e., information of 1000 buses is updated every 20 seconds (Table 2).

Each SIRI record (line in a CSV file) contains information about the current position (latitude and longitude) of a bus. The bus line is uniquely identified by two fields *line reference* and *journey pattern reference*. The boolean fields *direction*, *in congestion*, and *at stop* indicate respectively the bus direction along the bus line, if the bus is in congestion, and if the bus is at a stop point (*point number*). Information about line references, journey pattern references, and stop points is given separately through other CSV file; such information is static (or at least it changes very rarely). We have developed a simple $\mathcal{EL}^{++}$ ontology to represent SIRI data (DL samples in Fig. 2 and 3). Fig. 6 shows the main classes in the ontology. These classes model the core of the static SIRI information: line references, journey pattern references, and point numbers (bus stops). The class *InterStopDistance* is used to provide information about the distance between two point numbers along a journey pattern: following the relationships *fromPointNumber* and *toPointNumber*, it is possible to reconstruct the entire path of a bus line along with the distances between pairs of consecutive point numbers.

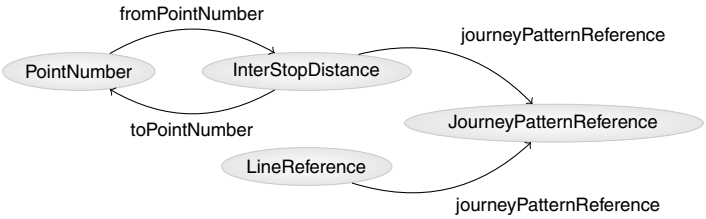


Fig. 6. Core Part of the SIRI Ontology modeling a *VehicleAtomicUpdate* Concept

The actual SIRI records (lines from the CSV files) are modeled as instances of the class *VehicleAtomicUpdate*, which has a property for each field of a SIRI record (Table 2). Note that some of the fields of a SIRI record may lack some field values: then the instance of *VehicleAtomicUpdate* will lack the corresponding properties. Based on an history of 217 SIRI data files (approximately 26 GB), referring to 217 days in 2011 (approximately 122 MB a day), 44.7% of the SIRI records generates 8 triples and 47.2% generate 11 triples (1 triple to define the type of the RDF resource, and 10 triples to specify the SIRI properties); the other records generate either 9 or 10 triples. The varying number of triples per record is due to some missing fields, mostly the line reference and the journey pattern reference. The instances of VehicleAtomicUpdate with missing properties are nevertheless useful to estimate the number of buses in a bounding box in a certain time window (latitude, longitude, timestamp are always available).

(**ii**) **City Events** were captured through *Eventful (http://eventful.com)* and *EventBrite (http://www.eventbrite.com)* where an average of 187 events a day (i.e., same days as

those captured for SIRI data) have been described using some LOD vocabularies e.g., DBpedia, Talis. In addition we enriched the events description with $\mathcal{EL}^{++}$ GCIs to capture fined descriptions of their categories. The latter has been considered for computing not only fined matching between historical and new events but also for computing the diagnosis report (Example 6). Each event has been described on average through 26 RDF triples. In this respect our event description model is not as complete as Event-F [15], but have some extensions for capturing city events (e.g., organizer, venue). The model is also more specific than LODE[7] and the Event ontology[8].

(**iii**) Similarly an average of 51 **Road Works and Maintenance**[9] records a day have also been enriched through 16 RDF triples each. An $\mathcal{EL}^{++}$ enrichment of this raw data ensures that historical and new records can be matched for diagnosis purposes. Again, the way they match is reported for diagnosis inspection.

(**iv**) We also injected $14,316$ $\mathcal{EL}^{++}$ GCIs (through 6 RDF triples each) to describe 4772 **Roads and their Interconnections**[10].

(**v**) The **Core Static Ontology**, which is used for representing SIRI, events, road works, road weather and Dublin weather data, is composed of 67 concepts with 24 role descriptions (25 concepts subsume the 42 remaining ones with a maximal depth of 4).

(**vi**) Finally a **History of** 217 **days of the Traffic Congestion Information** was computed based on stream bus data (encoded by more than 1×10^9 RDF triples) recorded for 217 days. Information about past events, road works, wheather information and road conditions was stored as 1.1×10^6 RDF triples. The traffic congestion diagnoser, consists of $10,856$ transitions and $4,128$ states, $4,076$ of which correspond to road intersections and 52 car parks of event locations. Every diagnoser transition had $4,320$ labels corresponding to all snapshots of a day. Each label contained 0 to 8 causes that with a probability of 0 to 0.74 have caused a traffic congestion at the particular snapshot and road.

4.2 Architecture, Implementation and Prototype

The prototype extends [1] (aiming at displaying traffic conditions in real-time) by providing explanation of road congestions. Congested roads are selectable and information about causes of such situations are displayed and refreshed in quasi real-time. Its implementation consists of (i) a RDFizer which encodes syntactic data in RDF[11], together with elements of Fig.4 i.e., (ii) road congestions detection, (iii) historic diagnosis computation and (iv) quasi real-time diagnosis.

• **On-Demand Stream Data RDFization** of the Dublin bus stream data is exposed as REST APIs (Fig.7). Its RDFization is important not only for capturing road congestions but also to identify potential source of causes. The REST API takes as input two

[7] http://linkedevents.org/ontology/
[8] http://purl.org/NET/c4dm/event.owl#
[9] CSV sample in http://www.dublinked.ie/datastore/metadata064.php
[10] CSV sample in http://www.dublinked.ie/datastore/metadata125.php
[11] We focus on the on-demand transformation of stream SIRI data. Standard approaches - http://www.w3.org/wiki/ConverterToRdf - can been used for RDFizing static CSV, XML data.

timestamps t_i and t_f, and it generates the RDF representation of the SIRI data in the interval $[t_i, t_f]$ i.e., instances of the ontology class *VehicleAtomicUpdate* in Fig.6. Due to large amount of data, we use an *Indexer* that periodically indexes SIRI data to quickly identify the file that contains t_i. Then we perform a binary search in that file to find the line having the closest timestamp to t_i; we start reading from that line and we continue (potentially across multiple files) until we reach t_f. Then we perform a binary search in that file to find the line having the closest timestamp to t_i. While reading the SIRI data for the requested time window, we generate RDF triples describing each record, and store the triples in an RDF store; the current prototype uses Jena TDB [12] as RDF store, but we are currently integrating IBM DB2 RDF store [13]. To avoid problems with possible TDB datasets corruptions, we currently create a new dataset for each SIRI-to-RDF transformation requested: the dataset will contain the static SIRI data in the default graph, and the RDF representation of the requested time window in a named graph. The REST API returns to users a unique identifier for its request, and then generates the RDF in background. The user can check when the requested transformation is completed by providing the unique identifier. We also provide a REST API for querying a dataset, supporting SPARQL SELECT, ASK and CONSTRUCT queries.

• **Road Congestion Detection** is achieved using a DL extension of InfoSphere Streams[14] [1] for real-time detection of road congestions.

• **Historic Diagnosis Computation** is done by elaborating the traffic congestion diagnoser i.e., Dublin city roads (footnote *h* in Table 1) annotated with the diagnosis results explaining historical congestions.

• **Quasi Real-Time Diagnosis:** The diagnoser [9] has been enhanced by reporting the approximation of historical and quasi real-time observations. We have implemented the semantic reasoning part using CEL DL reasoner[15] to check satisfiability, subsumption, and MAMAStng[16] to construct abduction between diagnosis.

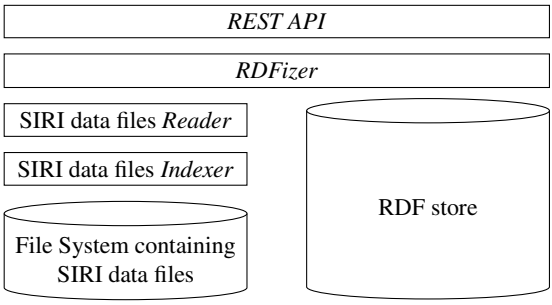


Fig. 7. Stream Data RDFization Architecture

[12] http://jena.apache.org/documentation/tdb/index.html
[13] http://www-01.ibm.com/software/data/db2-warehouse-10/
[14] http://www-01.ibm.com/software/data/infosphere/streams/
[15] http://lat.inf.tu-dresden.de/systems/cel
[16] http://dee-227.poliba.it:8080/MA-MAS-tng/DIG

4.3 Experimentation

We report the computation time of (i) the overall diagnosis approach and then more specifically (ii) the semantic matching part of the diagnosis approach. Experiments were run on a server of 4 Intel(R) Xeon(R) X5650, 2.67GHz cores and 6GB RAM.

• **Overall Approach Experimentation - Context:** Fig.8 illustrates the impact of windows of exploration and size of the ontology streams on the computation time of elements in Section 4.2. The window of exploration of the dynamic knowledge is experimented from a range of 1 min (i.e., 3 snapshots) to 20 min (i.e., 60 snapshots). Besides axioms about bus information, C_1 captures all stream information i.e., road works, events, road weather and weather condition (Section 4.1) while C_2 only captures axioms about bus information and road works i.e., 83% of axioms.

• **Overall Approach Experimentation - Results:** The larger the window size the more computation time is the RDFization of raw data. As an example the RDFization process of a 10 minutes window of a SIRI file (i.e., 30 snapshots described by 9565 lines) was achieved in 6720.4 ms, which gives a processing time of 0.70 ms/line. This transformation generated 97297 RDF triples, which gives an average throughput of 14477.86 triples/sec. We also note that the proportion of computation time vs. detection and diagnosis evolves with respect to the window size. For instance the RDFization process represents 82% of the overall process for a window size of 60 while its represents "only" 63% for a size of 3. The computation time of the detection and diagnosis process represents, on average, respectively 14% and 5% of the overall process. We also note that more heterogeneity in the sources for diagnosis (C_1 vs. C_2) the more time consuming is the RDFization. The quasi real-time aspect of our approach is preserved as 19.5 seconds is required in the worst case (i.e., < 20 seconds update of SIRI data).

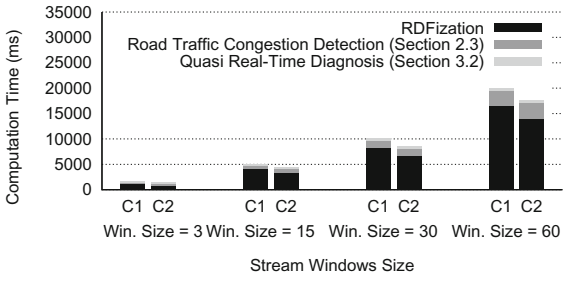


Fig. 8. Computation Time of the Overall Approach

• **Semantic Matching Experimentation - Context:** Fig.9 illustrates the computation time of the different DL reasoning used for computing the quasi real-time diagnosis i.e., DL subsumption (for evaluating Exact, PlugIn, Subsume, Intersection and Disjoint-based comparison of descriptions) and concept abduction (for reporting back the diagnosis approximation). The window of exploration of the dynamic knowledge is experimented from a range of 1 min (i.e., 3 snapshots) to 120 min (i.e., 360 snapshots) over a busy week-end (for maximizing the number of events). The complete context of

Section 4.1 has been considered in this experiment i.e., historical data of 217 days for road works, city events, weather conditions and weather information. Only historical data which fit the same time are considered.

• **Semantic Matching Experimentation - Results:** Our approach guarantees to obtain diagnosis report in suitable range of computation time: from 0.3 to 3.5 seconds for a window of exploration of respectively 1 and 120 minutes. Subsumption is the more time consuming reasoning since a large number of subsumption tests have to be performed between various real-time and historical events, road works, road conditions, weather information. Fig.9 shows a steep growth, mainly due to the increasing number of potential matching tests that required to be evaluated when extending the window size. Its computation becomes really problematic if the diagnosis is estimated on a larger windows e.g., 9.6 seconds is required for a window of 540 snapshots. However diagnosis used to be computed through an analysis up to 90 snapshots. The abduction computation does not vary significantly (on average 0.1 second) mainly because only one diagnosis report is elaborated, independently of the size of the stream window.

4.4 Lessons Learned

During the transformation of raw data into semantic description, we were facing the challenge of discovering the appropriate vocabulary, with the appropriate expressivity. We mainly used LOD vocabularies for linkage and integration with external source of data. However, some cases (i.e., Dublin Bus, events and road works data) required more specific and fine-grained descriptions with higher expressivity for matching and reasoning purposes. Towards this issue we carefully developed our own terminologies, aligned with the schema of raw data, for reusability and reasoning.

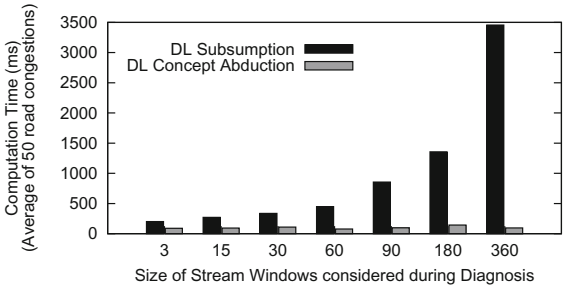


Fig. 9. Computation Time for Semantic Matching

The on-the-fly integration/linkage of new sources of causes (Table 1), exposed as raw data, with our existing RDFized data is not straightforward even with existing tools. Most relevant linkages had to be done manually during the first integration e.g., any road work is a sort of event in our context but none of existing tool has infer such a link.

We enriched exiting data sets from Dublin City with OWL $\mathcal{EL}^{++}$ based description not only for computing matching between events, road works (among others), but also to report the impact of approximating diagnosis conditions (through concept abduction). It is obvious that the computation performance of our approach would have

been strongly altered when considering much more expressive semantics such as OWL 2 Full or DL. Indeed the quasi real-time dimension cannot be met in OWL 2 Full or DL due to the complexity of subsumption and abduction. However considering OWL 2 Full or DL could have triggered more causes for road congestions, and improved the diagnosis precision. It would be also interesting to evaluate the impact of using a subset of $\mathcal{EL}^{++}$ on the computation performance and the diagnosis results. In other words, which *expressiveness does fit the better for this application* is still an open challenge. Further experiments are required to provide the most appropriate context and trade-off complexity/expressivity.

The transformation of SIRI data to RDF is a crucial part and the most time consuming part of the overall approach (Fig.8). Using pure Jena classes[17] to add triples requires 15 minutes to RDFize a 90 minutes window of SIRI data. In order to meet our quasi real-time constraint our platform provides a customized Java *InputStream* of RDF triples, that are generated in a buffered way by reading one or more CSV files according to the requested time window. We feed our customized *InputStream* to the class *TDBLoader*, which allows bulk loading into a TDB dataset. This approach requires less than 60 seconds to RDFize a 90 minutes window of SIRI data

5 Related Work

There are many approaches in traffic controls where domain experts are in charge of understanding effects of specific and targeted events on road conditions in order to take corrective actions. However the automated and real-time explanation of traffic congestions has not been tackled, making road traffic difficult to be efficiently managed.

Diagnosis, or the process of identifying the nature and cause of an anomaly (aka. conflict) has been largely studied by the Semantic Web community, but mainly in the context of an ontology. Existing works [16] applied and extended axioms pinpointing to derive how changes in an ontology may result in conflicts in the knowledge base. Following [17], the task of *diagnosis* consists in retrieving the causes of an anomaly (e.g., road congestions) by interpreting external sources of causes. We extended it by considering historical information to capture potential diagnosis. We make use of existing semantic matching techniques to approximate diagnosis in an open-world scenario i.e., new sources of causes of anomalies (e.g., road works). In addition concept abduction is considered to report back the information we gained or lost during diagnosis.

There are no existing approaches that integrate semantics and diagnosis techniques. However, its integration is needed since existing approaches cannot handle new events and observations as we consider in this work. They all assume a closed world scenario where the set of possible causes that could explain the effects is well defined and where cause-effect relationships can (at least with unlimited computational resources) be established. The closest diagnosis works to our approach are the ones that tackle the complexity problem of diagnosis approaches [18] by precomputing diagnosis results for *some* anomalies. If other anomalies are detected some machine learning methods are used to estimate the diagnosis result in these cases [19]. However, this estimation consists only of a numeric value rather than an expressive (semantic) explanation as in

[17] http://jena.apache.org/

our case. Furthermore these approaches consider only the problem of mapping anomalies to well defined sets of possible causes rather than to new causes as in our case.

Semantic web technologies and machine learning techniques have been coupled by [20] for (i) road traffic prediction and (ii) trip planning in Milano City. Our work goes further by explaining traffic congestions by revisiting AI diagnosis. Our work required a higher level of expressivity for interpreting diagnosis results in open-world scenarios.

[21] present a framework for publishing, RDFizing and linking transport data on the Web. Contrary to our work, they do not consider the stream dimension of transportation data, and no quasi real-time RDFization is presented. We targeted different applications where ours required more expressive and specific ontologies.

6 Conclusion and Future Work

Diagnosis, *or the method to connect causes to its effects*, is an important reasoning task for obtaining insight on cities, its road traffic and reaching the concept of sustainable and smarter cities that is envisioned nowadays. This work focused on diagnosing road traffic congestions in the real-world context of Dublin City where static and stream data of its road traffic domain has been exploited. Our approach coupled pure AI diagnosis approaches with semantic web technologies for accurate and quasi real-time diagnosing in an open-world context of heterogeneous and large data. The approach has shown high performance and applicability in the context of real and live data from Dublin City. In addition we raised some challenges we met during the implementation of the prototype.

We currently study the integration of our approach with IBM DB2 RDF and expect to serve real-time semantic streams by using IBM InfoSphere Streams. In future work, we will further evaluate the impact of the number of other data sources (e.g., real-time CCTV monitoring of Dublin City) on precision and scalability of the diagnosis approach. We also expect using citizen sensing (e.g., twitter traffic data) to validate diagnosis results. Finally, we will work on a model for predicting road traffic congestions by coupling semantic web technologies and machine learning approaches.

References

1. Biem, A., Bouillet, E., Feng, H., Ranganathan, A., Riabov, A., Verscheure, O., Koutsopoulos, H.N., Moran, C.: Ibm infosphere streams for scalable, real-time, intelligent transportation services. In: SIGMOD, pp. 1093–1104 (2010)
2. Luo, C., Thakkar, H., Wang, H., Zaniolo, C.: A native extension of sql for mining data streams. In: SIGMOD Conference, 873–875 (2005)
3. Babu, S., Widom, J.: Continuous queries over data streams. SIGMOD Record 30(3), 109–120 (2001)
4. Haase, C., Lutz, C.: Complexity of subsumption in the [escr][lscr] family of description logics: Acyclic and cyclic tboxes. In: ECAI, pp. 25–29 (2008)
5. Baader, F., Brandt, S., Lutz, C.: Pushing the el envelope. In: IJCAI, pp. 364–369 (2005)
6. Horrocks, I., Sattler, U.: Ontology reasoning in the shoq(d) description logic. In: IJCAI, pp. 199–204 (2001)
7. Ren, Y., Pan, J.Z.: Optimising ontology stream reasoning with truth maintenance system. In: CIKM, pp. 831–836 (2011)

8. Huang, Z., Stuckenschmidt, H.: Reasoning with Multi-version Ontologies: A Temporal Logic Approach. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 398–412. Springer, Heidelberg (2005)
9. Sampath, M., Sengupta, R., Lafortune, S., Sinnamohideen, K., Teneketzis, D.: Failure diagnosis using discrete event models. IEEE Trans. on Cont. Sys. Tech. 4(2), 105–124 (1996)
10. Sinnott, R.W.: Virtues of the haversine. Sky and Telescope 68(2), 159–197 (1984)
11. Schumann, A., Pencolé, Y., Thiébaux, S.: Symbolic models for diagnosing discrete-event systems. In: ECAI, pp. 1085–1086 (2004)
12. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic Matching of Web Services Capabilities. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 333–347. Springer, Heidelberg (2002)
13. Li, L., Horrocks, I.: A software framework for matchmaking based on semantic web technology. In: WWW, pp. 331–339 (2003)
14. Noia, T.D., Sciascio, E.D., Donini, F.M., Mongiello, M.: Abductive matchmaking using DLs. In: IJCAI, pp. 337–342 (2003)
15. Scherp, A., Franz, T., Saathoff, C., Staab, S.: F–a model of events based on the foundational ontology dolce+dns ultralight. In: K-CAP, pp. 137–144 (2009)
16. Parsia, B., Sirin, E., Kalyanpur, A.: Debugging owl ontologies. In: WWW, pp. 633–640 (2005)
17. Lécué, F.: Diagnosing changes in an ontology stream: A dl reasoning approach. In: AAAI (2012)
18. de Kleer, J., Mackworth, A.K., Reiter, R.: Characterizing diagnoses and systems. Artificial Intelligence 56(2-3), 197–222 (1992)
19. Keren, B., Kalech, M., Rokach, L.: Model-based diagnosis with multi-label classification. In: 22nd International Workshop on Principles of Diagnosis, DX 2011 (2011)
20. Valle, E.D., Celino, I., Dell'Aglio, D., Grothmann, R., Steinke, F., Tresp, V.: Semantic traffic-aware routing using the larkc platform. IEEE Internet Computing 15(6), 15–23 (2011)
21. Plu, J., Scharffe, F.: Publishing and linking transport data on the web. International Workshop On Open Data abs/1205.1645 (2012)

DEQA: Deep Web Extraction for Question Answering*

Jens Lehmann[1,2], Tim Furche[1], Giovanni Grasso[1], Axel-Cyrille Ngonga Ngomo[2], Christian Schallhart[1], Andrew Sellers[1], Christina Unger[3], Lorenz Bühmann[2], Daniel Gerber[2], Konrad Höffner[2], David Liu[1], and Sören Auer[2]

[1] Department of Computer Science, Oxford University, Wolfson Building, Parks Road, Oxford OX1 3QD
firstname.lastname@cs.ox.ac.uk
[2] Institute of Computer Science, University of Leipzig, 04103 Leipzig
lastname@informatik.uni-leipzig.de
[3] Bielefeld University, CITEC, Universitätsstraße 21–23, 33615 Bielefeld
cunger@cit-ec.uni-bielefeld.de

Abstract. Despite decades of effort, intelligent object search remains elusive. Neither search engine nor semantic web technologies alone have managed to provide usable systems for simple questions such as "find me a flat with a garden and more than two bedrooms near a supermarket."

We introduce DEQA, a conceptual framework that achieves this elusive goal through combining state-of-the-art semantic technologies with effective data extraction. To that end, we apply DEQA to the UK real estate domain and show that it can answer a significant percentage of such questions correctly. DEQA achieves this by mapping natural language questions to SPARQL patterns. These patterns are then evaluated on an RDF database of current real estate offers. The offers are obtained using OXPath, a state-of-the-art data extraction system, on the major agencies in the Oxford area and linked through LIMES to background knowledge such as the location of supermarkets.

1 Introduction

Answering questions such as "find me a flat to rent close to Oxford University with a garden" is one of the challenges that has haunted the semantic web vision since its inception [3]. Question answering has also been the holy grail of search engines, as recently illustrated by both Google and Bing touting "structured data" search and "query answering". Though both of these efforts have made

* The research leading to these results has received funding under the European Commission's Seventh Framework Programme (FP7/2007–2013) from ERC grant agreement DIADEM, no. 246858, IP grant agreement LOD2, no. 257943 and Eurostars E!4604 SCMS.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 131–147, 2012.

great strides in answering questions about general, factual knowledge, they have fallen short for more transient information such as real estate, tickets, or other product offerings. Vertical search engines and aggregators also fail to address such questions, mostly due to a lack of natural language understanding and limited background knowledge.

This is true even though data extraction and semantic technologies aim to address this challenge from quite opposite directions: On the one hand, the aim of web extraction is to obtain structured knowledge by analyzing web pages. This does not require publishers to make any changes to existing websites, but requires re-engineering the original data used to generate a website. On the other hand, semantic technologies establish the means for publishers to directly provide and process structured information, avoiding errors in extracting ambiguously presented data, but placing a considerable burden on publishers. Despite this chasm in how they approach question answering, neither has succeeded in producing successful solutions for transient, "deep" web data (in contrast to general, Wikipedia-like knowledge and web sites).

In this paper, we show that in this very dichotomy lies the solution to addressing deep web question answering: We present DEQA, a system that allows the easy combination of semantic technologies, data extraction, and natural language processing and demonstrate its ability to answer questions on Oxford's real estate market. The data is extracted from the majority of Oxford's real estate agencies, despite the fact that none publishes semantic (or other structured) representations of their data, and combined with background knowledge, e.g., to correlate real estate offers with points of interest such as the "Ashmolean Museum" or close-by supermarkets.

DEQA is the first comprehensive framework for *deep web question answering* approaching the problem as a combination of three research areas: **(1)** *Web data extraction* – to obtain offers from real estate websites, where no structured interface for the data is available (which happens to be the case for all Oxford real estate agencies). **(2)** *Data integration* – to interlink the extracted data with background knowledge, such as geo-spatial information on relevant points of interest. **(3)** *Question answering* – to supply the user with a natural language interface, capable of understanding even complex queries. For example a query like "find me a flat to rent close to Oxford University with a garden" can be answered by DEQA. However, this cannot be achieved without adaptation to the specific domain. The unique strength of DEQA is that it is based not only on best-of-breed data extraction, linking, and question answering technology, but also comes with a clear methodology specifying how to adapt DEQA to a specific domain. In Section 3, we discuss in detail what is required to adapt DEQA to a new domain and how much effort that is likely to be.

DEQA *Components.* We developed DEQA as a conceptual framework for enhancing classic information retrieval and search techniques using recent advances in three technologies for the above problems, developed by the three groups involved in DEQA: DIADEM at Oxford, AKSW at Leipzig, and CITEC at Bielefeld.

(1) OXPath is a light-weight data extraction system particularly tailored to quick wrapper generation on modern, scripted web sites. As demonstrated in [9], OXPath is able to solve most data extraction tasks with just four extensions to XPATH, the W3C's standard query language for HTML or XML data. Furthermore, through a sophisticated garbage collection algorithm combined with tight control of the language complexity, OXPath wrappers outperforms existing data extraction systems by a wide margin [9]. For the purpose of integration into DEQA, we extended OXPath with the ability to direct extract RDF data, including type information for both entities and relations.

(2) The real estate offers extracted with OXPath contain no or little contextual knowledge, e.g., about general interest locations or typical ranges for the extracted attributes. To that end, we link these extracted offers with external knowledge. This is essential to answer common-sense parts of queries such as "close to Oxford University". Specifically, we employ the LIMES [23,22] framework, which implements time-efficient algorithms for the discovery of domain-specific links to external knowledge bases such as DBpedia [1].

(3) To apply question answering in a straightforward fashion on the supplied, extracted, and enriched knowledge, we employ the *TBSL* approach [28] for translating natural language questions into SPARQL queries. TBSL disambiguates entities in the queries and then maps them to templates which capture the semantic structure of the natural language question. This enables the understanding of even complex natural language containing, e.g., quantifiers such as the most and more than, comparatives such as higher than and superlatives like the highest – in contrast to most other question answering systems that map natural language input to purely triple-based representations.

Using the combination of these three technologies allows us to adjust to a new domain in a short amount of time (see Section 3), yet to answer a significant percentage of questions about real estate offers asked by users (see Section 4).

Contributions. These results validate our hypothesis, that the combination of these technologies can (and may be necessary to) yield accurate question answering for a broad set of queries in a specific domain. This is achieved without requiring publishers to provide structured data and at a fairly low effort for domain adaptation. Specifically,

(1) DEQA is the first comprehensive *deep web question answering system for entire domains* that can answer the majority of natural language questions about objects only available in form of plain, old HTML websites (Section 2).
(2) These websites are turned into structured RDF data through an *extension of OXPath for RDF output,* providing a concise syntax to extract object and data properties (Section 2.1).
(3) By extracting this data into RDF and *linking* it with background knowledge, it can answer not only queries for specific attributes ("in postcode OX1"), but also queries using *common-sense criteria* ("close to Oxford University"), see Section 2.2.
(4) With TBSL, we are able to map such queries to SPARQL expressions even if they include *complex natural language expressions* such as "higher than".

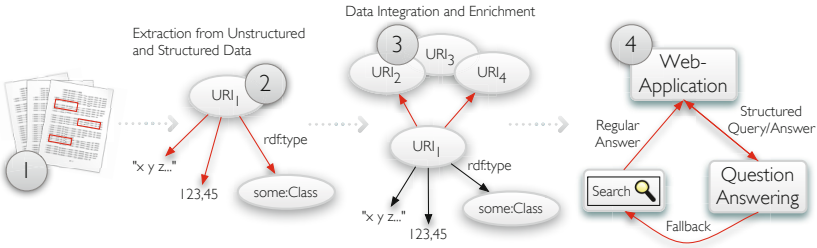


Fig. 1. Overview of the DEQA conceptual framework.

(5) DEQA provides a *methodology and framework* that can be rapidly instantiated for new domains, as discussed in Section 3.
(6) As a *case study*, we instantiate DEQA to Oxford's entire real estate market, involving the 20 largest real estate agents and all of their properties on sale, and illustrate the necessary effort.
(7) A user-centric *evaluation* demonstrates that DEQA is able to answer many of the natural language questions asked by users (Section 4).

With these contributions, DEQA is the first comprehensive framework for deep web query answering, covering the extraction and data collection process as well as the actual query answering, as elaborated in Section 5.

2 Approach

The overall approach of DEQA is illustrated in Figure 1: Given a particular domain, such as real estate, the first step consists of identifying relevant websites and extracting data from those. This previously tedious task can now be reduced to the rapid creation of OXPath wrappers as described in Section 2.1. In DEQA, data integration is performed through a triple store using a common base ontology. Hence, the first phase may be a combination of the extraction of unstructured and structured data. For instance, websites may already expose data as *RDFa*, which can then be transformed to the target schema, e.g. using *R2R* [4], if necessary. This basic RDF data is enriched, e.g. via linking, schema enrichment [16,6], geo-coding or post-processing steps on the extracted data. This is particularly interesting, since the LOD cloud contains a wealth of information across different domains which allows users to formulate queries in a more natural way (e.g., using landmarks rather than postcodes or coordinates). For instance, in our analysis of the real estate domain, over $100k$ triples for $2,400$ properties were extracted and enriched by over $100k$ links to the LOD cloud. Finally, question answering or semantic search systems can be deployed on top of the created knowledge. One of the most promising research areas in question answering in the past years is the conversion of natural language to SPARQL queries [28,18,27], which allows a direct deployment of such systems on top of a triple store. Finally, DEQA first attempts to convert a natural language query to SPARQL, yet can fall back to standard information retrieval, where this fails.

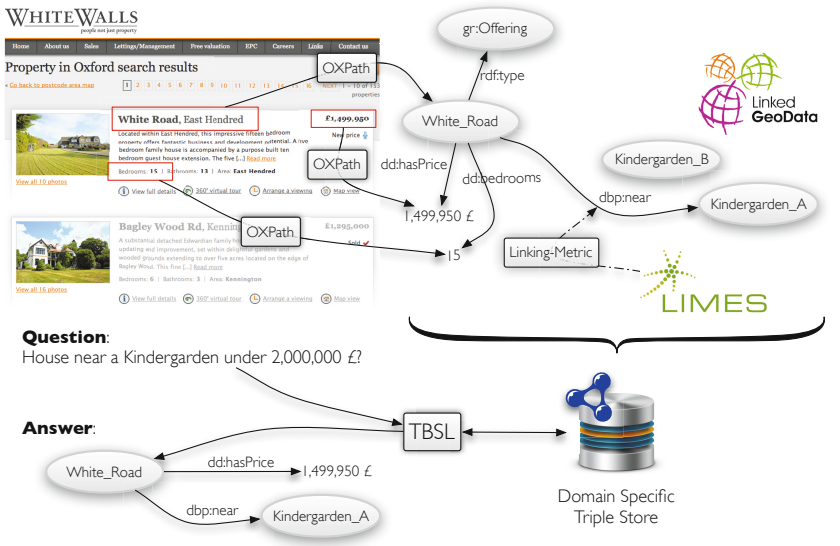


Fig. 2. Implementation of DEQA for the real-estate domain

The domain-specific implementation of the conceptual framework, which we used for the real estate domain, is depicted in Figure 2. It covers the above described steps by employing state-of-the-art tools in the respective areas, OXPath for data extraction to RDF, LIMES for linking to the linked data cloud, and TBSL for translating natural language questions to SPARQL queries. In the following, we briefly discuss how each of these challenges are addressed in DEQA.

2.1 OXPath for RDF Extraction

OXPath is a recently introduced [9] modern wrapper language that combines ease-of-use (through a very small extension of standard XPATH and a suite of visual tools [15]) with highly efficient data extraction. Here, we illustrate OXPath through a sample wrapper shown in Figure 3.

This wrapper directly produces RDF triples, for which we extended OXPath with *RDF extraction markers* that generate both data and object properties including proper type information and object identities. For example the extraction markers `<:(gr:Offering>` and `<gr:includes(dd:House)>` in Figure 3 produce – given a suitable page – a set of matches typed as `gr:Offering`, each with a set of `dd:House` children. When this expression is evaluated for RDF output, each pair of such matches generates two RDF instances related by `gr:includes` and typed as above (i.e., three RDF triples).

To give a more detailed impression of an OXPath RDF wrapper assuming some familiarity with XPATH, we discuss the main features of Figure 3:

(Line 1) We first load the web site `wwagency.co.uk`, a real estate agency serving the Oxford area, and click on their search button without restricting the

```
doc("http://wwagency.co.uk/")//input[@name='search']/{click/}/
(descendant::a[@class='pagenum']
    [text()='NEXT'][1]/{click[wait=1]/})*
 /descendant::div.proplist_wrap:<(gr:Offering)>
 [?.//span.prop_price:<dd:hasPrice(xsd:double)=
   substring-after(.,'£')>]
 [?.//a[@class='link_fulldetails']:<foaf:page=string(@href)>]
 [?.:<gr:includes(dd:House)>
   [?.//h2:<gr:name=string(.)>]
   [?.//h2:<vcard:street_address=string(.)>]
   [?.//div.prop_maininfo//strong[1]:<dd:bedrooms=string(.)>]
   [? .//img:<foaf:depiction=string(@src)>]
```

Fig. 3. OXPath RDF wrapper

search results. Therein, `{click/}` is an *action* which clicks on all elements in the current context set, in this case, containing only the search button. This action is *absolute,* i.e., after executing the action, OXPath continues its evaluation at the root of the newly loaded document.

(Lines 2–3) Next, we iterate through the next links connecting the paginated results. To this end, we repeat within a *Kleene star* expression the following steps: we select the first link which is of class `'pagenum'` and contains the text `'NEXT'`. The expression then clicks on the link in an absolute action `{click[wait=1]/}` and waits for a second after the `onload` event to ensure that the heavily scripted page finishes its initialization.

(Line 4) On each result page, we select all `div` nodes of class `proplist_wrap` and extract an `gr:Offering` instance for each such node. Aside from the CSS-like shorthand for classes (analogously, we provide the `#` notation for ids), this subexpression uses the first *RDF extraction marker:* This extraction marker `:<gr:Offering>` produces an *object instance,* because it does not extract a value necessary to produce a *data property.* The remainder of the expression adds object and data properties to this instance, detailing the offering specifics.

(Lines 5–6) We extract the price of the offering by selecting and extracting the `span` of class `prop_price` within the offering `div`. In particular, the marker `:<dd:hasPrice(xsd:double)=substring-after(.,'£')>` specifies the extraction of a `dd:hasPrice` data property of type `xsd:double` with the value stated after the `'£'` character. The nesting of RDF properties follows the predicate nesting structure, and thus, as the price is extracted inside a predicate following the extracted offering, this price is associated with the offering. We use an *optional predicate,* `[?`ϕ`]`, to ensure that the evaluation continues, even if an offering does not name a price and the predicate extraction fails.

(Line 7) Links to details pages are extracted as `foaf:page` data properties.

(Lines 8–12) Aside having a price, an offering also needs to refer to a property, extracted next. In Line 8, with `:<gr:includes(dd:House)>`, we extract an instance of the `dd:House` class as *object property* of the previous offering (because of the predicate nesting), related via `gr:include`. The remaining four lines extract the name,

address, the number of bedrooms, and the property images as data properties belonging to the `dd:House` instance, as all those properties are extracted within nested predicates.

This wrapper produces RDF triples as below, describing two instances, the first one `dd:g31g111` representing a house with 4 bedrooms in Marston, and the second one `dd:g31g109` representing an offer on this house at GBP 475000.

```
dd:g31g111
    a dd:House ;            dd:bedrooms 4 ;
    gr:name "William Street,  Marston OX3" ;
    vcard:street-address "William Street,  Marston OX3" ;
    foaf:depiction "http://www.wwagency.com/i_up/111_1299510028.jpg" .
dd:g31g109
    a gr:offering ;       dd:hasPrice "475000"^^xsd:double ;
    gr:includes dd:g31g111 .
```

For more details on OXPath, please refer to [9]. We also provide the full set of wrappers on the project home page.

2.2 LIMES

We discuss the LIMES specification used to link and integrate the RDF data extracted by OXPath with *LinkedGeoData* – a vast knowledge base extracted from OpenStreetMaps containing spatial data including points-of-interest such as schools. The following listing shows an excerpt of the specification that links houses extracted by OXPath with nearby schools. Every link discovery process requires a set S of source and T target instances that are to be linked. In LIMES, these can be defined by specifying the *restrictions* on the instances as well as the set of *properties* that these instances must possess to be linked. In our example, the set S (specified by the tag `<SOURCE>`) consists of `oxford:House` which possess a longitude and a latitude. Similarly, the set T (which is omitted in the listing for brevity) was defined as all the schools whose latitude lies between 50 and 52 degrees and whose longitude lies between -2 and -1 degrees. For instances $a \in S$ and $b \in T$, the similarity is set to

$$\frac{1}{1 + \sqrt{(a.\texttt{wgs:lat} - b.\texttt{wgs:lat})^2 + (a.\texttt{wgs:long} - b.\texttt{wgs:long})^2)}}. \quad (1)$$

Two instances are then considered close to each other (described by the predicate `dbp:near`) if their similarity was at least 0.95.

```
<SOURCE> <ID>oxford</ID>
...
        <VAR>?a</VAR>
        <RESTRICTION>?a a oxford:House</RESTRICTION>
        <PROPERTY>wgs:lat AS number</PROPERTY>
        <PROPERTY>wgs:long AS number</PROPERTY> </SOURCE>
...
<METRIC>euclidean(a.wgs:lat|wgs:long, b.wgs:lat|wgs:long)</METRIC>
```

```
  <ACCEPTANCE> <THRESHOLD>0.9975</THRESHOLD>
10             <FILE>allNear.ttl</FILE>
               <RELATION>dbp:near</RELATION> </ACCEPTANCE>
12 ...
```

The property values of all schools from LinkedGeoData that were found to be close to houses extracted by OXPath were subsequently retrieved by LIMES and loaded into the DEQA triple store.

2.3 TBSL Question Answering

Figure 4 gives an overview of our template-based question answering approach TBSL [28]. The system takes a natural language question as input and returns a SPARQL query and the corresponding answer(s) as output. First, the natural language question is parsed on the basis of its part-of-speech tags and a domain-independent grammar comprising for example wh-words, determiners, and numerals. The result is a semantic representation of the natural language query, which is then converted into a SPARQL query template. This template fixes the overall structure of the target query, including aggregation functions such as filters and counts, but leaves open slots that still need to be filled with URIs corresponding to the natural language expressions in the input question. For example, the question "Give me all flats near Oxford University" yields the following template query, which contains a class slot for some URI corresponding to "flats", a resource slot for some URI corresponding to "Oxford University", and a property slot that expresses the "near" relation:

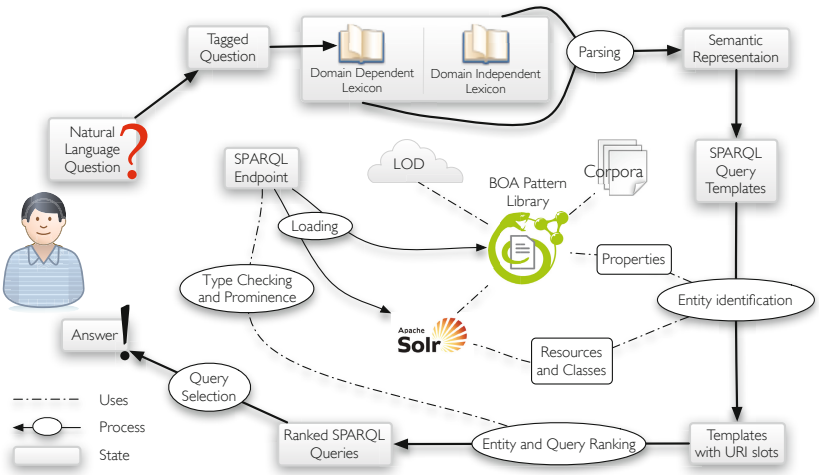


Fig. 4. Overview of the TBSL question answering engine (source: [28])

```
SELECT ?y WHERE {
     ?y ?p1 ?y0.
     ?y rdf:type ?p0.
}
```

– `y0`: "Oxford University" (resource)
– `p0`: "flats" (class)
– `p1`: "near" (object property)

In order to fill these slots, entity identification approaches are used to obtain appropriate URIs, relying both on string similarity and natural language patterns compiled from existing structured data in the Linked Data cloud and text documents (cf. [10]). This yields a range of query candidates as potential translations of the input question. Those candidates are ranked on the basis of string similarity values, prominence values, and schema conformance checks. The highest ranked queries are then tested against the underlying triple store and the best answer is returned to the user.

3 Domain Adaption Costs

DEQA requires instantiation for a specific domain, however, through advances in semantic and web extraction technologies this adaptation involves far less efforts than in the past and is now feasible even with limited resources. We substantiate this claim by discussing the resources required for our case study on Oxford's real estate for **(1)** system setup and domain adaptation and for and **(2)** maintaining the wrappers and links to background knowledge.

The first step in adapting DEQA to a new domain is the *creation or adaption of a suitable domain ontology* in RDFS. In our case, the ontology consists of 5 object properties, 7 data properties, 9 classes, and 10 individuals, all specified in less than 150 lines of turtle code. We were cautious to capture all relevant cases. Hence we build the ontology iteratively while fitting a dozen representative offers from 4 different agencies into the ontology – reaching already a saturation. The entire process of ontology creation took four domain experts a couple of hours.

Web Extraction. Having the ontology, we need to develop wrappers to extract the data from the relevant sites. The process consists of identifying the relevant DOM features to frame the data to be extracted, and running sufficiently many tests to check the wrapper's behavior on other pages from the same site. The wrappers we employ in our case study took on average 10 minutes each to create, such that it took an OXPath expert less than a day to identify the 20 most relevant web sites and write appropriate wrappers. To ease OXPath wrapper generation, we relied on Visual OXPath [15], a supervised wrapper induction system that generates highly robust wrappers from few examples: The system embeds a real browser, and records user interaction on the page (e.g., navigation, click, form filling). Once, the relevant data has been reached, the user marks the data to extract (e.g., price), and checks whether Visual OXPath generalizes the selection correctly, in case refining the selection. In our user study [15], we show

that even users without prior knowledge of OXPath can create a wrapper in less than three minutes (not counting testing and verification) on average.

Linking. Creating LIMES link specifications can be carried out in manifold ways. For example, LIMES provides active learning algorithms for the semi-automatic detection of link specifications that have been shown to require only a small number of annotations (i.e., $10 - 40$ depending on the data quality) to detect high-quality link specifications [24,21]. Given that we had clear definition of the two predicates `near` (for distances up to 2km) and `atWalkingDistance` (for distances up to 500m) to be computed for the domain at hand, we chose to create link specifications manually for each of these predicates.

Question Answering. The component for parsing a user question and constructing query templates requires only little domain adaptation. The core part of the lexicon that is used for parsing comprises domain-independent expressions that can be re-used, all other entries are built on the fly. The only part that was added for DEQA were lexical entries covering some typical tokens with fixed mappings to URIs in the given domain, e.g. "near". This has been done for six mappings, resulting in 20 domain-specific entries. The required manual effort amounts to less than an hour.

System Maintenance

The frequency to which a wrapper needs to be updated is directly correlated to its robustness. Robustness measures the degree of a wrapper to still select the same nodes after changes on the page. Both [15,12] show that wrappers without robustness consideration have limited lifespan, but Visual OXPath implements a number of techniques to prolong the fitness of its wrappers. In particular, given only a single example, Visual OXPath suggests a list of expressions ranked by robustness of the generated wrapper. We have evaluated the top-ranked suggested wrappers over 26 weeks, showing that they fail only after 26 weeks in contrast to average wrappers that fail in $9 - 12$ weeks. In Oxford real estate, we estimate that wrapper maintenance will involve about one failing wrapper per week.

Linking and Question Answering. The system maintenance for the LIMES link specifications is minimal. If the schema is not altered, the specifications created can simply be reran when the data endpoints are updated. In case of an alteration of the schema, the `PROPERTY` and `METRIC` tags of the specification need to be altered. This is yet a matter of minutes if the schema of both endpoints is known. If the new schema is not known, then the link specification has to be learned anew. Previous work [21] has shown that even on large data sets, learning such a specification requires only about 5 min. For question answering, no regular maintenance effort is usually required. An exception, both for linking and question answering, are schema changes. Such changes can in rare cases invalidate specifications, in which case they have to be altered manually. TBSL is flexible in terms of schema changes as long as entities use appropriate labels or URIs. For instance, in [28] was applied to the DBpedia ontology with hundreds of classes and properties without requiring manual configuration for adapting it to

Table 1. Evaluation results and failures

(a) Evaluation results

number of questions	100
—SPARQL queries created	71
—SPARQL queries returning results	63
—SPARQL queries with correct results	49
—exactly intended SPARQL query	30
—SPARQL queries with incorrect results	14

(b) Failure reasons

failures	
—data coverage	9
—linguistic coverage	18
—POS tagging	2
—other reasons	6

this schema. However, previously manually added domain-specific configuration entries for improving the performance of TBSL may require updates in case of schema changes.

4 Evaluation

The components comprising the DEQA platform have been evaluated in the respective reference articles, in particular [9] for OXPath, [23] for LIMES, and [28] for TBSL. Hence, we are mostly interested in an evaluation of the overall system, as well as specific observation for the Oxford real estate case study. The main benefit of DEQA is to enhance existing search functionality with question answering. Therefore, we evaluate the overall system for the real-estate domain by letting users ask queries and then verifying the results.

First DEQA was instantiated for Oxford real-estate as described in Section 3. The OXPath wrappers, the LIMES specs and the TBSL configuration are all publicly available at `http://aksw.org/projects/DEQA`. Our dataset consists of more than 2400 offers on houses in Oxfordshire, extracted from the 20 most popular real estate agencies in the area. The wrappers extract spatial information from 50% of the agencies, typically extracted from map links. For all properties in our dataset, we extract street address and locality. The full postcode (e.g., OX27PS) is available in 20% of the cases (otherwise only the postcode area, e.g., OX1 for Oxford central is available). 96% of all offers expose directly the price, the remaining 4% are "price on inquiry". Images and textual descriptions are available for all properties, but not all agencies publish the number of bathrooms, bedrooms and reception rooms. These offers are enriched by LIMES with 93,500 links to near (within 2 kilometres) and 7,500 links to very near (within 500 metres) spatial objects. The data is also enriched by loading 52,500 triples from LinkedGeoData describing the linked objects. Domain specific spatial mappings were added to TBSL, e.g. "walking distance" is mapped to "very near".

We asked 5 Oxford residents to provide 20 questions each. They were told to enter questions, which would typically arise when searching for a new flat or house in Oxford. We then checked, whether the questions could be parsed by TBSL, whether they could successfully be converted to a SPARQL query on the underlying data and whether those SPARQL queries are correct.

4.1 Results and Discussion

It turned out that most questions would be impossible to answer by only employing information retrieval on the descriptions of properties in Oxford. Many questions would also not be possible to answer via search forms on the respective real-estate websites, as they only provide basic attributes (price, bedroom numbers), but neither more advanced ones (such as "Edwardian", with garden) nor have a concept of close-by information (such as close to a supermarket). Even if they can be answered there, the coverage would be low as we extracted data using over 20 wrappers. While some questions had similar structures, there is little overlap in general.

The results of our experiment are shown in Tables 1a and 1b. Most questions can be converted successfully to SPARQL queries and many of those are the SPARQL queries intended by users of the system. Hence, DEQA provides significant added value in the real estate domain in Oxford despite the relatively small effort necessary for setting up the system. For the questions, which were not correctly answered, we analysed the reasons for failure and summarise them in Table 1b. If questions were not correctly phrased, such as "house with immediately available", they lead to part-of-speech tagging problems and parse failure. Such issues will be dealt with by integration query cleaning approaches into DEQA. In some cases TBSL could not answer the question because it lacks certain features, e.g. negation such as "not in Marston" or aggregates such as average prices in some area. But since TBSL uses a first order logical representation of the input query internally, those features can be added to the QA engine in the future. Support for some aggregates such as COUNT already exists. In some cases, on the other hand, data was insufficient, e.g. users asking for data that was neither extracted by OXPath nor available through the links to LinkedGeoData, e.g. "house in a corner or end-of-terrace plot". Moreover, some questions contain vague, subjective criteria such as "cheap", "recently" or even "representative", the exact meaning of which heavily depends on the user's reference values. In principle, such predicates could be incorporated in TBSL by mapping them to specific restrictions, e.g. cheap could be mapped to costs for flats less than 800 pounds per month. The extended version of DEQA will be compared with classical retrieval engines to quantify the added value of our approach.

An example of a successful query is "all houses in Abingdon with more than 2 bedrooms":

```
SELECT ?y WHERE {
  ?y a <http://diadem.cs.ox.ac.uk/ontologies/real-estate#House> .
  ?y <http://diadem.cs.ox.ac.uk/ontologies/real-estate#bedrooms> ?y0 .
  ?y <http://www.w3.org/2006/vcard/ns#street-address> ?y1 .
  FILTER(?y0 > 2) .
  FILTER(regex(?y1,'Abingdon','i')) .
}
```

In that case, TBSL first performs a restriction by class ("House"), then it finds the town name "Abingdon" from the street address and it performs a filter on

the number of rooms. Note that many QA systems over structered data rely on purely triple-based representations (e.g. PowerAqua [19]) and therefore fail to include such filters.

Another example is "Edwardian houses close to supermarket for less than 1,000,000 in Oxfordshire", which was translated to the following query:

```
SELECT ?x0 WHERE {
  ?x0 <http://dbpedia.org/property/near> ?y2 .
  ?x0 a <http://diadem.cs.ox.ac.uk/ontologies/real-estate#House> .
  ?v <http://purl.org/goodrelations/v1#includes> ?x0 .
  ?x0 <http://www.w3.org/2006/vcard/ns#street-address> ?y0 .
  ?v <http://diadem.cs.ox.ac.uk/ontologies/real-estate#hasPrice> ?y1 .
  ?y2 a <http://linkedgeodata.org/ontology/Supermarket> .
  ?x0 <http://purl.org/goodrelations/v1#description> ?y .
  FILTER(regex(?y0,'Oxfordshire','i')) .
  FILTER(regex(?y,'Edwardian ','i')) .
  FILTER(?y1 < 1000000) .
}
```

In that case, the links to LinkedGeoData were used by selecting the "near" property as well as by finding the correct class from the LinkedGeoData ontology.

4.2 Performance Evaluation

We conclude this evaluation with a brief look at the system performance, focusing on the resource intensive background extraction and linking, which require several hours compared to seconds for the actual query evaluation. For the real-estate scenario, the TBSL algorithm requires 7 seconds on average for answering a natural language query using a remote triple store as backend. The performance is quite stable even for complex queries, which required at most 10 seconds. So far, the TBSL system has not been heavily optimised in terms of performance, since the research focus was clearly to have a very flexible, robust and accurate algorithm. Performance could be improved, e.g., by using fulltext indices for speeding up NLP tasks and queries.

Extraction. In [9] we show that OXPath 's memory requirements are independent of the number of pages visited: For DEQA, the average execution time of our wrappers amounts to approximately 30 pages/min. As we do not want to overtax the agencies' websites, this rate is high enough to crawl an entire website in few minutes. For OXPath this rate is quite slow, but is rooted in inherent characteristics of the domain: **(1)** Many real estate websites are *unable to serve requests at higher rates,* and **(2)** supply *heavily scripted pages,* containing many images or fancy features like flash galleries. Indeed, the evaluation of OXPath is dominated by the browser initialisation and rendering time [9], amounting to over 80% in the real estate case.

Linking. The runtime of the link discovery depends largely on the amount of data to link. In our use case, fetching all data items for linking from the endpoints

required less than 3 minutes while the link discovery process itself was carried out in 0.6 seconds for discovering the near-by entities and 0.3 seconds for the entities at walking distance.

In summary, the data extraction and linking can be easily done in a few minutes per agency and can be run in parallel for multiple agencies. This allows us to refresh the data at least once per day, without overtaxing the resources of the agencies.

5 Related Work

DEQA is, to the best of our knowledge, the first *comprehensive deep web question answering* system addressing the whole process from data extraction to question answering. In contrast, previous approaches have been limited either with respect to their access to deep web data behind scripted forms [20] by targeting only common-sense, surface web data, or by requiring user action for form navigation (MORPHEUS, [11]). Though "federated" approaches that integrate data from different forms have been considered [17], none has integrated the extracted data with existing background knowledge, limiting the types of questions that can be answered. In the following, we briefly discuss related work for each of DEQA's components to illustrate why we believe this is the right combination.

Web Extraction. To extract the relevant data from the real estate agencies, we can resort essentially to three alternatives in web data information extraction [7], namely traditional information extraction, unsupervised data extraction, or supervised data extraction, with OXPath falling into the last category. *Information extraction* systems, such as [8,2], focus on extraction from plain text which is not suitable for deep web data extraction of product offers, where most of the data is published with rich visual and HTML structure, yielding much higher accuracy than IE systems. *Unsupervised data extraction* [30,13] approaches can use that structure, but remain limited in accuracy mostly due to their inability to distinguish relevant data from noise reliably. Thus, the only choice is a supervised approach. In [9] OXPath and related supervised approaches are discussed at length. In summary, OXPath presents a novel trade-off as a simpler, easier language with extremely high scalability at the cost of more sophisticated data analysis or processing capabilities. As shown in DEQA, such abilities are better suited for post-processing (e.g., through LIMES for linking).

Linking. LIMES [24] offers a complex grammar for link specifications, and relies on a hybrid approach for computing complex link specifications. In contrast to LIMES, which employs lossless approaches, [26] uses a candidate selection approach based on discriminative properties to compute links very efficiently but potentially loses links while doing so. Link Discovery is closely related with record linkage and deduplication [5]. Here, the database community has developed different blocking techniques to address the complexity of brute force comparison [14] and very time-efficient techniques to compute string similarities for record linkage (see e.g., [29]). In recent work, machine learning approaches

have been proposed to discover link specifications. For example [21] combine genetic programming and active learning while [25] learns link specifications in an unsupervised manner.

Question Answering. There is a range of approaches to QA over structured data, for an overview see [18]. Here we discuss TBSL in contrast to two prominent systems to exemplify two opposite key aspects: *PowerAqua* [19], a purely data-driven approach, and *Pythia* [27], which heavily relies on linguistic knowledge. TBSL specifically aims at combining the benefits of a deep linguistic analysis with the flexibility and scalability of approaches focusing on matching natural language questions to RDF triples. This contrasts with *PowerAqua* [19], an open-domain QA system that uses no linguistic knowledge and thus fails on questions containing quantifiers and comparisons, such as the most and more than. *Pythia* [27], on the other hand, is a system that relies on a deep linguistic analysis, yet requires an extensive, manually created domain-specific lexicon.

6 Conclusion

DEQA is a comprehensive framework for *deep web question answering*, which improves existing search functionality by combining web extraction, data integration and enrichment as well as question answering. We argue that recent advances allow the successful implementation of the DEQA framework and consider this to be one of the prime examples for benefits of semantic web and artificial intelligence methods. We instantiate DEQA for the real estate domain in Oxford and show in an evaluation on 100 user queries that DEQA is able to answer a significant percentage correctly. In addition, we provided a cost analysis which describes the setup and maintenance effort for implementing DEQA in a particular domain. All used software components as well as the actual queries and used configuration files are freely available (`http://aksw.org/projects/DEQA`).

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: A Nucleus for a Web of Open Data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC/ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
2. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open information extraction from the web. In: IJCAI, pp. 2670–2676 (2007)
3. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American (2001)
4. Bizer, C., Schultz, A.: The R2R framework: Publishing and discovering mappings on the web. In: COLD (2010)
5. Bleiholder, J., Naumann, F.: Data fusion. ACM Comput. Surv. 41(1), 1–41 (2008)
6. Bühmann, L., Lehmann, J.: Universal OWL Axiom Enrichment for Large Knowledge Bases. In: ten Teije, A., Völker, J., Handschuh, S., Stuckenschmidt, H., d'Acquin, M., Nikolov, A., Aussenac-Gilles, N., Hernandez, N. (eds.) EKAW 2012. LNCS, vol. 7603, pp. 57–71. Springer, Heidelberg (2012)

7. Chang, C.H., Kayed, M., Girgis, M.R., Shaalan, K.F.: A survey of web information extraction systems. IEEE TKDE 18(10), 1411–1428 (2006)
8. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Unsupervised named-entity extraction from the web: an experimental study. Artif. Intell. 165, 91–134 (2005)
9. Furche, T., Gottlob, G., Grasso, G., Schallhart, C., Sellers, A.: OXPath: A language for scalable, memory-efficient data extraction from web applications. In: VLDB, pp. 1016–1027 (2011)
10. Gerber, D., Ngonga Ngomo, A.-C.: Bootstrapping the Linked Data Web. In: Proc. of WekEx at ISWC (2011)
11. Grant, C., George, C.P., Gumbs, J.D., Wilson, J.N., Dobbins, P.J.: Morpheus: a deep web question answering system. In: iiWAS, pp. 841–844 (2010)
12. Gulhane, P., Madaan, A., Mehta, R., Ramamirtham, J., Rastogi, R., Satpal, S., Sengamedu, S.H., Tengli, A., Tiwari, C.: Web-scale information extraction with vertex. In: ICDE, pp. 1209–1220 (2011)
13. Kayed, M., Chang, C.H.: FiVaTech: Page-level web data extraction from template pages. IEEE TKDE 22(2), 249–263 (2010)
14. Köpcke, H., Thor, A., Rahm, E.: Comparative evaluation of entity resolution approaches with fever. In: VLDB, pp. 1574–1577 (2009)
15. Kranzdorf, J., Sellers, A., Grasso, G., Schallhart, C., Furche, T.: Spotting the tracks on the OXPath. In: WWW (2012)
16. Lehmann, J., Auer, S., Bhmann, L., Tramp, S.: Class expression learning for ontology engineering. J. of Web Semantics 9, 71–81 (2011)
17. Lin, J.: The Web as a resource for question answering: Perspectives and challenges. In: LREC 2002 (2002)
18. Lopez, V., Uren, V., Sabou, M., Motta, E.: Is question answering fit for the semantic web? A survey. Semantic Web J. 2, 125–155 (2011)
19. Lopez, V., Fernández, M., Motta, E., Stieler, N.: PowerAqua: Supporting users in querying and exploring the Semantic Web content. Semantic Web Journal (to appear), http://www.semantic-web-journal.net/
20. Mollá, D., Vicedo, J.L.: Question answering in restricted domains: An overview. Comput. Linguist. 33(1), 41–61 (2007)
21. Ngonga Ngomo, A.-C., Lyko, K.: EAGLE: Efficient Active Learning of Link Specifications Using Genetic Programming. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 149–163. Springer, Heidelberg (2012)
22. Ngonga Ngomo, A.-C.: A time-efficient hybrid approach to link discovery. In: OM@ISWC (2011)
23. Ngonga Ngomo, A.-C., Auer, S.: A time-efficient approach for large-scale link discovery on the web of data. In: IJCAI (2011)
24. Ngonga Ngomo, A.-C., Lehmann, J., Auer, S., Höffner, K.: Raven – active learning of link specifications. In: OM@ISWC (2011)
25. Nikolov, A., d'Aquin, M., Motta, E.: Unsupervised Learning of Link Discovery Configuration. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 119–133. Springer, Heidelberg (2012)
26. Song, D., Heflin, J.: Automatically Generating Data Linkages Using a Domain-Independent Candidate Selection Approach. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 649–664. Springer, Heidelberg (2011)

27. Unger, C., Cimiano, P.: Pythia: Compositional meaning construction for ontology-based question answering on the Semantic Web. In: NLDB (2011)
28. Unger, C., Bühmann, L., Lehmann, J., Ngomo, A.C.N., Gerber, D., Cimiano, P.: Template-based question answering over RDF data. In: WWW, pp. 639–648 (2012)
29. Xiao, C., Wang, W., Lin, X., Yu, J.X.: Efficient similarity joins for near duplicate detection. In: WWW, pp. 131–140 (2008)
30. Zhai, Y., Liu, B.: Structured Data Extraction from the Web Based on Partial Tree Alignment. IEEE TKDE 18(12), 1614–1628 (2006)

QuerioCity: A Linked Data Platform for Urban Information Management

Vanessa Lopez, Spyros Kotoulas, Marco Luca Sbodio,
Martin Stephenson, Aris Gkoulalas-Divanis, and Pól Mac Aonghusa

Smarter Cities Technology Centre, IBM Research, Ireland

Abstract. In this paper, we present QuerioCity, a platform to catalog, index and query highly heterogenous information coming from complex systems, such as cities. A series of challenges are identified: namely, the heterogeneity of the domain and the lack of a common model, the volume of information and the number of data sets, the requirement for a low entry threshold to the system, the diversity of the input data, in terms of format, syntax and update frequency (streams vs static data), and the sensitivity of the information. We propose an approach for incremental and continuous integration of static and streaming data, based on Semantic Web technologies. The proposed system is unique in the literature in terms of handling of multiple integrations of available data sets in combination with flexible provenance tracking, privacy protection and continuous integration of streams. We report on lessons learnt from building the first prototype for Dublin.

1 Introduction

Governments are increasingly making their data accessible to promote transparency and economic growth. Since the first data.gov initiative launched by the US government, many city agencies and authorities have made their data publicly available through content portals: New York City[1], London[2], San Francisco[3], Boston[4], and Dublin[5], to name a few.

Through these efforts, a large number of data sets from many different domains became available, allowing enterprises and citizens to create applications that can mash up data. However, the data sets shared in these portals come in different formats (csv, xml, kml, pdf,..), do not link to other sources on the Web, are heterogeneous and of variable quality. Semantic Web technologies have been adopted as a valuable solution to facilitate large-scale integration, sharing of distributed data sources and efficient access to government data [1,2]. However, converting raw government data into high quality Linked Government Data is

[1] http://www.nyc.gov/html/
[2] http://data.london.gov.uk/
[3] http://datasf.org/
[4] http://www.cityofboston.gov/doit/databoston/app/data.aspx
[5] http://www.dublinked.ie

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 148–163, 2012.

costly [3,4], and there is a lack of practical approaches for converting and linking government data at scale [5]. Consequently, linked RDF data is sparse and often limited to metadata, and content in data portals is difficult to consume by web developers and users. In addition, dealing with dynamic data sources like Streams and allowing for multiple integrations of Linked Data remains an open problem.

We have developed QuerioCity, an open urban information management platform, based on semantic technologies, to easily capture and consume urban open data, with a particular focus on transforming, integrating and querying heterogenous semi-structured data in an open and large environment. The key research questions challenged were: How to represent and manage city-scale data as an information resource in practical and consumable ways? What are the challenges involved in creating an urban information management architecture with acceptable performance levels? What are the benefits and costs of using Linked Data technologies to allow people and systems to interact with the information ecosystem of a city? How can we provide privacy protection and capture provenance in an open world where traditional notions of information governance and control may no longer apply? How do the answers to the questions above change when dealing with streams instead of static data?

This paper describes the challenges, findings and lessons learnt from the first prototype of QuerioCity. The platform differs from existing open data initiatives in the following aspects: A strong focus on operational *live data* coming from physical sensors (transport, water and energy) or social media, the ability to *mix public and restricted data*, the transformation of raw data and metadata coming from data publishers in *various formats and structures* into linked open data at *enterprise scale*, and the ability to *detect and thwart privacy threats*.

In this paper, we are using the data portal for Dublin, named Dublinked, as a use-case, providing valuable real-world data, insight and challenges. Dublinked provides a real experimental validation for this research, a live and scalable scenario, where real-world data sets are obtained from four city authorities in the Dublin area: Dublin City, Dun Laoghaire-Rathdown, South Dublin and Fingal County Councils. The methods and technologies proposed in this paper are gradually rolled-out in Dublinked.

The rest of the paper is structured as follows. Section 2 presents the state-of-the-art and elaborates on the challenges of dealing with Urban data. We describe the rationale of QuerioCity in Section 3, along with its enterprise software components in Section 3.4. Our discussion, in Section 4, includes lessons learned and an analysis of the costs and benefits of Semantic Technologies.

2 Related Work and Research Challenges

The rise of the Open Data movement has contributed to many initiatives whose aim is generating and publishing government and geographical data according to Linked Data principles, such as OpenStreetMaps[6] and OrdnanceSurvey[7].

There are automated approaches for turning tabular data into a Semantic Web format. Pattern-based methods for re-engineering non-ontological resources into ontologies [8] are based on the use of thesauri, lexica and WordNet for making explicit the relations among terms. TARTAR [9] automatically generates knowledge models out of tables. In this system, grounded in the cognitive table model introduced by Hurst [10], a table is handled from a structural, functional and semantic point of view by respectively identifying homogeneous regions (group of cells) in a table, distinguishing between attribute cells and instance cells, and then finding semantic labels for each region content with the help of WordNet. The coverage of these approaches depends on WordNet or an ontology that models the domains of interest. To annotate tables on the Web and improve search, [11] uses a column-based approach. A class label is attached to a column if a sufficient number of the values in the column are identified with that label in some "is-a" databases extracted from the Web.

In [2], a new dataset-specific ontology is constructed for each dataset, representing only the data stored in the particular database. To convert this data into RDF, scripts are developed in correspondence with their manually-designated and built ontologies.

A number of tools for automatically converting tabular data (mostly CSV) into RDF also exist, such as RDF123 [12]. W3C defines a standardised mapping language R2RML[6] and an approach for converting relational databases to RDF. In this W3C candidate recommendation, the first row is used to suggest properties and each other row refers to entities, with one of the columns uniquely identifying the entity. This approach is used, for example, in the Datalift project [13] to automate the conversion from the source format to "raw RDF", before transforming it to "well-formed" RDF by using selected vocabularies and SPARQL construct queries.

The approach presented in [3] is based on Google Refine for data cleaning and a reconciliation service extended with Linked Data capabilities to enable exporting tabular data into RDF, while keeping provenance description represented according to the *Open Provenance Model Vocabulary* [14]. In our experience with Dublinked, asking the users to use tools such as Google Refine and define templates to guide the conversion process into RDF have limited fitness-for-use for the non-expert.

More often than not, urban data is sourced from legacy non-relational systems or spreadsheets made for consumption by humans. Urban data does not follow a relational model, it is highly heterogeneous and the structure is unknown (from static data to spatial-temporal data obtained from physical sensors). State-of-the-art approaches do not solve the entity recognition and type identification problem since data does not always come in a tabular format, and when it does, we cannot assume that each row is an entity, or that all the entities described are explicitly labeled in the table. For instance, Dublin City Council (DCC) published a dataset about energy consumption in the City Council Civic offices, as part of an initiative to reduce its carbon footprint by 2030. The dataset

[6] http://www.w3.org/TR/r2rml/

contains files, partly represented in Figure 1, with energy readings recorded every 15 minutes. These readings are split in different files for the new building blocks (Block 1 & 2) and the old ones (not shown in the figure). The location (DCC civic offices) and kind of data described in the data sets (energy measurements) is given in the text description in the metadata. In the content, the first row contains a unique cell with the blocks where the measurement was taken, the first column in the table represents dates, and the second column after the third row is the time of the day when the measurement was taken. Thus, each row of this dataset contains multiple entities of type measurement with properties to represent a given value and a given timestamp. *Automated methods have focused on relational data and are not yet able to deal with such structures.*

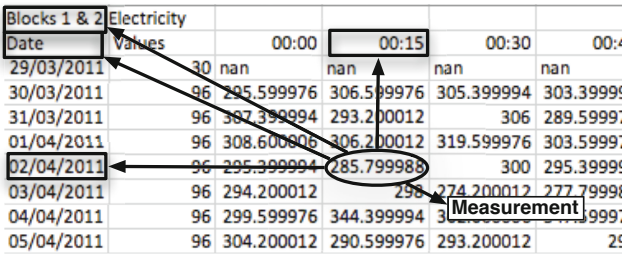


Fig. 1. Data sets about energy measurement in Dublin civic offices

As stated in [2], it is not realistic to assume that an organisation will subscribe to a single schema, or that different organisations will agree in a common semantic model. For instance, in Dublinked, each of the four county councils have published a data set about street lighting consisting in an inventory of pole locations. DCC represents this information in two CSV files: one including an ID and a name, and a second including an ID, Irish Grid spatial projections (easting, northing) and a location description. Fingal County council includes street name, outside / opposite house number, and easting / northing coordinates. Public Lighting locations in South Dublin County are described with IG spatial coordinates (ITMEast, ITMNorth), road names, and a set of "location" such as cul-de-sac, front, junction, school, lane way, etc. Dun Laonghaire county released the data as shape files. *Although all four data sets are valid, in the same domain, and machine-processeable, they are far from a common model.*

Urban information often comes in *Streams*. Although there have been efforts in integrating streams with Linked Data [15], the effort to do so has been centralized and manual.

Privacy has been traditionally offered on individual data sets of simple data types, where sensitive inferences are easy to predict. In Linked Data, and in an urban information management domain, data sets are characterized by the four Vs (velocity, variety, volume and veracity) and sensitive inferences can be drawn across multiple data sets and are thereby difficult to predict.

Summarizing, managing urban information raises challenges in terms of:

- *Fitness-for-use.* The users of the system are not data integration experts and not qualified to use industry data integration tools. Furthermore, they are not able to query data using structured query languages.
- *Domain modeling.* The domain of the information is very broad and open. As such, generating and mapping data to a single model is infeasible or too expensive. Even if such a model was to be created, it's complexity would hamper its use, given the target audience of the system.
- *Global integration.* Addressing the information needs for solving problems in an urban environment requires integration with an open set of external data sets. Furthermore, it is desirable that city data becomes easily consumable by other parties.
- *Scale.* The data in a city changes often (streams), is potentially very large and it is interlinked with an open set of external data.
- *Privacy.* Data sets may contain sensitive information that needs to be privacy-protected prior to their sharing. Even more, the linkage of data sets may lead to sensitive inferences which have to be blocked in accordance with legislation.

3 The QuerioCity Approach

We propose an approach where the integration effort is fundamentally incremental and split between the two major roles in the system: *data publisher* and *data consumer*. Data publishers need interactive tools and patterns to "lift" the data as much as possible, adding meaning to data through semantic annotations, linking across data sets and (partially) with large existent corpora in the Web, and protecting any sensitive information. On the other side, data consumers pull the data in order to fulfill their needs through searches for potentially relevant data sets, and by executing complex queries across data sets in an intuitive and exploratory fashion. In the process, the integration effort of consumers is reusable.

The data currency of Queriocity is a *Dataset*[7]. *Datasets* consist of the *Metadata* described in Section 3.1, a number of data *Graphs* or data *Streams*, and *Provenance* information, described in Section 3.2.

Figure 2 illustrates the overall approach taken in QuerioCity. Vertically, we illustrate the progression from raw source data to consumption. Initially, the system archives and catalogs *metadata* of the content (for example, keywords and publishing date), enabling rich queries over this meta-information, in combination with full-text search using Lucene[8] inverted indexes.

Good metadata is important in order to allow individuals to easily discover relevant data. However, this is not enough to answer queries that span various

[7] In this paper, we will refer to *Dataset* as a data artifact. We will refer to data set as the abstract notion.

[8] http://lucene.apache.org

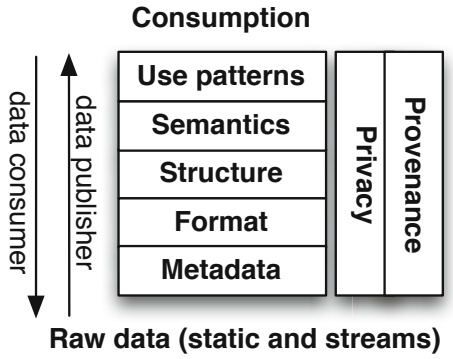


Fig. 2. The QuerioCity approach

data sets. By transforming the input into a uniform *format*, we are allowing queries where the data consumer knows the structure and the content of each file (possible by manual examination of the input). By harmonizing the *structure* of the various inputs, the user is able to query based on manually-mapped properties across data sets. Integrating the data into a common *semantic structure* allows transparent querying across data sets, without the need for manual mappings. Finally, given the heterogeneity (and consequently the semantic complexity) of the data, inferring *use patterns* allows users to create and share meaningful "views" over the data.

There are some issues that span all aspects of the system. Given the open nature of the integration process, it is imperative that the system records *information provenance*. In addition, given the potentially sensitive nature of the information, the system provides functionality for detecting *privacy threats* and tools for *anonymization* [16].

In what follows, we go into more detail for some of the key components of QuerioCity.

3.1 Metadata

In any data portal, and in Dublinked in particular, the metadata provides a fairly rich description of the data sets, which is critical for discovery and navigation. Some fields are given by the publisher, such as title, subject, keywords, publishing agency, license terms, collection purpose, period of coverage; and some others are updated automatically, such as date of creation, latest update and download links.

In this section, we outline, on the one hand, the process of generating Linked Data from Dublinked metadata: generation of the ontology model and the RDF data, and alignment of the datasets and links to external sources; and on the other hand how the generated Linked Data is used for user consumption and for building a publishing interface.

The ontological model created to represent the metadata catalogs is based on standard and widely used vocabularies, namely dublin core[9], FOAF[10] and DCAT[11]. New resources are accessible through HTTP following the W3C best practices for publishing Linked Data: resolving a dereferenceable URI gives relevant facts about the entity across all metadata sources.

Datatypes are standardised (e.g., using *xsd:date* and *dcterms:PeriodOfTime*) and instances are created for representing agencies, spatial administrative areas, access rights, update frequency, distribution format, keywords and categories. Consistent metadata is created by defining the range of properties as a given type. For instance, spatial administrative areas can refer to instances of county councils (Dun Laoghaire-Rathdown, Fingal, SouthDublin), city councils (Dublin city council), city regions (Greater Dublin, South East Inner City, Dublin city centre), and so on.

We are getting lot of value in standardisation and cleansing of the original data by virtue of this approach, e.g., spelling mistakes and duplicates are eliminated (e.g.,"DLR" council" and "Dun Laoghaire Rathdown County Council" are alternative labels for the same entity) and datasets are linked by the area they cover, the publishing agency, publishing date, etc.

The metadata is also linked to authoritative and external sources on the Web. Categories and keywords are mapped to the Integrated Public Sector Vocabulary (IPSV), which is a controlled vocabulary for populating the e-Government Metadata Standard that UK public sector organisations are required to comply with. The IPSV is available as an RDF Schema (based on SKOS) by the esd-toolkit[12]. Keywords are not fixed a priori, but categories are predefined in the ontology model. New categories are automatically added only if they have a direct mapping with an IPSV category. String distance metrics (Cohen et al., 2000), mainly a combination of Jaro and TFIDF, are used to automatically map the metadata to the best matching terms in IPSV (including all labels) and to avoid duplicated, e.g., keywords such as "coast" and "coastline" are both matched to the IPSV entity ipsv#527 "Coasts". However, noise and inaccuracies can be introduced by automatic approaches, e.g., "asset management" is matched to the IPSV term "waste management". These inaccuracies can be avoided at publishing time with the use of a semantically-aware publishing interface.

Equivalence *owl:sameAs* links to corresponding DBpedia entities are added, if any, to describe subjects and locations such as administrative counties. Linking to IPSV and external sources improves interoperability and discoverability of related datasets, for instance datasets are not just related because they use the same keyword, but also because they link to related, broader or narrower categories in IPSV. Furthermore, as shown in the examples in Section 2, for many of the datasets the entity described by the content is only explicitly mention in the metadata description. Linking the metadata to entities in DBpedia and IPSV

[9] http://dublincore.org/specifications
[10] http://xmlns.com/foaf/spec
[11] http://www.w3.org/TR/vocab-dcat/
[12] http://doc.esd.org.uk/IPSV/2.00.html

allows us to solve the type identification problem in many cases. Using the pole locations dataset, which is linked to the IPSV terms *street lighting* (ipsv#1404), *Street furniture* (ipsv#3103) and *Road and pathway maintenance* (ipsv#3025), the entities described by the data are of the most specific IPSV entity type, and also DBpedia category, "street lighting" (also known as "lamp posts").

As mentioned before, for newly published datasets we use a semantically-aware publishing interface that allows us to (1) validate on the fly external or internal mappings while the publisher fills in the metadata fields, (2) limit the user input to a set of instances of the appropriate range for a given property, or (3) allow the user to input free text but use existent metadata and the IPSV vocabulary to present suggestions to asset publishers annotating their metadata. For example, if the user start writing the keyword "parking", the system will propose further refining the input with the annotations "car park", "car parking permits", "resident parking", "disabled parking", "parking fines", "parking meters".

Besides a SPARQL endpoint, ranked searches and RESTful services are also provided. Figure 2 shows a summary view of the RDF metadata in Dublinked for the 209 datasets currently available, about 27K triples.

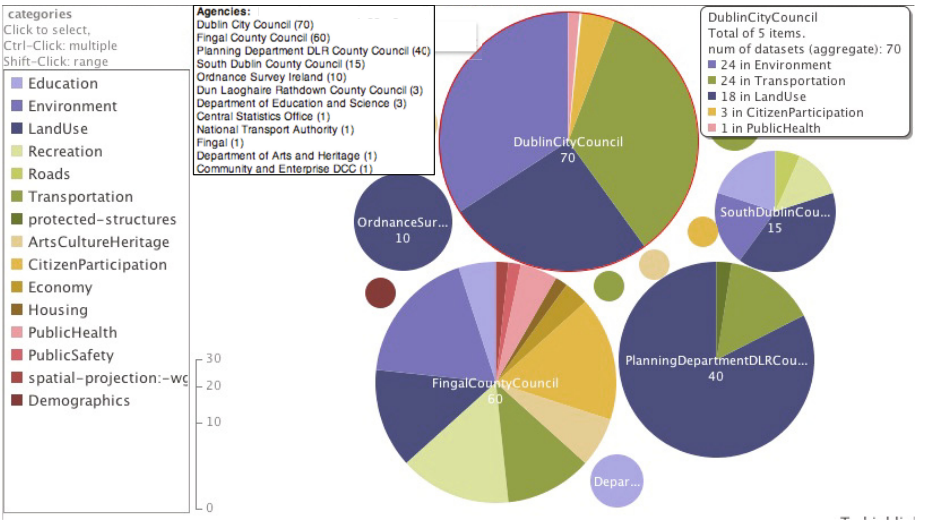


Fig. 3. Screenshot illustrating metadata from Dublinked by category and agency

In the future, we intend to use content to automatically push into the metadata information about the bounding box covered by the data. As such different datasets in different topics can also be dynamically link by spatial properties. Currently this field exists in the metadata, but none of the publishers fill it in.

3.2 Content

QuerioCity takes an incremental approach to managing content. Referring to figure 2, after extracting the relevant metadata, data sets go through the following steps towards consumption:

Format. All datasets in the system are preserved in their original formats. In addition, we transform datasets with known file formats to a simple RDF representation. This representation is not intended to capture semantics, but rather to provide a convenient and uniform way to represent the content of file. For example, the CSV file in figure 1, would be converted in triples of the form `:c1 a Cell. :c1 col "1". :c1 row "1". :c1 value "Blocks 1 & 2"`. This choice is motivated by the fact the it is not always possible to automatically convert CSV files to representations of entities.

Structure. Once homogeneous data areas are identified and validated with the user, pre-defined templates can be used as semantic masks that capture the intent of the publisher and guide the extraction of entities, making explicit the relations that hold between the entities described on the tables. Templates can be defined a priori but they can also be learned through user interaction and saved to be reused. The three dominant structures for tabular data in Dublinked are: geographically referenced entities (i.e. tables with two columns for longitude and latitude), measurements with a single entity per row and a column indicating the temporal aspect, and structures similar to that in Figure 1, representing measurements that reference time through both a given column and a row.

Semantics. The platform leverages semantic data types (geographical coordinates, dates, etc.) and automatically converts units of measurement. *Owl:sameAs* and *owl:equivalentAs* properties are used to link entities, eliminating the need for tight physical integrations imposed by relational databases and adopting a pay-as-you-go approach. These properties are discovered using a combination of existent reconciliation and state-of-the-art mapping techniques to detect common types and entity co-reference as well as user input. In addition, we can consume services provided by the `http://sameAs.org` web site for getting co-referent URIs for a given URI.

Use Patterns. Given the open nature of the domain, QuerioCity allows for multiple integrations and usage paths for Datasets. As we explain in the next paragraph, the system allows for efficient maintenance of such paths.

Data Model. In the QuerioCity data model, instance and schema information is stored in separate Named Graphs from the Metadata. Graphs are shared between Datasets (i.e. a Dataset may reference multiple Graphs, and a Graph can be referenced by multiple Datasets). Data integration tasks entail creating new Datasets. Instance data is append-only and schema information is read-only. This provides, possibly externally referenced Datasets with stability, while avoiding unnecessary data duplication.

Streams are handled using the same model. In the context of our system, Streams are analogous to Graphs; a Dataset can reference a set of Streams using URIs (possibly, in combination with Graphs). We make a distinction between

two types of queries over streams: querying over historical stream information, by generating RDF on-demand, and querying over live Streams. For live streams we use IBM Infosphere Streams[13], with a custom extension that allows for referring to data fields from a stream within a C-SPARQL [17] query.

To process historical stream data, we developed a REST API which transforms on-demand a time window of an archived data stream into RDF. Archived streams are stored as CSV files on a file system. The platform indexes the CSV files, and can convert a time window (potentially spanning multiple files) into an RDF Graph that is stored in an RDF strore for future use. The stream-data-to-RDF transformation process is asynchronous, because, depending on the requested time window, the processing time may be considerable.

As an example, we provide some details about the stream with information about Dublin buses. The stream provides information about approximately 600 active buses (bus line, location, delay, congestion, etc.) with updates every 20 seconds, and it is archived on a daily basis. We currently have 26 GB of bus stream data. On average, one line of a bus stream CSV file is transformed into 10 RDF triples, and we achieve a throughput of around 13000 triples / sec.

Querying. In the Semantic Web, there are three main approaches towards data integration: Query rewriting, dataset transformations (e.g. through mappings/links) and using reasoning. In QuerioCity, we take the reasoning approach, that presents the distinct advantages that the data integration is both transferable to other datasets and concise while the other two methods each present one of these advantages. In fact, integration through this method does not result in complicated queries: For example, a SPARQL query over a given Dataset would be in the form: `SELECT ... WHERE {?d rdf:type void:Dataset. ?g void:inDataset ?d. GRAPH ?g {...}}`.

Privacy. Compared to state-of-the-art platforms, an important differentiator of QuerioCity is privacy provisioning. Data privacy in QuerioCity is offered both at the dataset-level and on the graph-level (i.e., Linked Data). In particular, datasets that are uploaded to the platform undergo a semi-automated vulnerability check that aims at identifying potential privacy leaks. Based on the outcome of this process and on user input, the data is subsequently privacy-protected prior to being shared. As an example of the dataset-level privacy-protection mechanism, assume that a data publisher wishes to share the dataset (a) shown in Figure 4[14]. Applying vulnerability checking on this dataset, reveals that attributes *Position* and *Department* can lead to re-identification attacks, because their attribute-values combination is unique for some individuals. Moreover, the data publisher may indicate that *Salary* is a sensitive attribute that should be

[13] http://www-01.ibm.com/software/data/infosphere/streams/

[14] This is a sample of a dataset recently published online by the City of Chicago. Conforming to the US laws, the dataset lists current government employees, complete with full names, departments, position, and annual salaries. In our example, we consider a de-identified version of the dataset. The complete dataset is available at: https://data.cityofchicago.org/Administration-Finance/Current-Employee-Names-Salaries-and-Position-Title/xzkq-xp2w

Empl. ID	Position	Department	Salary
ID1	ADMIN. ASS	TRANSPORTN	$50280
ID2	ADMIN. ASST	INSPECTOR GEN	$70164
ID3	ADMIN. ASST	MAYOR'S OFF.	$40008
ID4	ADMIN. ASST	MAYOR'S OFF.	$62496
ID5	FIN. OFFICER	LAW	$80256
ID6	FIN. OFFICER	STREETS	$91864

(a)

Position	Department	Salary
ADMIN. ASS	*	$50K-$75K
ADMIN. ASST	*	$50K-$75K
ADMIN. ASST	MAYOR'S OFF.	$40K-$65K
ADMIN. ASST	MAYOR'S OFF.	$40K-$65K
FIN. OFFICER	LAW/STREETS	$80K-$95K
FIN. OFFICER	LAW/STREETS	$80K-$95K

(b)

Fig. 4. Example table before and after privacy protection

disclosed in ranges. Accordingly, an anonymization of this dataset, which protects individuals from re-identification attacks, can result in the dataset (b) shown in Figure 4.

Provenance. We keep both dataset-level provenance and graph-level provenance, storing *derivedFrom* relationships for both Datasets and Graphs. Graph-level provenance is tunable to the resolution required, by splitting Graphs. In the extreme case, we can keep a single graph per triple, so as to have triple-level provenance. Needless to say, this will have a negative impact on performance and we have yet to encounter the need for it. In QuerioCity, provenance is *operational* with regard to privacy. When a privacy threat is detected for a given Dataset, it to not sufficient to protect this Dataset in isolation, since the process to generate it could be repeated. We use the *derivedFrom* relations and protect *all* Datasets that were used to create the Dataset with privacy vulnerabilities.

3.3 Putting It All Together

We illustrate the design rationale through the example in Figure 5. When importing a data set into the system, we create two Datasets: Dataset A links to the Metadata (not shown in the figure) and a pointer to the URL of the source files, similar to a standard content portal. Dataset B links to the *same* metadata and the simple RDF representation described in the previous paragraph. Extracting entities would require structural changes to the RDF representation, which means that a new Graph will be generated to which the Dataset C will refer to. It is further possible to use standard datatypes (e.g. normalize spatial projections to WSG84[15]). In QuerioCity, this is accomplished by creating new properties with such values. Dataset D can refer to *both* the Graph containing the entities and a new Graph containing such new properties. Finally, we can also map Dataset D to Dataset F, to generate Dataset E. In this case, Dataset E can refer to the Graphs of Dataset D, Dataset F and a Graph containing the mappings. For each of these Datasets, we keep coarse-grain provenance information.

3.4 Deployment

In this section, we describe an internal deployment of the QuerioCity platform, consisting mainly of IBM technology, with some open-source components. The

[15] W3C Basic Geo (WGS84 lat/long) Vocabulary www.w3.org/2003/01/geo/

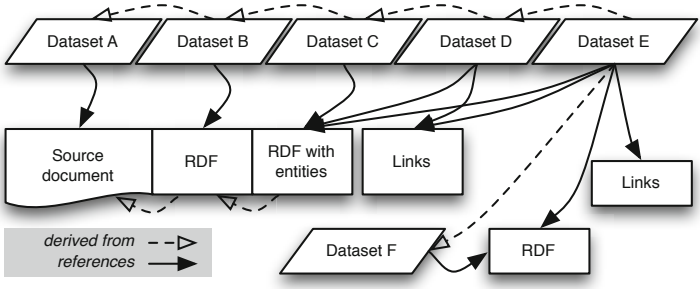


Fig. 5. Example for content data model

use well established commercial components, which can be clustered as required, ensures scalability and robustness. Figure 6 outlines the main software components of our system.

The Secure FTP (*SFTP*) server component allows publishers to securely upload multiple files for publishing. Such files are then mirrored on the Storage Area by the the *Publish & Sync App*. We use a fibre-connected *IBM Storage Array Network* to store the published data: this is secure, resilient and recoverable. *IBM InfoSphere Streams* provides the processing capabilities for live streams. The platform provides data access control using *IBM Tivoli Directory Server (TDS)*, which stores the credentials of users that can access restricted data sets or publish data sets. Users can retrieve data sets through HTTP: the platform uses *IBM HTTP Server (IHS)* to answer these requests after having checked the user credentials with *TDS*. We use *IBM WebSphere Application Server (WAS)* to host the (i) QuerioCity Data Layer, (ii) the Web user interface for searching and publishing metadata, and (iii) a set of REST APIs for searching, querying, browsing and downloading data. Finally, a SPARQL endpoint is available to work directly on RDF data, which are stored on a DB2 RDF store.

QuerioCity is based on commercial, enterprise-grade software. Critical components, such as the WAS, DB2 and the HTTP server can be clustered as required, providing scalability and robustness.

4 Lessons Learned

The infrastructure to run enterprise Semantic Web applications is finally mature. As described in Section 3.4, QuerioCity runs almost exclusively on enterprise-grade components. Nevertheless, emerging research technologies in Linked Data and semantic integration are well behind similar technologies for relational data. Selecting the appropriate ontology for describing a dataset, if it exists, and linking across datasets, are tedious tasks, which require significant expertise and lead to publication processes that are not scalable. A significant obstacle in building an urban Linked Data platform is to strike a balance between automatic approaches and user interaction to achieve continuous and incremental integration of streams.

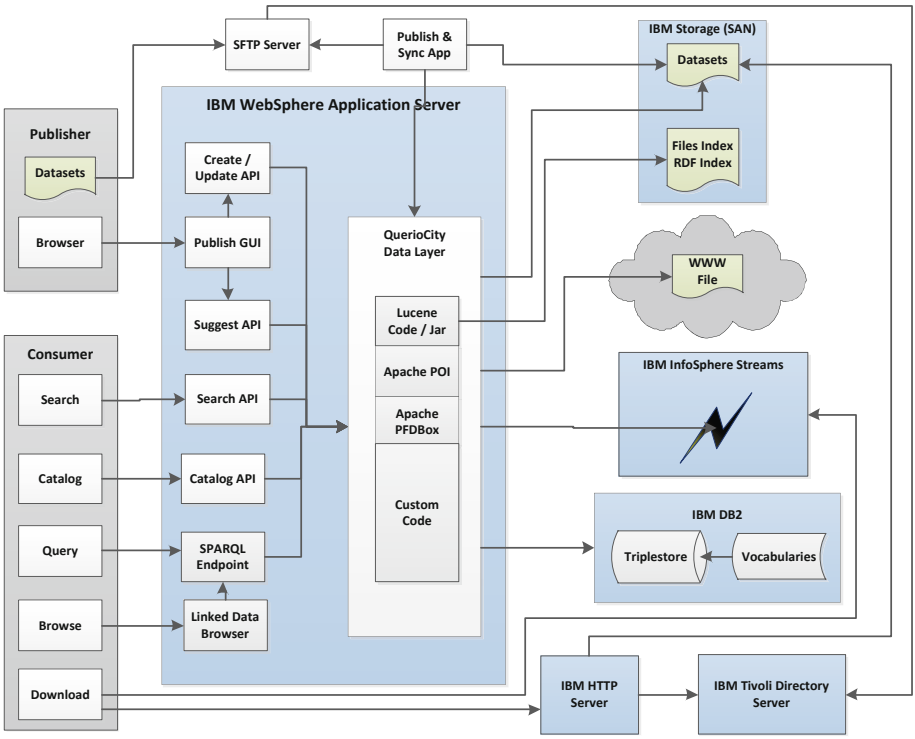


Fig. 6. Software components in QuerioCity

A pay-as-you-go semantic approach mitigates risk, since it allows publishers to partially complete and re-use integration tasks. Data need to be delivered in any shape and format with minimum cost to encourage participation and engage data providers. More work is required to find practical and inexpensive approaches to semantically annotate and RDFize the content of the collected data, provide insights into its quality and allow seamless data integration that can augment the value of the data.

In our experience, visualisations with high levels of aggregation, such as the one displayed in Figure 3, are preferable for citizen and city officials.

Visualizations are layered on SPARQL query results to enable citizens make use of and benefit from open data. As city data is situated on a specific temporal and geographical context, further insight is given by comparing datasets through spatial-temporal visualizations or heat maps (optionally points of interest can be extracted from sources such as Linked Open Street Maps [6]). By integrating diverse information sources, and making them consistently queriable, we enable the next generation of visual analytics.

The Benefits and Costs of Semantics. While relational databases deliver excellent performance and data integrity, triple stores allow for data storage without the need of prior schema definition. Thus, triple stores are more suitable than relational databases for situations where underlying data structures and schemata are changing, and where each dataset needs its own schema.

Relational schemata are not easily extensible; adding new datasets and relationships across data requires new link tables (with the foreign keys of rows to be linked) or very generic table structures. Furthermore, changes at the schema or the data definition level need to be reflected in the applications accessing the data.

QuerioCity is not the first attempt at developing an infrastructure for managing Urban Information in IBM Ireland Research Lab. Previously, we had investigated a relational schema to model data for Dublinked. This schema quickly became complicated, as it needed to be extended with semantic types and be amenable to extensions. Moreover, providing inference capabilities in a relational infrastructure proved particularly cumbersome. We got positive feedback from experienced engineers who had not worked with semantic technologies before with regard to (i) the flexibility of not having a fixed schema, (ii) the fact that SPARQL queries did not have to be adapted when the schema would change and (iii) the fact that Semantic Web technologies are in principle compatible with enterprise systems. Furthermore, the fact that RDF references are global and can be collected from many sources is proven invaluable. In contrast, the main inhibitors reported were the uncertainty over the implications of inference to the security model and the paradigmatic shift of not being able to inspect data in a relational schema.

The main drawback with using an RDF representation lies in the sparsity of the format. We report on statistics for 4959 files in CSV format from Dublinked.

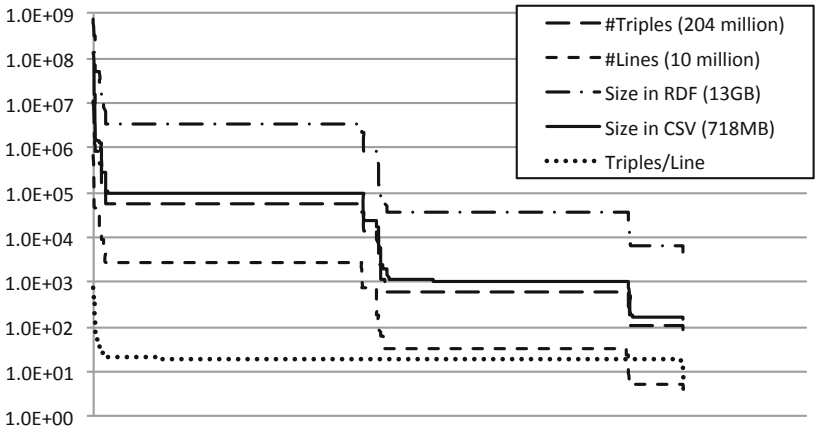


Fig. 7. Distribution of size metrics for data from Dublin. The values in parentheses represent aggregates. For readability, we have ordered values.

In Figure 7, we show the distribution of the number of triples when converted to the simple RDF format, the number of lines in the CSV files, the size of the RDF in Turtle format, the size of the input CSV files and the number of triples per CSV line. We observe that (i) the number of triples is 20 times larger than the number of lines, (ii) the size of the data in RDF format is 18 times the size of the CSV data, and (iii) the vast majority of CSV files contain between 3 and 7 columns, corresponding to 10-20 triples. Connected to this, there is also significant cost in indexing this data.

5 Conclusions and Future Work

In this paper, we presented QuerioCity, a Linked Data-based approach for managing the information of a city. The novelty of our approach lies in the flexibility of the integration, the provisioning for privacy, the efficiency of the storage model and the ability to handle streams. The main lessons learned concern tackling the domain complexity of city data and the maturity of related technologies, while the main benefits and costs of a semantic representation lie in flexibility and sparsity.

The QuerioCity platform creates an opportunity to further understand how citizens and city officials use the system and what are the typical questions they are trying to answer. Query logs obtained from users explorations and applications (SPARQL queries and consecutive HTTP requests) can be used to create models of usage to enhance the search and explorations with information more commonly sought by users, and learn how linked urban data can be exploited to answer potentially complex information needs and user queries.

In future work, we also plan to investigate an urban information management platform across cities, through the use of a federated catalog and indexes for data distributed in different cities repositories that will allow to discover, fuse and compare data and cities. Considering the complexity of the domain and the heterogeneity of the information, natural querying that scales becomes of paramount importance. With breakthroughs such as Watson allowing complex queries in natural language, we further plan to investigate methods to perform quantitative queries in an open domain.

Acknowledgments. The authors would like to thank the authorities in the Dublin area for providing datasets and NUI, Maynooth for their contribution in the development of Dublinked.

References

1. Berners-Lee, T.: Putting government data online (2009)
2. Alani, H., Dupplaw, D., Sheridan, J., O'Hara, K., Darlington, J., Shadbolt, N., Tullo, C.: Unlocking the Potential of Public Sector Information with Semantic Web Technology. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 708–721. Springer, Heidelberg (2007)

3. Maali, F., Cyganiak, R., Peristeras, V.: A Publishing Pipeline for Linked Government Data. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 778–792. Springer, Heidelberg (2012)
4. Ding, L., Lebo, T., Erickson, J.S., DiFranzo, D., Williams, G.T., Li, X., Michaelis, J., Graves, A., Zheng, J.G., Shangguan, Z., Flores, J., McGuinness, D.L., Hendler, J.: Twc logd: A portal for linked open government data ecosystems. Web Semantics (2011) (in press)
5. Sheridan, J., Tennison, J.: Linking uk government data. In: LDOW. CEUR Workshop Proceedings, vol. 628. CEUR-WS.org (2010)
6. Auer, S., Lehmann, J., Hellmann, S.: LinkedGeoData: Adding a Spatial Dimension to the Web of Data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 731–746. Springer, Heidelberg (2009)
7. Goodwin, J., Dolbear, C., Hart, G.: Geographical linked data: The administrative geography of great britain on the semantic web. Transaction in GIS 12(1), 19–30 (2009)
8. García-Silva, A., Gómez-Pérez, A., Suárez-Figueroa, M.C., Villazón-Terrazas, B.: A Pattern Based Approach for Re-engineering Non-Ontological Resources into Ontologies. In: Domingue, J., Anutariya, C. (eds.) ASWC 2008. LNCS, vol. 5367, pp. 167–181. Springer, Heidelberg (2008)
9. Pivk, A.: Automatic ontology generation from web tabular structures. AI Communications 19, 2006 (2005)
10. Hurst, M.: Layout and language: Challenges for table understanding on the web, pp. 27–30 (2001)
11. Venetis, P., Halevy, A., Madhavan, J., Paşca, M., Shen, W., Wu, F., Miao, G., Wu, C.: Recovering semantics of tables on the web. VLDB Endow. 4(9), 528–538 (2011)
12. Han, L., Finin, T., Parr, C.S., Sachs, J., Joshi, A.: RDF123: From spreadsheets to RDF. In: Sheth, A., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 451–466. Springer, Heidelberg (2008)
13. Scharffe, F., Atemezing, G., Troncy, R., Gandon, F., Villata, S., Bucher, B., Hamdi, F., Bihanic, L., Kepeklian, G., Cotton, F., Euzenat, J., Fan, Z., Vandenbussche, P.-Y., Vatant, B.: Enabling linked-data publication with the datalift platform. In (AAAI 2012) Workshop on Semantic Cities (2012)
14. Zhao, J.: Open provenance model vocabulary specification. Tech. rep., University of Oxford (2010), http://open-biomed.sourceforge.net/opmv/ns.html
15. Le-Phuoc, D., Parreira, J., Hausenblas, M., Han, Y., Hauswirth, M.: Live linked open sensor database. In: Proceedings of the 6th International Conference on Semantic Systems, I-SEMANTICS 2010, pp. 46:1–46:4. ACM, New York (2010)
16. Fung, B.C.M., Wang, K., Chen, R., Yu, P.S.: Privacy-preserving data publishing: A survey of recent developments. ACM Comput. Survey 42(4), 14:1–14:53 (2010)
17. Barbieri, D., Braga, D., Ceri, S., Della Valle, E., Grossniklaus, M.: Querying rdf streams with c-sparql. SIGMOD Record 39(1), 20–26 (2010)

Semantic Similarity-Driven Decision Support in the Skeletal Dysplasia Domain

Razan Paul[1], Tudor Groza[1], Andreas Zankl[2,3], and Jane Hunter[1]

[1] School of ITEE, The University of Queensland, Australia
{razan.paul,tudor.groza}@uq.edu.au, jane@itee.uq.edu.au
[2] Bone Dysplasia Research Group
UQ Centre for Clinical Research (UQCCR)
The University of Queensland, Australia
[3] Genetic Health Queensland,
Royal Brisbane and Women's Hospital, Herston, Australia
a.zankl@uq.edu.au

Abstract. Biomedical ontologies have become a mainstream topic in medical research. They represent important sources of evolved knowledge that may be automatically integrated in decision support methods. Grounding clinical and radiographic findings in concepts defined by a biomedical ontology, e.g., the Human Phenotype Ontology, enables us to compute semantic similarity between them. In this paper, we focus on using such similarity measures to predict disorders on undiagnosed patient cases in the bone dysplasia domain. Different methods for computing the semantic similarity have been implemented. All methods have been evaluated based on their support in achieving a higher prediction accuracy. The outcome of this research enables us to understand the feasibility of developing decision support methods based on ontology-driven semantic similarity in the skeletal dysplasia domain.

1 Introduction

Similarity plays a central role in medical knowledge management. Like most scientific knowledge, medical knowledge is also inferred from comparing different concepts (such as phenotypes, populations, and species) and analyzing their similarities and differences. However, medical science is unlike other sciences in that its knowledge can seldom be reduced to a mathematical form. Thus, medical scientists usually record their knowledge in free form text, or lately in biomedical ontologies. New concepts that emerge in the domain are firstly compared and judged based on their degree of similarity to existing concepts before being integrated into the overall domain knowledge.

Biomedical ontologies are knowledge bases that have emerged and evolved over time following this process. Most of them are used not only to model and capture specific domain knowledge, but also to annotate, and hence enrich, diverse resources like patient cases or scientific publications. The adoption of biomedical ontologies for annotation purposes provides a means for comparing medical

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 164–179, 2012.

concepts on aspects that would otherwise be incomparable. For example, the annotation of a set of disorders (directly or via patient cases) using the same ontology enables us to compare them, by looking at the underpinning annotation concepts. The actual comparison is subject to a semantic similarity measure, i.e., a function that takes two or more ontology concepts and returns a numerical value that reflects the degree of similarity between these concepts.

Over the course of the last decade, there has been significant research performed on semantic similarities over biomedical ontologies. One key remark that needs to be taken into account is that meaningful similarity measures are dependent on the domain knowledge, as only by using the explicit semantics of the domain one can compare concepts in an appropriate manner. In this paper we report on our experiences with using semantic similarity over domain knowledge and annotated patient cases for disorder prediction in the skeletal dysplasia domain.

Skeletal dysplasias are a group of heterogeneous genetic disorders affecting skeletal development. There are currently over 450 recognised bone dysplasias, structured into 40 groups. Patients suffering from such disorders have complex medical issues, ranging from bowed arms and legs to neurological complications. Since most dysplasias are very rare ($<$ 1:10,000 births), data on clinical presentation, natural history and best management practices is very sparse. A different perspective on data sparseness is introduced also by the small number of clinical and radiographic phenotypes typically exhibited by patients from the vast range of possible characteristics globally associated with these disorders.

Decision support methods can usually assist clinicians and researchers both in the research, as well as in the decision making process, in general, in any domain. However, building efficient or meaningful decision support methods in a domain affected by data sparseness, such as bone dysplasias, is a very challenging task. On the other hand, semantic similarity measures can facilitate the objective interpretation of clinical and radiographic findings by using knowledge captured in biomedical ontologies or annotated patient cases to provide decision support. In this paper we aim to bridge the two worlds, by investigating different approaches for determining the semantic similarity between sets of phenotypes encoded as ontological concepts and its application to disorder prediction.

The context of our work is provided by the SKELETOME project that develops a community-driven knowledge curation platform for the bone dysplasia domain [1]. The underlying foundation of the platform is a two-phase knowledge engineering cycle which enables: (1) semantic annotation of patient cases – connecting domain knowledge to real-world cases; and (2) collaborative diagnosis, collaborative knowledge curation and evolution – evolving the domain knowledge, based on real-world cases. The semantic annotation process relies on clinical and radiographic findings grounded in the Human Phenotype Ontology (HPO) [2] – an emerging de facto standard for capturing, representing and annotating phenotypic features encountered in rare disorders. At the same time, the domain knowledge is modeled via the Bone Dysplasia Ontology (BDO) [3], which

at a conceptual level associates bone dysplasias and phenotypes represented by HPO terms.

These two sources of knowledge, i.e., domain knowledge from BDO and raw knowledge from annotated patient cases, together with the structure of HPO, which underpins the formalization of phenotypes, enable us to investigate the use of several semantic similarity measures in order to achieve disorder prediction. More concretely, this paper: (i) analyzes which semantic similarity performs better on each of the two types of data, and (ii) performs an extensive empirical evaluation of the application of these semantic similarities for disorder prediction, using a real-world dataset.

The remainder of the paper is structured as follows: Section 5 discusses existing related work, Section 2 provides a comprehensive background on the knowledge and data sources used within our experiments, while Section 3 details our methodology. Before concluding in Section 6 we present an extensive evaluation and discuss the experimental results in Section 4.

2 Background

This section provides a brief overview of the background of our work. It introduces the Human Phenotype Ontology (HPO) and discusses some of its characteristics (Section 2.1), then describes the two knowledge sources used in our experiments, i.e., the Bone Dysplasia Ontology and the largest bone dysplasia patient dataset (Section 2.2) and finally, presents briefly some of the most commonly used similarity measures (Section 2.3).

2.1 Human Phenotype Ontology

The Human Phenotype Ontology [1] is a controlled vocabulary that captures and represents clinical and radiographic findings (or phenotypes in general), in principle, in hereditary diseases listed in Online Mendelian Inheritance in Man (OMIM) database [2]. The ontology consists of around 9,000 concepts describing modes of inheritance, onset and clinical disease courses and phenotypic abnormalities. This last category represents around 95% of the ontology and is the main subject of our study. Phenotypic abnormalities are structured in a hierarchical manner (via class–subclass relationships) from generic (e.g., `HP_0000929` – *Abnormality of the skull*) to specific abnormalities (e.g., `HP_0000256` – *Macrocephaly*).

One aspect that needs to considered when using the structure of HPO is the multiple inheritance. All children of a particular class share some information (which is logical in a typical ontology), however, the type of this shared information (i.e., not the specific information) can be different. More concretely, abnormalities may share their anatomical localization or they may share the

[1] http://www.human-phenotype-ontology.org/
[2] http://www.omim.org/

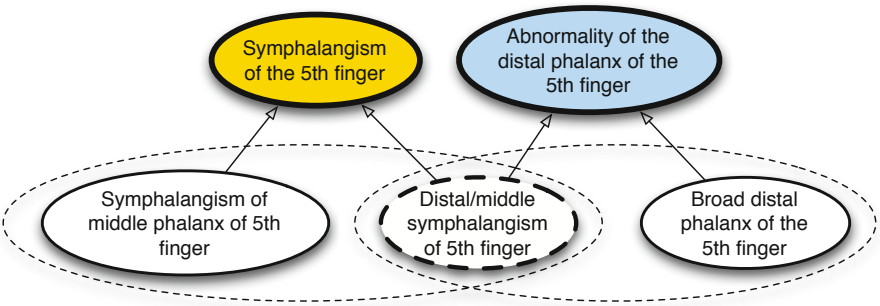


Fig. 1. An example of multiple inheritance in HPO (arrows denote class–subclass relations)

intrinsic type of abnormality. Fig. 1 depicts an example of such multiple inheritance. `HP_0009244` (*Distal/middle symphalangism of 5th finger*) is a sibling of `HP_0009178` (*Symphalangism of middle phalanx of 5th finger*) – they represent the same type of abnormality, i.e., *Symphalangism*, and hence are both children of `HP_0004218` (*Symphalangism of the 5th finger*), but also a sibling of `HP_0009240` (*Broad distal phalanx of the 5th finger*) – they share the anatomical localization of the abnormality, and hence are both children of `HP_0004225` (*Abnormality of the distal phalanx of the 5th finger*). This remark is important because it influences the computation of the most specific common ancestor for two concepts, a central element of most semantic similarity measures.

2.2 Bone Dysplasia Knowledge Sources

As mentioned in Section 1, in the context of the SKELETOME project, we have two major knowledge sources: the Bone Dysplasia Ontology (BDO) [3] and a set of semantically annotated patient cases. The clinical and radiographic findings that characterize both are underpinned by the Human Phenotype Ontology.

BDO has been developed to model and capture essential (and mature) knowledge in the skeletal dysplasia domain. As depicted in Fig. 2, it associates bone dysplasias to gene mutations and phenotypic characteristics, which are then further specialised via concepts defined by external ontologies, such as HPO. In [3] we provide a comprehensive overview of the design process of BDO. With respect to the work described in this paper, there is one remark that is worth being noted. BDO describes associations (via class axioms) between more than 250 disorders (out of the 450 in total) and around 2,000 findings (represented by HPO concepts). These associations have been created from the clinical synopses of the corresponding disorders in OMIM and represent, in principle, the current state of *conceptual* understanding of their clinical manifestations. As a result, the phenotypic findings listed there are a mixture of more generic (e.g., *abnormal*

[3] `http://purl.org/skeletome/bonedysplasia`

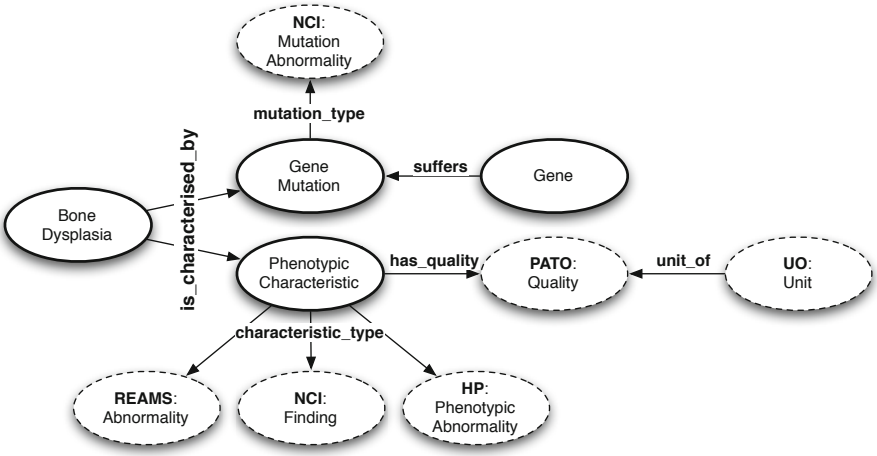


Fig. 2. A snapshot of the Bone Dysplasia Ontology from [1]

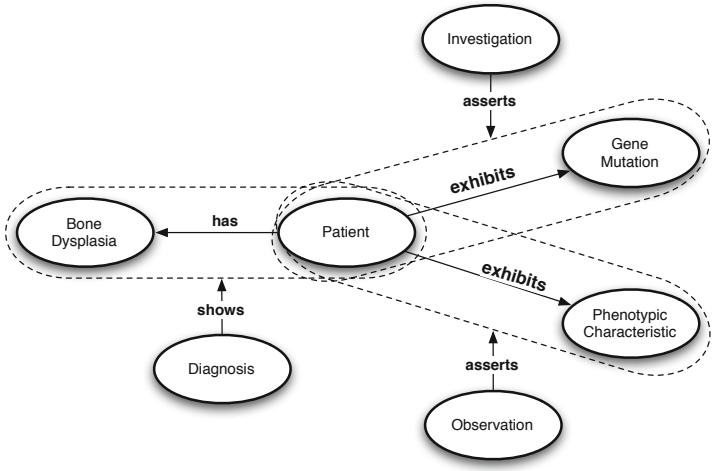


Fig. 3. A snapshot of the Patient Ontology from [1]

femoral neck) and fairly specific (e.g., *short, broad femoral neck*) terms. This reflects the balance achieved by capturing both the clinical interpretation of sets of patient cases (for the more common disorders), as well as singular or particular patient cases (for those that are extremely rare).

In addition to the domain knowledge, SKELETOME focuses also on capturing instance data, i.e., annotated patient cases. The actual modeling is done via the Patient Ontology (depicted in Fig. 3), which associates patients to clinical and radiographic findings, gene mutations and bone dysplasias. The main

source of patient data is the registry of the European Skeletal Dysplasia Network (ESDN) [4], which is a pan-European research and diagnostic network aimed to provide community driven help and diagnostic expertise for rare bone disorders. Our current dataset comprises a total of 1,200 semantically annotated closed ESDN cases. Each patient case has been modeled using the Patient Ontology and captures HPO concepts denoting clinical and radiographic findings and BDO dysplasias denoting the final diagnosis. In contrast to the knowledge in BDO, the level of specificity present in the clinical descriptions is, as expected, fairly high, i.e., the general tendency is to find more specific findings rather than more generic ones.

2.3 Semantic Similarity

As mentioned earlier, there has been a great amount of research done on semantic similarities. Here, we intend only to introduce some basic concepts and to provide a brief overview of the measures used within our experiments. A detailed survey on semantic similarity on biomedical ontologies can be found in [4].

There are two main types of semantic similarities: (1) node-based similarities and (2) edge-based similarities. The former uses nodes and their properties as information source, whereas the latter focuses on edges and their types.

Node based approaches rely on the notion of Information Content (IC) to quantify the informativeness of a concept. IC values are usually calculated by associating probabilities to each concept in ontology by computing the negative likelihood of its frequency in large text corpora. The basic intuition behind the use of the negative likelihood in the IC calculation is that the more probable the presence of a concept in a corpus is, the less information it conveys. IC is expressed in Eq. 1, with $p(c)$ being the probability of occurrence of c in a specific corpus. In our case $p(c)$ represents the probability of occurrence of an HPO concept in the context of a bone dysplasia, either from the domain knowledge, or from the raw patient cases.

The foundational node based similarity measures are Resnik [5], Lin [6] and Jiang and Conrath [7]. Resnik was the first to leverage IC for computing semantic similarity and expressed semantic similarity between two terms as the IC of their most informative common ancestor (MICA – Eq. 2). The intuition is that similarity depends on the amount of information two concepts, c_1 and c_2, share. This, however, does not consider how distant the terms are in their information content and from a hierarchical perspective. Consequently, Lin (Eq. 3) and Jiang and Conrath (Eq. 4) have proposed variations of Resnik's similarity to take into account these aspects.

$$IC(c) = -logp(c) \tag{1}$$

$$sim_{Res}(c_1, c_2) = IC(c_{MICA}) \tag{2}$$

[4] http://www.esdn.org

$$sim_{Lin}(c_1, c_2) = \frac{2 * IC(c_{MICA})}{IC(c_1) + IC(c_2)} \tag{3}$$

$$sim_{JC}(c_1, c_2) = 1 - IC(c_1) + IC(c_2) - 2 * IC(c_{MICA}) \tag{4}$$

Edge-based approaches take into account the paths existing between the concepts in the ontology. Subject to the domain and ontology, such paths could be considered by following *is-a* relationships (the most common approach) or other types of relationships defined by the ontology. Examples of such similarity measures include: (i) Wu & Palmer [8] (Eq. 5), where LCS is the least common subsumer of c_1 and c_2 and N_1 is the length of the path from c_1 to root, N_2 the length of the path from c_2 to root and N_3 the length of the path from LCS to root; or (ii) Leacock-Chodorow [9] (Eq. 6), where D is the overall depth of the ontology. A more recent measure has been described in [10] and considers, among other aspects, the number of changes in direction of the shortest path between two concepts (i.e., how many times on the shortest path the traversing direction changes from child to parent and vice-versa).

$$sim_{W\&P}(c_1, c_2) = \frac{2 * N_3}{N_1 + N_2 + 2 * N_3} \tag{5}$$

$$sim_{L\&C}(c_1, c_2) = -log \frac{len(c_1, c_2)}{2 * D} \tag{6}$$

A third category of similarity measures could be considered for the hybrid approaches, i.e., combining node and edge-based similarities (e.g., [11] or [12]). Our work aims to integrate both information content and structural relationships in order to gain as much as possible from the semantics provided by HPO. As described in the following section, we also propose a series of such hybrid measures tailored on specific requirements emerged from our knowledge sources.

3 Methodology

The goal of our work is to predict disorders given an annotated patient case description. More concretely, given a background knowledge base (i.e., BDO or the annotated patient dataset) and a set of HPO concepts (representing clinical and radiographic findings of a new patient case), we aim to predict the most plausible bone dysplasias, ranked according to their probability. This is a typical multi-class classification problem, however, due to data sparseness that characterises the skeletal dysplasia domain, typical Machine Learning algorithms achieved a very low accuracy [5]. Our intuition is that by using semantic similarity measures on patient findings (i.e., HPO concepts) we are able to leverage and use intrinsic

[5] A series of classification experiments we have performed revealed a maximal accuracy of around 35% for Naive Bayes, in a setting in which we have considered only six disorders, i.e., those that had more than 20 cases.

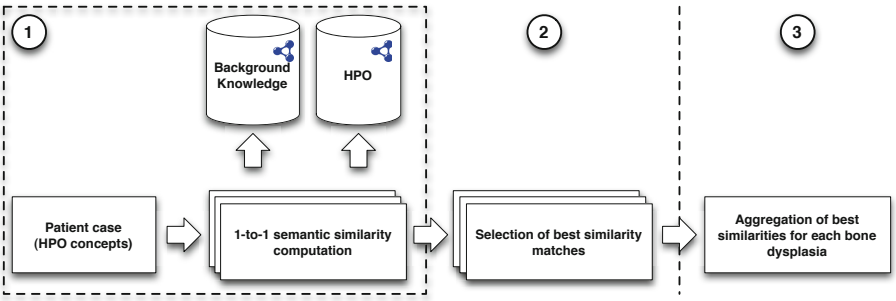


Fig. 4. Block diagram of the prediction methodology

associations between phenotypes that cannot, otherwise, be acquired by typical Machine Learning methods (due to their term-based matching process). As an example, if the background knowledge base lists `HP_0000256` (*Macrocephaly*) as a phenotype of `Achondroplasia` and a new patient exhibits `HP_0004439` (*Craniofacial dysostosis*) we want to use the semantic similarity between `HP_0000256` and `HP_0004439` to also associate the later to `Achondroplasia` with a certain probability[6]. The semantic similarity between the two concepts could be inferred via their most common ancestor `HP_0000929` (*Abnormality of the skull*). Such an association is not possible when employing typical Machine Learning methods since each term would be considered individually and only in the context provided by the background knowledge base.

Fig. 4 depicts the overall methodology. In the first step, we compute the semantic similarity between all HPO concepts representing clinical and radiographic findings of the given patient case and all phenotypes associated with bone disorders in the background knowledge base (please note that we do not make any assumptions about the background knowledge base). If we consider $\{S_1, S_2, \ldots, S_n\}$ to be patient findings and $\{P_1, P_2, \ldots, P_n\}$ phenotypes of bone dysplasia D, the best similarity match between S_i and D is given by:

$$BestMatch(S_i, D) = \underset{j=1}{\overset{n}{\operatorname{argmax}}}\{sim(S_i, P_j)\} \tag{7}$$

The semantic similarity in Eq. 7 can be any of the classical similarities mentioned in Section 2 or, for example, one of the measure we introduce later in this section. The evaluation described in Section 4 has been performed on multiple such similarity measures. Once the best matches have been computed, we calculate the final probability by aggregating them:

$$P(S_1, S_2, \ldots, S_n|D) = \frac{1}{n} * \sum_{i=1}^{n} BestMatch(S_i, D) \tag{8}$$

[6] As a remark, there is no direct relationship in HPO between the concepts `HP_0000256` and `HP_0004439`. A relationship exists only via the parent of `HP_0000256` (i.e., `HP_0000240` – *Abnormality of skull size*), which is a sibling of `HP_0004439`.

As mentioned previously, a good semantic similarity measure needs to take into account the specific aspects of the target domain. Below we have summarized a series of requirements for the similarity measure that have emerged from the bone dysplasia domain and the structure of HPO:

- Given two HPO concepts and their LCA (lowest common ancestor), we consider the concept closer to the LCA to be more similar to the LCA than the concept located at a bigger distance. E.g., `HP_0004439` (*Craniofacial dysostosis*) will be considered more similar to `HP_0000929` (*Abnormality of the skull*) than `HP_0000256` (*Macrocephaly*), because it is a direct descendent of `HP_0000929`;
- The information content of an LCA is dependent on its specificity (i.e., its location in the overall hierarchy). More concretely we consider the more specific LCA to be more informative. E.g., `HP_0004439` (*Craniofacial dysostosis*) (as an LCA) should be considered more informative than `HP_0000929` (*Abnormality of the skull*), which is in this case, its direct parent.
- A smoothing parameter may be required to deal with missing LCA information content. As described in [12], one of the main issues of IC is that its values are derived by analyzing large corpora (in our case a given background knowledge base), which may not even contain certain concepts. This is also the case with LCAs computed on certain pairs of findings, aspect dependent on the background knowledge base. Unfortunately, neither the intrinsic information content defined in [13], nor the extended information content defined in [12] can be employed in our domain, because we need the IC of a concept to be defined in the context of a given disorder (see below) and not only based on its children or surrounding concepts in the ontology. In other terms, we cannot use only the local IC definition provided by HPO without the scope provided by and associated disorder.

In addition to these requirements, we need to define the Information Content (IC) of a finding in the context of a disorder. Independently of the background knowledge base used for experiments, we have considered $IC(C_P)$ (i.e., the IC of the concept C grounding phenotype P) to be:

$$IC(P) = -log\frac{N_{DP}}{N_D} \tag{9}$$

where N_{DP} represents the number of disorders associated with P and N_D is the total number of disorders.

In the following we define a series of hybrid semantic similarities that take into account the above listed requirements.

HSS1 quantifies the semantic similarity between concepts according to the information content of the LCA and the position of LCA in regards to the concepts. HSS1 neglects the specificity of LCA.

$$HSS1(C_1, C_2) = \frac{Any - Node - Based - Similarity}{DIST(C_1, LCA) + DIST(C_2, LCA)} \tag{10}$$

where $DIST(C,C) = 0$ and $DIST(C_1,C_2) = len(SPath(C_1,C_2))$ (SPath = shortest path).

For example, HSS1 used with Resnik (sim_{Res}) would be:

$$HSS1(C_1,C_2) = \frac{IC(LCA)}{DIST(C_1,LCA) + DIST(C_2,LCA)} \tag{11}$$

HSS2 introduces the specificity of LCA, however, it neglects the missing LCA information content. HSS2 is defined below.

$$HSS2(C_1,C_2) = \frac{L}{D} * HSS1 \tag{12}$$

where L is the length of the path from the root to LCA and D is the depth of the ontology.

HSS3 and HSS4. In order to fulfil the last requirement, we have experimented with two additional measures (HSS3 and HSS4 defined below), that introduce different smoothing parameters: HSS3 uses a constant K, where $K = 1/N_D$ (N_D = total number of disorders), while HSS4 considers a joint information content of the two concepts.

$$HSS3(C_1,C_2) = \frac{\frac{L}{D} * (IC(LCA) + K)}{DIST(C_1,LCA) + DIST(C_2,LCA)} \tag{13}$$

$$HSS4(C_1,C_2) = \frac{\frac{L}{D} * (IC(LCA) + \frac{IC(C_1) * IC(C_2)}{IC(C_1) + IC(C_2)})}{DIST(C_1,LCA) + DIST(C_2,LCA)} \tag{14}$$

4 Experimental Results

Taking into account the context provided by the SKELETOME project, i.e., a platform used by clinicians, we have tested the disorder prediction on a subset of the patient dataset described in Section 2. We performed three different experiments, described in the following:

- Firstly, we used a part of the patient dataset as knowledge source,
- Secondly, we used the Bone Dysplasia Ontology as knowledge source,
- Thirdly, we compared the semantic similarity-based prediction against a term matching-based prediction (i.e., an approach that uses only the frequency of the patient findings in the context of each disorder).

Each experiment tested different semantic similarity measures (applied in a HPO concept to concept setting). To assess the efficiency provided by the semantic similarity, we have calculated the overall accuracy of the disorder prediction. Node-based similarities have used the information content calculated on the background knowledge used in the experiment (i.e., IC on BDO or on patient

Table 1. Experimental results of disorder prediction using patient cases as background knowledge

Similarity	A@1 (%)	A@2 (%)	A@3 (%)	A@4 (%)	A@5 (%)
Resnik	10.96	21.92	32.88	35.62	41.10
Lin	6.85	12.33	17.81	28.77	34.25
J&C	2.74	9.59	12.33	13.70	20.55
HSS1	31.51	46.58	54.79	64.38	71.23
HSS2	32.87	49.32	56.16	64.38	69.86
HSS3	**39.73**	**52.05**	**61.64**	**69.86**	**75.34**
HSS4	39.73	52.05	60.27	68.49	73.97

cases), while the hybrid similarities have used both this information content and the structure of HPO.

In Section 2 we have discussed some of the foundational differences between the two knowledge sources with respect to the phenotypes' specificity. Another aspect that needs to be mentioned is that, since the raw knowledge we are using emerges from real patient cases, it will contain clinical and radiographic features that are directly related to the disorder, but also phenotypes that are not necessarily relevant. This is a normal phenomenon, because clinicians record all their findings before considering a diagnosis. For example, a clinical summary may contain findings such as, *bowed legs*, *macrocephaly* and *cleft palate*, which are relevant for the final *Achondroplasia* diagnosis, but it may also contain *fractured femur* and *decreased calcium level*, which are not relevant in the context of the final diagnosis. The set of unrelated findings are termed as noise.

Noise is the one of the most important contributing factors to the prediction accuracy, and it is inverse proportional to it. Hence, the prediction accuracy depends on the noise introduced both by the background knowledge, as well as the test data. As we are considering both the domain knowledge (via BDO) and patient cases as background knowledge bases in two different assessments, we will be able to judge which of the two types of knowledge contains more noise. This is realized by testing both on the set test dataset and comparing the resulted accuracy.

In all experiments detailed below we compute the prediction accuracy as the overall percentage of correctly predicted disorders at a given recall cut-off point (i.e., by taking into account only the top K predictions, for different values of K, where K is the recall cut-off point). Hence, a success represents correctly predicted disorder (the exact same, and not a sub or super class of it), while a miss represents an incorrectly predicted disorder. If N is the total number of test cases and L is the number of corrected predicted disorders, then Accuracy $A = L/N$. This is expressed in percentages in Tables 1, 2 and 3.

Table 2. Experimental results of disorder prediction using BDO as background knowledge

Similarity	A@1 (%)	A@2 (%)	A@3 (%)	A@4 (%)	A@5 (%)
Resnik	2.74	4.10	4.10	6.84	8.21
Lin	1.37	2.74	2.74	4.10	4.10
J&C	0	0	0	0	0
HSS1	16.43	21.91	32.87	43.84	**47.95**
HSS2	10.96	16.43	17.80	24.66	27.40
HSS3	10.96	16.44	19.18	23.29	27.40
HSS4	10.96	17.80	19.18	21.92	28.77

4.1 Experiment 1: Patient Data as Knowledge Base

This first experiment considers patient cases as background knowledge. As discussed in Section 2, we collected a dataset of 1,200 patient cases from ESDN and annotated them with HPO terms. In order to provide an accurate view over the prediction, the experiment has been performed as a 5-fold cross validation with an 80-20 split (80% knowledge base, 20% test data). Table 1 lists the resulted average accuracy at five different recall cut-off points.

Overall, HSS3 has performed the best in this experiment, more or less on par with HSS4, and has confirmed that it is important for all three requirements listed in Section 3 to be fulfilled. Moreover, this experiment shows the improvement brought by a hybrid method over traditional information content based approaches. HSS1 outperforms the IC-based similarities because it considers the distance to the LCA and not only the IC of the LCA – i.e., the closer the two terms are to the LCA (and implicitly between them) the more similar they are. At the same time, HSS3 outperformed HSS1 because it smooths the missing information content, while at the same time introducing the specificity (L/D) – which is characteristic to the background knowledge. Finally, the similarity between HSS3 and HSS4 (that can also be observed in experiment 2) shows that the parameter $K = 1/N_D$ is a good approximation of the joint information content of the two concepts.

4.2 Experiment 2: BDO as Knowledge Base

The second experiment evaluated the disorder prediction with BDO as background knowledge. We have performed the same rounds of experiments as in the first case, i.e., we tested the prediction accuracy for the exact same 5 test folds resulted from experiment 1 and computed the final average accuracy for each semantic similarity. Results are listed in Table 2.

As in the case of the first experiment, all hybrid similarities outperformed the classical information content approaches. This time, however, HSS1 has achieved the best result, proving that the HPO concepts captured by BDO are more generic, as we have expected. The specificity factor L/D in HSS2, HSS3 and

Table 3. Experimental results on term matching vs. semantic similarity

Method	A@1 (%)	A@2 (%)	A@3 (%)	A@4 (%)	A@5 (%)
Patient cases as background knowledge					
Term matching	26.02	38.36	50.68	56.16	61.64
Semantic similarity	39.73	52.05	61.64	69.86	**75.34**
BDO as background knowledge					
Term matching	8.21	15.06	21.91	26.02	27.4
Semantic similarity	16.43	21.91	32.87	43.84	**47.95**

HSS4 takes low values because L is generally smaller (i.e., terms are located higher in the hierarchy and hence more generic) which leads to smaller values for these measures. This is also the reason why, the same similarities have performed worse when BDO was considered background knowledge, as opposed to using patient cases as background knowledge. The specificity of the ancestor improves the accuracy on patient cases but it decreases it on domain knowledge. Finally, a different reason for the lower accuracy is the multiple inheritance used in HPO, which leads to additional missing information content for LCAs.

4.3 Experiment 3: Term Matching vs. Semantic Similarity

Finally, in order to gain insight in the importance of using semantic similarity measures in disorder prediction, we have compared the results of the best performing similarity for each background knowledge against prediction calculated on term-based matching. Firstly, the results listed in Table 3 show that using semantic similarity is generally a good strategy as the overall accuracy is improved when compared to term-based matching, independently of the background knowledge. Interestingly, in this comparison, the specificity factor that heavily influences the accuracy based on the background knowledge has proved to be beneficial in the context of BDO, when compared against term matching. Secondly, returning to the comparison based on background knowledge, we can conclude that the domain knowledge introduces more noise than patient cases, which seems to contradict our initial belief (since clinicians will list in a case all observed findings, including those that may turn out to be irrelevant for the final diagnosis). In reality, in this case we are dealing with a different kind of noise, as the domain knowledge has the tendency to dilute the discriminatory findings when aggregating the information resulted from analyzing groups of patients. We intend to deal with this issue by including knowledge on differential diagnosis in the Bone Dysplasia Ontology.

5 Related Work

The research presented in [14] is the most relevant related work in the context of this paper. Kohler et al. have developed a semantic similarity search applica-

tion named Phenomizer, which takes as input a set of HPO terms and returns a ranked list of diseases from OMIM, to their semantic similarity values. Phenomizer uses the Resnik semantic similarity and arithmetic mean as aggregation strategy (similar to our approach). According to the experiments discussed in the paper, their solution outperforms term-based matching approaches that do no consider any relationships between terms. Our research follows closely the work done in Phenomizer, however, we use real patient data to test the disorder prediction (as opposed to the synthetically generated data in their case), and we try to tailor the semantic similarity to map onto the requirements emerging from the domain. Furthermore, we test several semantic similarities in order to get a better understanding of the most appropriate combination that serves our prediction goal. Finally, we evaluate the prediction accuracy using two types of background knowledge – domain and raw knowledge, as opposed to only domain knowledge in their case.

Additional related work includes [15], where the authors use a threshold of lowest semantic similarity value to find best-matching term pairs with the goal of predicting molecular functions of genes in Gene Ontology (GO) [16] annotations. Similar to our work, the authors tailor the semantic similarity measures according to fit the structure of GO and their application requirements. Lei et al. [17] assess protein similarity within GO to predict the subnuclear location. They compared the prediction accuracy of several similarity measures, including classical ones such as Resnik, and term-based matching to find insignificant differences between them. The authors also evaluate several aggregation strategies for the similarity values (e.g., sum, average, multiplication) and have found that the sum of the term-based matching method produces the best predictive outcome. Subnuclear location of a gene is associated with specific GO terms in most of the cases. As a result, using the hierarchical structure of the ontology via semantic similarity methods may not bring significant improvements.

In [18], the authors use ontological annotations and a proposed semantic similarity measure to find a correlation between protein sequence similarity and semantic similarity across GO. Similarly, Washington et al. [19] investigate the ontological annotation of disease phenotypes and the application of semantic similarities to discover new genotype-phenotype relationships within and across species. Finally, Ferreira et al. [20] use semantic similarity measures to classify chemical compounds and have showed that employing such techniques improves the chemical compound classification mechanisms. To achieve this, they employed measures tailored on the semantics of the Chemical Entities of Biological Interest Ontology (ChEBI).

6 Conclusion

In this paper we have reported on our experiences in using semantic similarity measures for disorder prediction in the skeletal dysplasia domain. The SKELETOME project provides two types of knowledge sources: (1) domain knowledge, modeled by and captured in the Bone Dysplasia Ontology and (2) raw knowledge

emerging from patient cases. In both cases the clinical and radiographic findings are grounded in Human Phenotype Ontology concepts. The data sparseness that characterises this domain required us to consider alternative approaches in performing disorder prediction. Hence, we took advantage of the semantics provided by HPO and experimented with different semantic similarity measures, using both types of knowledge sources.

The experimental results have led to the conclusion that applying only information theoretic approaches in computing semantic similarity over the Human Phenotype Ontology, in our domain, does not provide the optimum result. Instead, we need to take into account particular requirements that emerge from the data characteristic to the bone dysplasia domain, i.e., a combined path between findings and their common ancestor, the specificity of this common ancestor and a smoothing parameter for the cases when the information content of the common ancestor is missing. Another conclusion of our experiments has been the need for differential diagnosis information in the domain knowledge in order to increase the weight of the discriminatory findings. Finally, we have shown that using semantic similarities improved the prediction accuracy when compared to term-based (frequency) matching prediction.

Acknowledgments. The work presented in this paper is supported by the Australian Research Council (ARC) under the Discovery Early Career Researcher Award (DECRA) – DE120100508 and the Linkage grant SKELETOME – LP100100156.

References

1. Groza, T., Zankl, A., Li, Y.-F., Hunter, J.: Using Semantic Web Technologies to Build a Community-Driven Knowledge Curation Platform for the Skeletal Dysplasia Domain. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 81–96. Springer, Heidelberg (2011)
2. Robinson, P.N., Kohler, S., Bauer, S., Seelow, D., Horn, D., Mundlos, S.: The Human Phenotype Ontology: A Tool for Annotating and Analyzing Human Hereditary Disease. The American Journal of Human Genetics 83(5), 610–615 (2008)
3. Groza, T., Hunter, J., Zankl, A.: The Bone Dysplasia Ontology: integrating genotype and phenotype information in the skeletal dysplasia domain. BMC Bioinformatics 13(50) (2012)
4. Pesquita, C., Faria, D., Falcao, A., Lord, P., Couto, F.: Semantic Similarity in Biomedical Ontologies. PLoS Computational Biology 5(7) (2009)
5. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: Proc. of the 14th IJCAI, pp. 448–453 (1995)
6. Lin, D.: An information-theoretic definition of similarity. In: Proc. of the 15th ICML, pp. 296–304 (1998)
7. Jiang, J., Conrath, D.: Semantic similarity based on corpus statistics and lexical taxonomy. In: Proc. of the 10th Conf. on Research on Comp, Linguistics, Taiwan (1997)

8. Wu, Z., Palmer, M.: Verb semantics and lexicon selection. In: Proc. of the 32nd ACL, pp. 133–138 (1994)
9. Chodorow, M., Leacock, C.: Combining local context and WordNet similarity for word sense identification. Fellbaum, 265–283 (1997)
10. Schickel-Zuber, V., Faltings, B.: OSS: A Semantic Similarity Function based on Hierarchical Ontologies. In: Proc. of the 20th IJCAI, pp. 551–556 (2007)
11. Li, Y., Bandar, Z., McLean, D.: An approach for measuring semantic similarity between words using multiple information sources. ITEE Transactions on Knowledge and Data Engineering 15(4), 871–882 (2003)
12. Pirró, G., Euzenat, J.: A Feature and Information Theoretic Framework for Semantic Similarity and Relatedness. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 615–630. Springer, Heidelberg (2010)
13. Seco, N., Veale, T., Hayes, J.: An Intrinsic Information Content measure for Semantic Similarity in WordNet. In: Proc. of ECAI 2004, pp. 1089–1090 (2004)
14. Kohler, S., Schulz, M.H., Krawitz, P., Bauer, S., Dolken, S., Ott, C.E., Mundlos, C., Horn, D., Mundlos, S., Robinson, P.N.: Clinical diagnostics in human genetics with semantic similarity searches in ontologies. The American Journal of Human Genetics 85(4), 457–464 (2009)
15. Tao, Y., Sam, L., Li, J., Friedman, C., Lussier, Y.A.: Information theory applied to the sparse gene ontology annotation network to predict novel gene function. Bioinformatics 23(13), i529–i538 (2007)
16. Berardini, T.Z., et al.: The Gene Ontology in 2010: extensions and refinements. Nucleic Acids Research 38, D331–D335 (2010)
17. Lei, Z., Dai, Y.: Assessing protein similarity with Gene Ontology and its use in subnuclear localization prediction. BMC Bioinformatics 7(1), 491 (2006)
18. Lord, P.W., Stevens, R.D., Brass, A., Goble, C.A.: Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. Bioinformatics 19(10), 1275–1283 (2003)
19. Washington, N.L., Haendel, M.A., Mungall, C.J., Ashburner, M., Westerfield, M., Lewis, S.E.: Linking human diseases to animal models using ontology-based phenotype annotation. PLoS Biology 7(11) (2009)
20. Ferreira, J.D., Couto, F.M.: Semantic similarity for automatic classification of chemical compounds. PLoS Computational Biology 6(9) (2010)

Using SPARQL to Query BioPortal Ontologies and Metadata

Manuel Salvadores, Matthew Horridge, Paul R. Alexander, Ray W. Fergerson, Mark A. Musen, and Natalya F. Noy

Stanford Center for Biomedical Informatics Research
Stanford University, US
{manuelso,matthew.horridge,palexander,
ray.fergerson,musen,noy}@stanford.edu

Abstract. BioPortal is a repository of biomedical ontologies—the largest such repository, with more than 300 ontologies to date. This set includes ontologies that were developed in OWL, OBO and other languages, as well as a large number of medical terminologies that the US National Library of Medicine distributes in its own proprietary format. We have published the RDF based serializations of all these ontologies and their metadata at sparql.bioontology.org. This dataset contains 203M triples, representing both content and metadata for the 300+ ontologies; and 9M mappings between terms. This endpoint can be queried with SPARQL which opens new usage scenarios for the biomedical domain. This paper presents lessons learned from having redesigned several applications that today use this SPARQL endpoint to consume ontological data.

Keywords: Ontologies, SPARQL, RDF, Biomedical, Linked Data.

1 SPARQL In Use In BioPortal: Overview of Opportunities and Challenges

Ontology repositories act as a gateway for users who need to find ontologies for their applications. Ontology developers submit their ontologies to these repositories in order to promote their vocabularies and to encourage inter-operation. In biomedicine, cultural heritage, and other domains, many of the ontologies and vocabularies are extremely large, with tens of thousands of classes.

In our laboratory, we have developed BioPortal, a community-based ontology repository for biomedical ontologies [11]. Users can publish their ontologies to BioPortal, submit new versions, browse the ontologies, and access the ontologies and their components through a set of REST services. BioPortal provides search across all ontologies in its collection, a repository of automatically and manually generated mappings between classes in different ontologies, ontology reviews, new term requests, and discussions generated by the ontology users in the community. BioPortal contains metadata about each ontology and its versions as well as mappings between terms in different ontologies.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 180–195, 2012.

We had numerous requests from users to open a public SPARQL endpoint, which would enable them to query and analyze the data and metadata in much more fine-ground and application-specific ways than our set of REST APIs allowed. In December 2011, we released `sparql.biooontology.org` to provide direct access to ontology content, metadata and mappings. We describe the details of the structure of the dataset and how it implements the Linked Data principles elsewhere [17] and provide a short overview here in Section 2.

In the first months of the endpoint deployment, we have received valuable feedback from our user community and were able to identify some points of continuous debate around the use of `sparql.bioontology.org`. In this paper, we address the following three points that both proved challenging and provided a clear use case for the advantages of SPARQL:

- Retrieval of common attributes from multiple ontologies: BioPortal's ontologies, with 300 ontologies and growing, have been developed by different institutions and groups. Even though there are standard vocabularies and best practices, the flexibility of ontology languages allow ontology authors to use different techniques and patterns. If a user needs to query for information across several ontologies (e.g., looking for a string in a textual definition), she must know how each ontology developer modeled the definitions. Thus querying across multiple ontologies for common properties becomes cumbersome and error-prone. We try to alleviate this issue by providing simple reasoning (Section 3).
- Best practices in using a shared SPARQL endpoint: We have faced challenges in scaling on two different aspects: (1) the processing of complex queries and (2) client applications processing large outputs as result of a query. To develop more robust and fast applications and to promote a fair usage of our resources, we adopted simple best practices. Some of the best practices are SPARQL query constructions, others are design recommendations. We discuss these best practices in Section 4.
- Complex Query Articulation: The non-trivial mapping of complex OWL objects to RDF graphs can make queries verbose and difficult to articulate. A W3C recommendation [13] describes how OWL 2 maps to RDF graphs. Most libraries that transform OWL into RDF conform to this recommendation. To load OWL ontologies in our triple store we use the OWL-API that follows these recommendation [6]. We discuss the query articulation for OWL ontologies that are stored in a triplestore in Section 5.

In this paper, we do not try to solve these issues from a research point of view but rather we describe them so that other Semantic Web developers can plan ahead with a sense of what they will encounter. For each of these points, we present our pragmatic solutions that at least alleviated these issues. We discuss these points from a developer's point of view. When possible, we link the discussion to the current state of Semantic Web standards and technology in terms of solving a particular issue.

2 Background: Dataset Description

Researchers and practitioners in the Semantic Web normally deal with two types of information: (1) ontologies or TBoxes; and (2) instance data or simply *data*. BioPortal's content is almost exclusively ontologies and related artifacts. Other popular datasets of the Linked Data Cloud focus on *instance data* and ontologies and schemas play only a small role there. In the biomedical domain, ontologies play a very active and important role and many ontologies and vocabularies are extremely large, with tens of thousands of classes and complex expressions. For example, SNOMED CT, one of the key terminologies in biomedicine, has almost 400,000 classes [14]. The Gene Ontology (GO) has 34,000 classes [4]. These ontologies and terminologies are updated on a regular basis, some very frequently. For example, a new version of GO is published daily.

To host BioPortal's RDF content we use 4store as SPARQL server [5]. The best practices and opportunities that we describe in this paper not only apply to a particular RDF database and can be extrapolated to other deployments.

Ontology Content. The core of the BioPortal dataset is the content of each ontology that users have submitted to BioPortal. The BioPortal repository keeps multiple versions of each ontology. However, at the moment, `sparql.bioontology.org` exposes only the latest version of each. There are three main ontology formats in BioPortal:

- **OBO format** is a format that many developers of biomedical ontologies prefer because of its simplicity. OBO Editor, a tool that many ontology developers in biomedicine use, produces ontologies in this format [3]. The OWL API now provides a translation from OBO syntax into OWL syntax [22].
- The **Rich Release Format (RRF)** is primarily used by the US National Library of Medicine to distribute the vocabularies that constitute the Unified Medical Language System (UMLS) [8].
- **OWL** is a W3C recommendation for representing ontologies on the Semantic Web [9].

For OBO and OWL ontologies, the content in the triple store is the ontology that includes the closure of the *owl:imports* statements [18]. Prior to our recent quad store implementation, our data had not been stored as triples in our backend systems and therefore we need to follow a different workflow for each format to expose the existing content as RDF triples. To handle the RRF syntax we have developed the UMLS2RDF project [16]. UMLS2RDF is a set of scripts that connect to the UMLS MySQL release and transforms its content into RDF triples. To process OBO and OWL ontologies, we use the OWL-API [6]. The OWL-API can read the OBO syntax and all the OWL syntaxes (e.g: OWL/XML, Manchester, RDF and Manchester syntax). We also use the OWL-API to extract the import closure. We fetch imports from the web and materialize them, saving the whole materialized ontology in the data store.

Ontology Metadata. In addition to ontology content, we track metadata related to each ontology in the system. We represent the metadata using an OWL ontology that we developed for this purpose, the BioPortal Metadata Ontology, which extends the Ontology Metadata Vocabulary (OMV) [15]. The metadata is a set of instances in this OWL ontology. The two main entities in the metadata are *meta:VirtualOntology* and *omv:Ontology*. *meta:VirtualOntology* represents a container for all versions of an ontology; an *omv:Ontology* represents a particular ontology version (Figure 1).

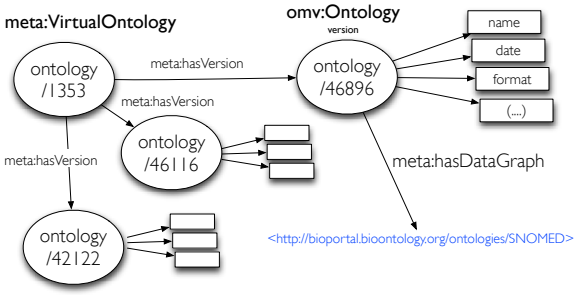


Fig. 1. Metadata: Virtual Ontologies and Version Ontologies

Mapping Data. The third type of data are the mappings between terms in different ontologies, which constitute an important part of the BioPortal repository [12]. Users can submit mappings to BioPortal through the Web interface or the REST APIs. In addition, the BioPortal team runs a series of processes to generate mappings automatically. A mapping in BioPortal connects two terms from different ontologies. It may also connect one term to many terms. We abstract the mappings into entities that record the provenance information of the mapping: the process that generated the mapping, when and how it was produced, the user who submitted it, the type of relation between classes, and so on. This information is represented in two sets of triples (a) the mapping itself and (b) the process information, which is referenced by all the mappings that the process generated.

3 Retrieval of Common Attributes from Multiple Ontologies

Ontologies in BioPortal vary in their content and structure. There are very rich representations, such as those found in the NCI Thesaurus, which has 111K *rdfs:subClassOf* relations. There are also terminologies, with no single transitive taxonomic relation, such as Medical Subject Headings (MeSH).

Ontology authors use different properties to represent common relations and attributes. The ontologies in BioPortal use 17 different properties to represent a preferred label of a term, and 28 different properties to store synonyms—even though standards, such as SKOS, provide recommendations for the properties to use in these cases. The two types of queries that we observe far more frequently than the rest are queries to (a) browse a taxonomy or (b) extract labels, synonyms and definitions. These type of queries are of particular interest for visualization and for building annotation tools. Both type of applications are very popular in the biomedical domain.

3.1 Retrieval of Preferred Names, Synonyms and Definitions

Preferred names, synonyms and definitions provide valuable term characterizations, often used in biomedical applications like annotation of clinical and scientific documents. For example, the tools developed by the BioPortal group include the NCBO Annotator, which allowed users to annotate their documents with ontology terms [21] and the NCBO Resource Index, which allows users to query an already annotated large collection of biomedical resources [7].

These types of resources must access lexical information in the ontologies such as preferred names and synonyms of terms. However, because different ontologies use different predicates to record each of this elements, it is difficult to developers to make their tools flexible enough to use any ontology in the repository. In order to provide the users of the BioPortal dataset with a uniform access to these properties, we link these different properties to the standard SKOS properties using *rdfs:subPropertyOf* relation. When ontology authors upload ontologies into BioPortal they have to choose what are the predicates that represent these attributes.

For example, properties that individual ontologies use for preferred labels all become subproperties of *skos:prefLabel* in a "globals" graph; properties that individual ontology authors chose to represent synonyms all become subproperties of *skos:altLabel*. As the result, we have a set of common predicates to query on lexical annotations across ontologies. The globals graph contains a hierarchy of properties that maps each custom attribute property to one of the standard predicates. We use this hierarchy of predicates to rewrite internally the SPARQL query using backward-chaining reasoning. Figure 2 shows an example of a SPARQL query for an ontology that uses a custom predicate to record preferred labels. In this case, the user does not need to know the specific predicate and she can query on the standard *skos:prefLabel*.

Internally our triple store rewrites the SPARQL query and not only binds the property *skos:prefLabel* but also the *rdfs:subPropertyOf* closure. Thus, the query in Figure 2 will also contain `http://NIF-RTH.owl#core_prefLabel`. BioPortal maintains a mirror of the 4store database where this backward-chained reasoning is implemented.[1] The entailment regime that we implemented follows the Minimal RDFS Semantics [10,19].

[1] `https://github.com/ncbo/4store`

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
SELECT DISTINCT ?termURI ?prefLabel
   FROM <http://bioportal.bioontology.org/ontologies/NIF-RTH>
   FROM <http://bioportal.bioontology.org/ontologies/globals>
WHERE {
   ?termURI a owl:Class;  skos:prefLabel ?prefLabel .
}
```

```
<http://NIF-RTH.owl/#nif_resourcep17:birnlex_2353> "Commercial license"
<http://NIF-RTH.owl/#nif_resourcenif_resource:nlx_res_2009014> "Biomaterial supply resource"
<http://NIF-RTH.owl/#nif_resourcep17:birnlex_2218> "3D Time-series analysis software"
<http://NIF-RTH.owl/#nif_resourcenif_resource:nlx_res_090904>  "Reference atlas"
                                        ( 150 solutions omitted)
```

Fig. 2. SPARQL Query on a standard preferred label property. The query result returns preferred labels for an ontology even though the authors used a nonstandard property for this attribute. The predicate used in this case is `http://NIF-RTH.owl#core_prefLabel`

The use *rdfs:subPropertyOf* reasoning was successful for our use case. Most applications consuming labels and definitions care only about the default predicates we provide as root elements of each property hierarchy. We documented this technique in our Wiki and we have noticed that users tend to rework the examples keeping the "globals" graph to use the standard predicates.[2]

3.2 Hierarchy Retrieval

Historically, browsing a taxonomic hierarchy is one of the most common ways to browse ontologies. The way in which ontologies represent their taxonomic hierarchy does not differ greatly among ontologies. By far, the predominant predicate for this purpose is *rdfs:subClassOf*. There are 6M triples with this predicate in the BioPortal triple store.

The fact that in modern triples stores we can load 300+ ontologies is of particular interest to applications that need to navigate multiple ontologies. Having this kind of remote service allows for incremental browsing. Applications need to know only a class IRI to start the navigation, such as root nodes. Then, with simple SPARQL queries the application would be able to browse the hierarchy. A SPARQL query like the one in Figure 3 would retrieve, by default, only direct asserted subclasses with their labels.

In `sparql.bioontology.org`, the default behaviour for queries like the one in Figure 3 is to retrieve only direct asserted subclasses; or direct superclasses if we invert the *rdfs:subClassOf* pattern. However, many times our users want to retrieve the subclass closure of a given node. If a triple store does not implement any reasoning, getting the closure requires us to query recursively until we reach all nodes. We argue in Section 5 that property paths are also not an option

[2] `http://www.bioontology.org/wiki/index.php/SPARQL_BioPortal`

```
SELECT ?subClass ?prefLabel
FROM <http://bioportal.bioontology.org/ontologies/SNOMED>
FROM <http://bioportal.bioontology.org/ontologies/globals>
WHERE {
        ?subClass  rdfs:subClassOf  <http://purl.bioontology.org/ontology/SNOMEDCT/12738006>;
                   skos:prefLabel ?prefLabel .}
```

Fig. 3. SPARQL query to retrieve all subclass elements from a given class and their labels. The IRI in the example is from the SNOMED CT ontology and represents the term "Brain Structure." This example combines taxonomy browsing with preferred name retrieval. This query outputs five results in the current SNOMED CT ontology.

to return hierarchy closures efficiently. To help users with querying hierarchy closures we again use Minimal RDFS reasoning and 4store's backward chained reasoner [19]. Backward-chain reasoning works very well in this case because it allows us to provide switchable structural *rdfs:subClassOf* reasoning. Sometimes, users only care about direct asserted subclasses and superclasses, like when visualizing the hierarchy tree. In this case, they can use an CGI parameter indicating that the reasoning should be switched off. In other cases, when users need the full closure, they can just switch it on.[3]

Some applications need to traverse the hierarchy for a fixed number of steps. For instance, the NCBO Annotator [21] includes an option to annotate text including ancestors of the terms that appear directly in the text up to a certain level. Neither direct superclasses nor structural subclass reasoning can help with this issue. But, with the SPARQL union operator it is possible to formulate queries for this purpose. The query in Figure 4 uses the UNIONS together with SPARQL 1.1 BIND operator to provide this functionality. The BIND operator allow us to identify the provenance of each resulting solution. We show two solutions at the bottom, "Parasite identification" is the direct superclass and "Identification procedure for living organism" is the superclass in distance 2. We can apply ORDER BY to the *nstep* variable to rank the results. In this case, classes that are closer to the query term are ranked higher.

The performance of queries like the one in Figure 4 is critical to the NCBO Annotator. Recently, we have measure how these queries perform as we add more UNION/BIND blocks into the query structure. Figure 5 shows the performance of queries for SNOMED CT. SNOMED CT contains a rich taxonomy with 539K subclass axioms, 395K classes and up to 32 levels in the hierarchy. We studied groups of queries that request terms within 1, 5, 10, 15, and 20 hierarchical levels from a given term. The average performance of these 5 groups that we studied remains below 0.11 seconds. Average performance for getting 5 and 10 levels—common choices by BioPortal users—are 0.019 and 0.13 seconds respectively. For NCBO Annotator this performance is within an expected range of acceptable performance. We did not see a drastic performance degradation as we added more UNION blocks into the queries.

[3] https://github.com/msalvadores/4sr/wiki/4sr-reasoning-and-queries

```
SELECT DISTINCT ?ss0 ?nstep ?label {
   { GRAPH <http://bioportal.bioontology.org/ontologies/SNOMEDCT> {
      <http://purl.bioontology.org/ontology/SNOMEDCT/122040007> rdfs:subClassOf ?ss1 .
      ?ss1 rdfs:subClassOf ?ss0 .
      ?ss0 skos:prefLabel ?label
      BIND ('2' AS ?nstep) .
    } }
    UNION { GRAPH <http://bioportal.bioontology.org/ontologies/SNOMEDCT> {
      <http://purl.bioontology.org/ontology/SNOMEDCT/122040007> rdfs:subClassOf ?ss0 .
      ?ss0 skos:prefLabel ?label
      BIND ('1' AS ?nstep) .
    } }
 } ORDER BY ?nstep

<http://purl.bioontology.org/ontology/SNOMEDCT/122069003> "1" "Parasite identification"@EN
<http://purl.bioontology.org/ontology/SNOMEDCT/108266002> "2" "Identification procedure for living organism"@EN
```

Fig. 4. SPARQL query to retrieve two ancestors of a given element. The first GRAPH block retrieves ancestors in distance 2 and the second GRAPH block ancestors in distance 1.

BioPortal allows users to browse hierarchies that are interconnected by mappings. For instance, if we try to find subclasses of term A to extract their labels and we do not find any, we can retrieve mappings for the term A and look for subclasses of a related term in a different ontology. Mappings provide connection paths that can be crossed depending on the application needs. For instance, when looking at the term "malignant hyperthermia" from the Human Disease Ontology we do not see any subclasses. But this term in BioPortal has 24 mappings. Figure 6 shows the query that would find subclasses from other ontologies where a term is mapped to "malignant hyperthermia." The query in Figure 6 retrieves 13 solutions with terms and labels from 3 other ontologies (SNOMED CT, MESH and CTV3). We discussed the details of the RDF structure of BioPortal mappings elsewhere [17].

4 Best Practices in Using a Shared SPARQL Endpoint

Our RDF data store contains 203M triples with more than 2,000 different predicates. SPARQL allows the construction of very complex queries. One typical problem are complex joins that generate very large intermediate results. The size of our database and some of the queries that our users run make this issue a recurring one. SPARQL engines are still to improve in terms of query planning optimisations and we have identified a few best practices that can be beneficial to develop applications that use shared SPARQL endpoints.

Queries containing non-selective graph patterns tend to generate large intermediate results in SPARQL engines. This issue is particularly problematic in an open SPARQL endpoint. An open SPARQL endpoint is normally a shared resource. From the perspective of the publisher, we can optimise resources if we delegate some of the query processing to the client's application. Thus, we encourage our users to query an open SPARQL endpoint iteratively with selective queries, which overall provide the functionality of a non-selective query.

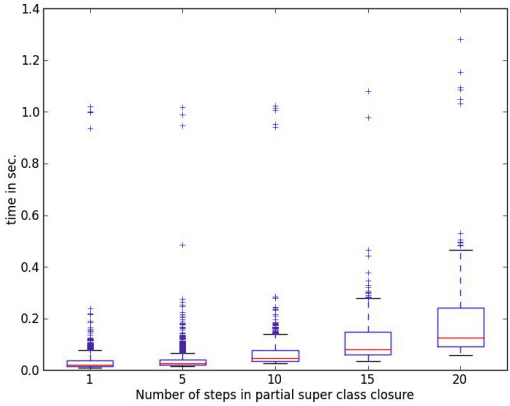


Fig. 5. Performance analysis of queries that retrieve N step superclass elements together with their labels. X-axis represents the number of steps. We have measured 1, 5, 10, 15 and 20 steps. We processed 2,800 SPARQL queries for each number of steps. We used 2,800 random leaf classes from SNOMED CT to construct queries like the one shown in Figure 4.

Figure 7 shows two different approaches to retrieving the same information from our SPARQL endpoint. Figure 7(a) uses a query that contains one non-selective triple pattern "?a ?p ?b" joined with two other triple patterns. The combination of these three triple patterns generates large intermediate results and computationally expensive joins. Queries in Figure 7(b) release the SPARQL server from some of the processing. Queries like the first one in Figure 7(b) are optimised in most SPARQL engines. The second query, that is part of an inner loop, becomes more selective. Here we do not argue that one is faster than the other. Our point is that the second approach, with multiple selective queries, is more likely to succeed in the open Web. Most triple stores implement mechanisms to restrict resources used by queries. These restrictions are often implemented in terms of "query timeouts" or "join space restrictions." `sparql.bioontology.org` uses two 4store mechanisms to limit execution of expensive queries. These are soft limits and join space restrictions.[4] Wherever non-selective queries are likely to fail, there is usually another approach with selective queries that is likely to succeed.

For the same reasons, we recommend the use of OFFSET and LIMIT to paginate over results. Queries that return all preferred names from an ontology (cf. Section 2) are likely to hit resource limits and fail on some of the largest ontologies in BioPortal, such as NCBITaxon with more than 500K terms and 4.6M triples. In these cases, paginating the results with OFFSET and LIMIT will help to return all the solutions. Most triple stores will produce a consistent

[4] `http://4store.org/trac/wiki/SparqlServer`

```
SELECT DISTINCT ?subClassOf ?label WHERE {
        ?s maps:source <http://purl.obolibrary.org/obo/DOID_8545>;
           maps:target ?target .
        ?subClassOf rdfs:subClassOf ?target .
        ?subClassOf skos:prefLabel ?label .
}
```

```
<http://purl.bioontology.org/ontology/SNOMEDCT/213026003> "Malignant hyperpyrexia due to anesthetic"
<http://bioonto.de/mesh.owl#C531737>   "Malignant fever"
<http://bioonto.de/mesh.owl#C535695>   "Malignant hyperthermia susceptibility type 2"
<http://bioonto.de/mesh.owl#C538343>   "Native American myopathy"
                                ( 8 results omitted)
```

Fig. 6. SPARQL query that uses mappings for the term `http://purl.obolibrary.org/obo/DOID_8545` to reach other ontology term and retrieve their labels. The query returns 13 solutions, in the figure only 4 are shown the rest are omitted.

(a) non-selective SPARQL queries

```
SELECT DISTINCT ?p WHERE {
 GRAPH <http://bioportal.bioontology.org/ontologies/NIF> {
        ?a a owl:Class .
        ?a ?p ?b .
        ?b a owl:Class .
} }
```

```
SELECT DISTINCT ?p WHERE {
GRAPH <http://bioportal.bioontology.org/ontologies/NIF> {
        ?a ?p ?b .
}}
```

for each $P

```
ASK {
    GRAPH <http://bioportal.bioontology.org/ontologies/NIF> {
        ?a <$P> ?b .
        ?a a owl:Class .
        ?b a owl:Class .
    }
}
```

(b) selective SPARQL queries

Fig. 7. (a) shows a SPARQL query to retrieve all the predicates that connect two resources that are instances of *owl:Class*. (b) shows two queries, the top query gets all the distinct predicates in the graph; the bottom query uses ASK to see if a predicate P connects one pair of instances of *owl:Class*. Both (a) and (b) are restricted to the NIF ontology graph.

output when using OFFSET and LIMIT without ORDER BY. The sequence of the solution will, in most cases, follow the output of the last join operation.

The use of ORDER BY and GROUP BY do not play very well with resource contention. To compute these two operators, the SPARQL query engine needs to aggregate or sort all query solutions. Thus, the output needs to be kept in memory for a longer period of time and disqualifies the engine from streaming out results. If the solution space is large enough, the query is likely to hit timeouts.

Frequently, very simple queries produce an extremely large resultset. These resultsets have to be moved from our endpoint to the client application. Even though sometimes it looks like the issued query is taking a long time to be processed, in reality, significant portion of this time is spent transferring the result and parsing the result on the client side. For instance, the retrieval of all preferred names from NCBITaxon—500K solutions—generates a JSON or XML output of 97MB or 121MB, respectively. On average, this query takes 7.05 seconds in the query engine. Parsing the JSON output takes 55 seconds using

Python 2.6 and the built-in JSON parser and 15 seconds using the python-cjson library.[5] With Java, Jena's ARQ SPARQL client library processes XML SPARQL results as a stream of solutions adding little memory overhead to the client. For the previous example, parsing the result adds only 1.6 seconds of extra processing time.[6] Sesame, also for Java, does the same trick and adds 2.4 seconds of processing time when parsing the SPARQL XML result. One can see that times can differ greatly depending on the mechanism we use to process the SPARQL results. It is important that we choose the library that performs best on a given platform; and often a library that will parse the SPARQL resultset on-demand as rows are read by the application.

We encourage users to implement caching between their applications and our SPARQL endpoint. They cannot expect that they will always get SPARQL answers within a small response time range. Like many other SPARQL endpoints in the Linked Data Cloud, `sparql.bioontology.org` is a shared resource and its performance does not depend only on the queries submitted by one user. The overall load of the server affects all users. In that sense, we do not allow singles IPs to have more than 6 threads running simultaneously. We require API keys to grant access to private graphs [17]. Though we do not yet use API keys to queue user queries and implement resource contention, this is something that we have considered for future work.

5 Complex Query Articulation

The normative exchange syntax for OWL 2 ontologies is RDF/XML. The OWL 2 specification contains a document which describes how to map OWL 2 constructs into triples that form an RDF graph [13]. For all ontologies that users submit to BioPortal in OWL format, we parse and translate them to their RDF graph representations using this mapping. In the case of OBO ontologies these are first translated to OWL using the mapping at [22]. In some cases, such as when OWL axioms contain only class, property or individual names, the mapping is straightforward. For example if :A and :B refer to class names, SubClassOf(:A :B) is mapped to one triple :A rdfs:subClassOf :B. Similarly, ObjectPropertyDomain(:R :A) is mapped to :R rdfs:domain :A. However, when ontology contains complex class expressions or OWL 2 nary axioms, such as disjoint classes axioms with more than 2 classes, a single OWL axiom may be mapped into multiple triples. These triples form tree shaped graphs connected with RDF blank nodes. For example, consider the axiom SubClassOf(:A ObjectSomeValuesFrom(:hasPart :B)) which states that all instances of :A have hasPart relationships to instances of :B (an extremely common form of axiom in biomedical ontologies, particularly OBO ontologies). We map this axiom into the following RDF Graph:

[5] `http://pypi.python.org/pypi/python-cjson`
[6] `http://jena.apache.org/`

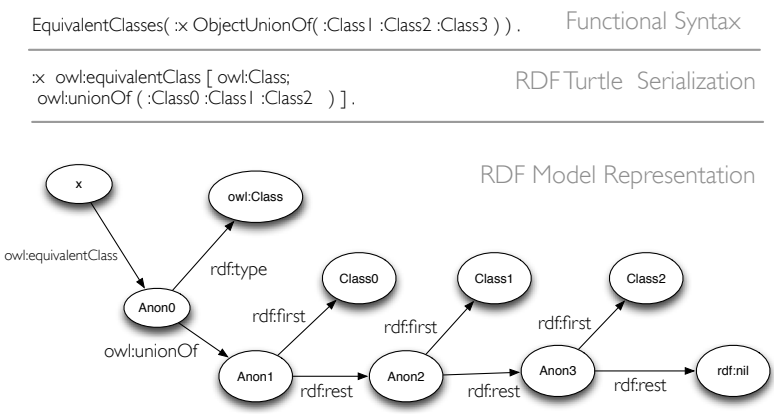


Fig. 8. Example of an equivalent class definition as a union of three classes using the OWL language. The top part of the figure shows the functional syntax often used in OWL documents. The next representation is in RDF/Turtle, a readable RDF serialization. The bottom part of the figure is the representation of the same OWL object very much as it would look like in triples.

```
:A rdfs:subClassOf _:x .
_:x rdf:type owl:Restriction .
_:x owl:onProperty :hasPart .
_:x owl:someValuesFrom :B .
```

This single axiom requires four triples. Also notice the use of blank nodes to tie everything together. For axioms that contain sets of objects, it is common to RDF lists are used to serialize these sets. For example, EquivalentClasses(:A ObjectUnionOf(:B :C :D)) gets mapped to the graph shown in Figure 8.

Ultimately, the mapping of axioms to RDF graphs is non-trivial. Querying these graphs to explore the structure of axioms requires special knowledge of this mapping, and understanding of how special RDF constructions such as RDF lists work, and multiple calls to retrieve the subgraph associated with a single axiom. From a modeler's perspective, these constructs look simple when presented in Functional or Manchester Syntax but appear overly complicated when represented as an RDF graph. In summary, the triple-based representation is not an end-user facing representation or presentation syntax.

Figure 9 shows two examples taken from BioPortal ontologies. The left-hand side is the definition of the term "Vaccine" from the Vaccine Ontology. "Vaccine" is characterized with several data type properties that give different descriptions of the term "Vaccine." "Vaccine" is a subclass of a class name (OBI_0000047[7]) but also an equivalent class of an intersection of three classes. Two of these classes are themselves complex class expressions. The RDF serialization of the

[7] OBI_0000047: Is a material entity that is created or changed during material processing.

```
obo:VO_0000001
    a owl:Class ;
    rdfs:label "vaccine" ;
    rdfs:seeAlso "MeSH: D014612" ;
    obo:IAO_0000115 "A vaccine is a processed (...) " ;
    obo:IAO_0000116 "Many vaccines are developed (...) " ;
    obo:IAO_0000117 "YH, BP, BS, MC, LC, XZ, RS" ;
    rdfs:subClassOf obo:OBI_0000047 ;
    owl:equivalentClass [
        a owl:Class ;
        owl:intersectionOf (obo:OBI_0000047
            [
                a owl:Restriction ;
                owl:onProperty obo:BFO_0000085 ;
                owl:someValuesFrom [
                    a owl:Class ;
                    owl:intersectionOf (obo:VO_0000278
                        [
                            a owl:Restriction ;
                            owl:onProperty obo:BFO_0000054 ;
                            owl:someValuesFrom obo:VO_0000494
                        ]
                    )
                ]
            ]
            [
                a owl:Restriction ;
                owl:onProperty obo:OBI_0000312 ;
                owl:someValuesFrom obo:VO_0000590
            ]
        )
    ] .
```

```
fma:AAL
    fma:FMAID "276388"^^xsd:string ;
    a owl:FunctionalProperty,
      owl:ObjectProperty ;
    rdfs:domain [
        a owl:Class ;
        owl:unionOf (
            fma:Segment_of_telencephalon
            fma:Putamen
            fma:Amygdala
            fma:Region_of_cerebral_cortex
            fma:Organ_component_of_neuraxis
            fma:Neuraxis
            fma:Dura_mater
            fma:Segment_of_neural_tree_organ
            fma:Globus_pallidus
            fma:Lobule_of_cerebral_hemisphere
            fma:Set_of_neuraxis_structures
            fma:Caudate_nucleus
        )
    ] ;
    rdfs:range fma:AAL_term .
```

Fig. 9. Examples of relatively complex OWL constructions formatted in RDF/Turtle. Left side is an example taken from the Vaccine Ontology. Right side is an example from the Foundational Model of Anatomy.

term "Vaccine" generates 10 blank nodes and 10 SPARQL queries are required to browse this graph.[8] The right-hand side of Figure 9 shows the construction of an object property where the domain is represented as the union of a collection of 12 classes.

Users using `sparql.bioontology.org` often ask how to retrieve ontology elements like the ones that we show in Figure 9. It is somehow challenging for users sitting in front of a SPARQL query editor to articulate queries that will extract the triples that construct these OWL objects. They need libraries that recursively browse the RDF graph, libraries that understand the OWL mapping to RDF graphs by means of sets of SPARQL queries that are connected properly. To the best of our knowledge, tools that offer this functionality are not yet available in the open source community. There are tools that can parse ontologies in RDF but there are no tools that can extract parts of an ontology using SPARQL.

Browsing recursively an RDF graph with SPARQL can be problematic. Some SPARQL parsers and triple store databases do not allow the use of blank nodes as IRI literals in SPARQL queries. In this case, we need to re-articulate the query that led us to the blank node to retrieve any out-going elements from the blank node. Because this issue is such a major one, most SPARQL engines have addressed it with out-of-specification implementations. RDF 1.1 is in the process of specifying an official solution. The early RDF 1.1 draft says that systems are allowed to replace blank nodes with IRIs if they follow an IRI pattern that will

[8] Assuming that we use only SPARQL 1.0 and the SPARQL 1.1 property path specification is not available.

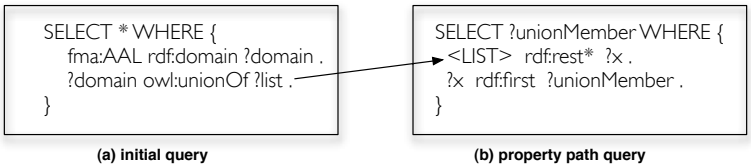


Fig. 10. Example of two queries that would retrieve the definition in Figure 9–right side. The initial query instantiates the beginning of the list. The property path query (b) uses gets as input the beginning of the from (a) and using the * property path operator traverses all the RDF list blank nodes binding the list elements in the variable *?unionMember*

be IETF registered. This technique is also known as Skolem IRIs [2]. The triple store used in `sparql.bioontology.org` has already implemented Skolem IRI replacement and our users can safely traverse RDF graphs with blank nodes [5].

The problem of retrieving RDF lists with SPARQL is, in theory, mitigated in the SPARQL 1.1 specification. In SPARQL 1.1, the property path specification defines enhanced navigational functionalities [20]. Property paths are defined as regular expressions that help to traverse RDF graphs. A property path query retrieves pairs of connecting nodes where the paths that link those nodes satisfy a path defined with a regular expression. Figure 10 shows the use of property paths to retrieve the definition in the right-hand side of Figure 9. Even though property paths provide a convenient way to query structures such as RDF lists with SPARQL, many current implementations of property paths—as of November 2011—have poor performance. Arenas and colleagues argued that the poor performance is not the result of particularly bad implementations but rather is due to the complexity of the specification itself [1]. Furthermore, according to the current specification, a property path query using rdf:rest*/rdf:first does not have to return the elements in the order they occur. The preamble of the property paths specification acknowledges this issue but contends that adding order to property paths would add significant complexity. In OWL, even though the language uses RDF lists to implement sequences of class expressions, the order of the elements does not change the "meaning." A union of classes A and B is the same as the union of B and A. Thus, many applications could potentially use property paths without order. However, many ontology visualization and editing tools like to maintain a fixed order of elements because humans tend to remember the position of UI components.

In summary, our experience shows that with or without property paths, retrieving OWL objects from a SPARQL endpoint can be challenging. The discussion that we presented in this section has described how OWL, RDF and SPARQL converge to solve parts of the problem and which parts are still matter of discussion in the Semantic Web community.

6 Conclusions

In general, RDF stores work extremely well as backend technology for quering ontology repositories due to their schema-less nature. In recent years, triple store technology has improved dramatically. Our deployment shows that it is feasible to publish 300+ ontologies—some with hundreds of thousands of classes and millions of axioms—in a public shared SPARQL endpoint. Triple stores have also become an important component in the Web of Data due to the standarization and adoption of RDF and SPARQL.

The BioPortal community had demanded access to our data via the SPARQL query language and in this paper we describe some of the design issues behind the implementation and deployment of `sparql.bioontology.org`. Our use of SPARQL is different from many other use cases because our data are primarily ontologies themselves and not data about individuals. Our experience shows that SPARQL and a small amount of reasoning can be particularly powerful in providing easy access to common attributes from our dataset, such as preferred names, synonyms, definitions and taxonomies—even though ontology authors use different RDF properties to represent these attributes. However, our experience also highlighted challenges in running a shared open SPARQL endpoint. We can overcome these challenges if we encourage developers to conform to a set of simple best practices. Finally, because our dataset includes OWL ontologies, we need to use `sparql.bioontology.org` to query the structure of these ontologies. Our experience shows that exposing OWL through a SPARQL endpoint poses a number of challenges. In future work, we plan to develop a set of SPARQL query templates to make it easier for others to explore the structure of these ontologies through `sparql.bioontology.org`.

Acknowledgments. This work was supported by the National Center for Biomedical Ontology, under grant U54 HG004028 from the National Institutes of Health.

References

1. Arenas, M., Conca, S., Pérez, J.: Counting beyond a Yottabyte, or how SPARQL 1.1 property paths will prevent adoption of the standard. In: WWW, pp. 629–638 (2012)
2. Cyganiak, R., Wood, D.: RDF 1.1 concepts and abstract syntax, `http://www.w3.org/TR/2012/WD-rdf11-concepts-20120605/`
3. Day-Richter, J., Harris, M.A., Haendel, M.: Gene Ontology OBO-Edit Working Group, Lewis, S.: OBO-Edit–an ontology editor for biologists. Bioinformatics 23(16), 2198–2200 (2007), `http://dx.doi.org/10.1093/bioinformatics/btm112`
4. Gene Ontology Consortium: The Gene Ontology (GO) project. Nucleic Acids Research 34(suppl. 1), D322–D326 (2006)
5. Harris, S., Lamb, N., Shadbolt, N.: 4store: The Design and Implementation of a Clustered RDF Store. In: Scalable Semantic Web Knowledge Base Systems, SSWS 2009, pp. 94–109 (2009)

6. Horridge, M., Bechhofer, S.: The OWL API: A Java API for OWL ontologies. Semantic Web 2(1), 11–21 (2011)
7. Jonquet, C., LePendu, P., Falconer, S.M., Coulet, A., Noy, N.F., Musen, M.A., Shah, N.H.: NCBO Resource Index: Ontology-based search and mining of biomedical resources. Journal of Web Semantics (JWS) 9(3) (2011)
8. Lindberg, C.: The Unified Medical Language System (UMLS) of the National Library of Medicine. Journal of American Medical Record Association 61(5), 40–42 (1990)
9. Motik, B., Patel-Schneider, P.F., Parsia, B.: Owl 2 web ontology language structural specification and functional-style syntax, http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/
10. Muñoz, S., Pérez, J., Gutierrez, C.: Simple and efficient minimal RDFS. Web Semant 7, 220–234 (2009)
11. Noy, N.F., Shah, N.H., Whetzel, P.L., Dai, B., Dorf, M., Griffith, N., Jonquet, C., Rubin, D.L., Storey, M.A., Chute, C.G., Musen, M.A.: BioPortal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Research 37(suppl. 2), W170–W173 (2009), http://dx.doi.org/10.1093/nar/gkp440
12. Noy, N.F., Griffith, N., Musen, M.A.: Collecting Community-Based Mappings in an Ontology Repository. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 371–386. Springer, Heidelberg (2008)
13. Patel-Schneider, P.F., Motik, B.: OWL 2 Web Ontology Language Mapping to RDF Graphs, http://www.w3.org/TR/owl2-mapping-to-rdf/
14. Price, C., Spackman, K.: SNOMED clinical terms. British Journal of Healthcare Computing & Information Management 17(3), 27–31 (2000)
15. Project, N.: The BioPortal Metadata Ontology (2012), http://purl.bioontology.org/ontology/BPMetadata
16. Salvadores, M.: UMLS2RDF Unified Medical Language System (UMLS) into RDF (2012), https://github.com/ncbo/umls2rdf
17. Salvadores, M., Alexander, P.R., Musen, M.A., Noy, N.F.: Bioportal as a dataset of linked biomedical ontologies and terminologies in RDF. Semantic Web Journal. IOS Press Journal (under review)
18. Salvadores, M., Alexander, P.R., Musen, M.A., Noy, N.F.: The quad economy of a semantic web ontology repository. In: The 7th International Workshop on Scalable Semantic Web Knowledge Base Systems, SSWS 2011 (2011)
19. Salvadores, M., Correndo, G., Harris, S., Gibbins, N., Shadbolt, N.: The Design and Implementation of Minimal RDFS Backward Reasoning in 4store. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part II. LNCS, vol. 6644, pp. 139–153. Springer, Heidelberg (2011)
20. Seaborne, A.: SPARQL 1.1 property paths, http://www.w3.org/TR/2010/WD-sparql11-property-paths-20100126/
21. Shah, N.H., Jonquet, C., Chiang, A.P., Butte, A.J., Chen, R., Musen, M.A.: Ontology-driven indexing of public datasets for translational bioinformatics. BMC Bioinformatics 10 (suppl. 2), S1 (2009)
22. Tirmizi, S.H., Aitken, S., Moreira, D.A., Mungall, C., Sequeda, J., Shah, N.H., Miranker, D.P.: OBO & OWL: Roundtrip ontology transformations. In: SWAT4LS (2009)

Trentino Government Linked Open Geo-data: A Case Study

Pavel Shvaiko[1], Feroz Farazi[2], Vincenzo Maltese[2], Alexander Ivanyukovich[3], Veronica Rizzi[2], Daniela Ferrari[4], and Giuliana Ucelli[4]

[1] TasLab, Informatica Trentina S.p.A., Italy
[2] DISI, University of Trento, Italy
[3] Trient Consulting Group S.r.l., Italy
[4] Segreteria SIAT, Autonomous Province of Trento, Italy

Abstract. Our work is settled in the context of the public administration domain, where data can come from different entities, can be produced, stored and delivered in different formats and can have different levels of quality. Hence, such a heterogeneity has to be addressed, while performing various data integration tasks. We report our experimental work on publishing some government linked open geo-metadata and geo-data of the Italian Trentino region. Specifically, we illustrate how 161 core geographic datasets were released by leveraging on the geo-catalogue application within the existing geo-portal. We discuss the lessons we learned from deploying and using the application as well as from the released datasets.

1 Introduction

Our work is settled in the context of the public administration (PA) domain. It gathers applications with a variety of constraints, interests and actors including citizens, academia and companies. Within PA, data can come from different bodies, can be produced and stored in different formats and can have different levels of quality. Thus, such a heterogeneity has to be addressed, while performing various data integration tasks.

In this paper we describe how, within the semantic geo-catalogue application [7,18], the Autonomous Province of Trento (PAT) has published some of its core geo-data accompanied with the corresponding metadata following the open government data (OGD) and the linked open data (LOD) paradigms. The goal is to experiment in practice with the realization of such paradigms in order to obtain insights on how the services offered by the PA can be improved and the above mentioned heterogeneity can be tackled more efficiently.

The need for coherent and contextual use of geographic information between different stakeholders, such as departments in public administrations, formed the basis for a number of initiatives aiming at sharing spatial information, e.g., the INfrastructure for SPatial InfoRmation in Europe (INSPIRE)[1]. See, for instance, [19,22]. Even though the publication of LOD is not required by the INSPIRE directive [1] our approach can be

[1] http://www.ec-gis.org/inspire/

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 196–211, 2012.

considered as a novel good practice to this end. In fact, in parallel with the standardization and regulation effort, the implementation of INSPIRE should take into account the linked data principles, since they facilitate data harmonization. For instance, the issue is to identify the most relevant vocabularies for RDF representation of the INSPIRE metadata elements. Also geo-data, modeled as INSPIRE themes, can be represented as RDF triples in order to facilitate its discovery and future re-use. Within the European Commission, the process has already started, for example for the INSPIRE data theme "addresses" specification which was used as a basis to model the "Address" class of the Core Location Vocabulary of the Interoperability Solutions for European Public Administration (ISA) program[2].

In turn, the OGD paradigm encourages governments to publish their data in an open (from both technical and legal perspectives) manner in order to foster transparency and economic growth (through data re-use). The theme of linking open government data gains more interest as it aims at simplifying data integration [27], e.g., by providing explicit links in advance to other relevant datasets. See for example the respective US[3] [5] and UK[4] [16] initiatives.

The contributions of the paper include:

- Description and analysis of concrete problems in the eGovernment domain;
- Details of the implementation and usage scenarios of a semantic application that manages the released 161 core geographic datasets;
- Lessons learned from deploying and using the application and the datasets.

The rest of the paper is organized as follows. Section 2 provides the problem statement. Section 3 articulates the approach adopted. Sections 4, 5, 6 present the solution realized. Section 7 outlines the related work. Section 8 discusses the lessons learned. Finally, Section 9 reports on the major findings of the paper.

2 The Application Setting

Our application domain is *eGovernment*. By eGovernment we mean here an area of application for information and communication technologies to modernize public administration by optimizing the work of various public institutions and by providing citizens and businesses with better (e.g., more efficient) and new (that did not exist before) services.

More specifically, we focus on geographic applications for eGovernment. At the european level, the INSPIRE directive aims at creating the framework for sharing spatial information by providing the respective rules leading to the establishment of such a framework. At the national level, DigitPA has produced the so-called Repertorio Nazionale Dati Territoriali (RNDT)[5] that constrains further the INSPIRE requirements for Italy. At the regional level these developments have been subsequently put in practice by requiring the existing systems to evolve in the respective directions.

[2] The ISA program: http://tinyurl.com/72538jm
[3] http://www.data.gov
[4] http://data.gov.uk/
[5] http://www.digitpa.gov.it/fruibilita-del-dato/dati-territoriali/repertorio-nazionale-dati-territoriali

2.1 The Context

One of the key components of the INSPIRE architecture is a *discovery service*, that ought to be implemented by means of the Catalogue Service for the Web (CSW)[6] - a recommendation of the Open Geospatial Consortium (OGC) - which is often realized within a geo-catalogue. See Figure 1 for an overview.

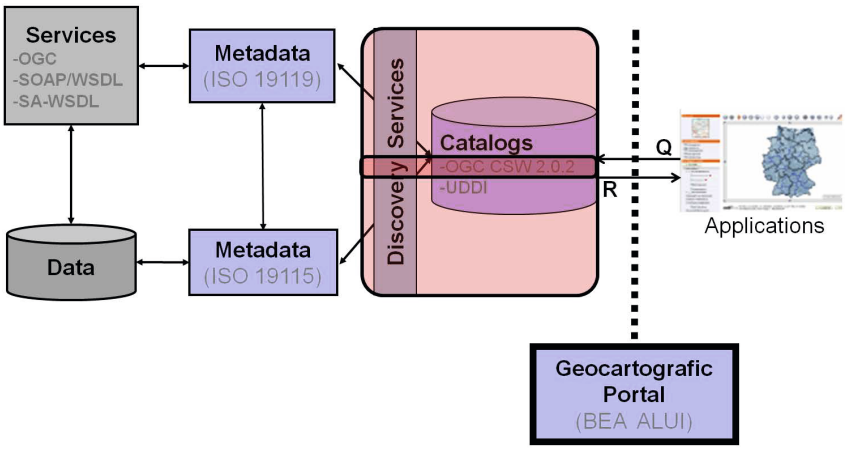


Fig. 1. Discovery services

Specifically, geo-data (e.g., in shape files) is described by metadata conforming to the ISO19115 standard. In turn, it can also be made available through services, such as OGC WMS (web map service) for map visualization or WFS (web feature service) for downloading maps (features), which are described by metadata conforming to the ISO19119 standard. Metadata is handled through a catalogue service, such as OGC CSW. The catalogue can be accessed either through applications or a web portal. We focus only on the latter.

Essentially, the geo-catalogue offers a standard mechanism to classify, describe and search information on geo-data and geo-services conforming to the above mentioned standards. There are several implementations of the CSW-based geo-catalogue, e.g., Deegree[7] and GeoNetwork[8]. We have used GeoNetwork Opensource (version 2.6). Its major functionalities include: (i) *metadata management*, namely the possibility to search, add, import, modify metadata; (ii) *user and group management*, namely the possibility to import users, their role, transfer metadata ownership; (iii) *system configuration*, namely the possibility to use various languages, harvest metadata from remote sites.

[6] http://www.opengeospatial.org/standards/cat
[7] http://www.deegree.org/
[8] http://geonetwork-opensource.org/

2.2 Towards Trentino Open Government Data

The benefits of opening some government data have been recognized at the regional level, namely in terms of: (i) the increased transparency for the public administration, (ii) the potential economic growth through data reuse, and hence, creation of new business opportunities, (iii) the potentially increased participation of citizens in PA.

Nevertheless, a critical mass has not been created yet to launch a transversal initiative in the data.gov.uk spirit. Thus, we have followed "a low hanging fruits first" approach by postponing a global strategy formulation and a roadmapping activity to a later stage, though by taking already into account the available studies in these respects [15,24].

Operationally, we have introduced the task of experimenting with open government data within an ongoing project, which is on realizing a semantic geo-catalogue [7,18]. This choice was made in order to rapidly create a practical evidence on the expected benefits with reduced costs. Thus, we have done a vertical experimentation by adapting the available geo-catalogue system, rather than by creating a new dedicated one (which we view as future work).

3 The Approach

The OGD paradigm fosters openness in both legal and technical directions. With respect to the legal openness, data should be published under a suitable license, such that third parties could freely use, reuse and redistribute it. The Open Knowledge Foundation (OKF) community provides a summary for such licenses[9]. To this end, under the recent regional deliberation n. 195/2012, the PAT formally decided to adopt Creative Common Zero (public domain) license to release 161 of its geographical core datasets. Some examples of these datasets include: bicycle tracks, administrative boundaries, ski areas and CORINE land cover.

With respect to the technical side, Trentino has been the first administration in Italy at the regional level, which experimented the publication of its data following the linked open data principles, also known as a five star rating system [2]. Specifically, we followed a standard publishing pipeline, similar to the one proposed in [12], constituted by the following sequential phases:

- *Conversion of raw data in RDF.* Data and metadata of the identified datasets were automatically converted in RDF. Data was available in shape (SHP) files and metadata in XML. Data was pre-processed with GeoTools[10] to produce XML. Both data and metadata were then processed with a standard SAX Parser[11] to extract information that were finally given in input to the Jena tool[12] to produce the corresponding RDF.
- *Linking.* To favour interpretation of the terms used and interoperability among different datasets, data and metadata are linked to external vocabularies. The high

[9] http://opendefinition.org/licenses
[10] http://www.osgeo.org/geotools
[11] http://www.saxproject.org/
[12] http://jena.apache.org/

quality of links was guaranteed by validating them manually. This has been done at the level of classes, entities and their attributes. Even if this is clearly somewhat time consuming in general, in our case this is motivated by the limited number of datasets and because of the unsatisfactory quality of the links that we obtained by using the existing linking facilities, such as Google Refine [12] and Silk [25].

- *Sharing.* The RDF data produced is made available for sharing. Our datasets are published on a web server and can be downloaded from the Trentino geo-portal. For each class (e.g., river, bicycle track) a different RDF file can be accessed.
- *Evaluation.* RDF data is evaluated by means of a developed mash-up. This has been done through the use of DERI pipes [13] that has allowed fast prototyping of mash-ups using different data sources. We have also run a workshop with the participation of the public administration, academia and industry to share and discuss the experience gained with the exercise[13].

In the next sections we describe in detail each of these phases.

4 Conversion and Linking

Within this task, both metadata and data of the 161 selected geographic datasets were automatically converted into RDF and manually linked to relevant vocabularies. To facilitate discovery and re-use, each dataset - corresponding to a different geographical feature - was converted into a different RDF file.

Metadata was initially available in the XML format. For the conversion of XML metadata into RDF, currently the available tools usually rely on a rule file providing the mapping between the source XML and the target RDF objects [26]. However, the work following this line is often limited by the non trivial requirement of learning a tool-specific rule language and the unsatisfactory quality of the generated RDF. Therefore, as an alternative to this option we have used a SAX parser to retrieve metadata from XML files. Among the widely used tools for parsing XML, we chose SAX over DOM[14] because of the high memory consumption limitation of the latter.

Geo-data was available in shape files. GeoTools, an open source java library, was used to convert them into XML, which were then parsed using SAX to retrieve data. Both metadata and data were then fed to Jena to produce RDF.

4.1 Geo-metadata Conversion

With the emergence of LOV[15] (Linked Open Vocabulary) many vocabularies are being published and similar ones are being grouped together. As a result, finding a suitable vocabulary for publishing a specific dataset in RDF has become easier. In case of unavailability of a suitable one, users can eventually propose a new vocabulary. However,

[13] http://www.taslab.eu/trentino-open-data-primi-risultati
[14] http://www.w3schools.com/dom/dom_parser.asp
[15] http://labs.mondeca.com/dataset/lov/

in order to maximize interoperability among datasets it is important to select a vocabulary among those with wider consensus. For this reason, we have encoded geographic metadata - originally provided following the ISO19115 standard - using Dublin Core (DC)[16] and DCMI-BOX[17] standard vocabularies. See an example in Figure 2.

In particular, we have focused on those metadata elements which fall in the intersection of INSPIRE/ISO Core metadata and Dublin Core. They were grouped under a resource, which was given a URI generated by appending the file identifier, for example, p_tn:piste_ciclabili metadata attribute to the http://www.territorio.provincia.tn.it/geodati/ namespace URI for the Trentino datasets.

The metadata resource language, online locator, distribution format, use limitation, title, responsible organization, version and creation date were (obviously) mapped to dc:language, dc:identifier, dc:format, dc:rights, dc:title, dc:creator, dc:version and dc:date, respectively; the geographic bounding box attributes west bound longitude, east bound longitude, south bound latitude and north bound latitude were mapped respectively to dcmibox:westlimit, dcmibox:eastlimit, dcmibox:southlimit and dcmibox:northlimit.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dcmibox="http://dublincore.org/documents/dcmi-box/"
    xmlns:dc="http://purl.org/dc/elements/1.1/" >
<rdf:Description rdf:about="http://www.territorio.provincia.tn.it/geodati/p_tn:piste_ciclabili">
    <dc:language>it</dc:language>
    <dcmibox:westlimit>10.41</dcmibox:westlimit>
    <dcmibox:eastlimit>11.97</dcmibox:eastlimit>
    <dcmibox:southlimit>45.60</dcmibox:southlimit>
    <dcmibox:northlimit>46.60</dcmibox:northlimit>
    <dc:identifier>http://www.naturambiente.provincia.tn.it/</dc:identifier>
    <dc:format>shp</dc:format>
    <dc:rights>Uso limitazione: nessuna limitazione. Altri vincoli: Dato pubblico</dc:rights>
    <dc:title>Piste ciclabili</dc:title>
    <dc:creator>Dipartimento Risorse Forestali e Montane</dc:creator>
    <dc:version>1.0</dc:version>
    <dc:date>2008-09-26</dc:date>
  </rdf:Description>
</rdf:RDF>
```

Fig. 2. Fragment of encoding geo-metadata in RDF

4.2 Geo-data Conversion

An example of how geographic data from shape files was selectively published in RDF can be found in Figure 3. To express the geographic position of the features, the UTM coordinate system was preserved. New terms were created only in case not suitable candidates were available in the standard vocabularies [10]. Specifically, we have created the length, area, perimeter and polyline terms. When available, we have specified the length of the features modeled as polylines and the area and perimeter of the features modeled as polygons.

Geometric objects that are found in data are points, polylines and polygons. A point consists of a latitude and a longitude geographical coordinate. A polyline shape is formed by a set of points, with two consecutive points that are connected by a line. A polygon shape is formed by a set of points, with two consecutive points that are connected by a line and with the first point and the last point that are the same. We have encoded all the points of the polylines and polygons in RDF.

[16] http://dublincore.org/documents/dces/
[17] http://dublincore.org/documents/dcmi-box/

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:geontology="http://www.territorio.provincia.tn.it/geodati/ontology/"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos#" >

  <rdf:Description rdf:about="http://www.territorio.provincia.tn.it/geodati/resource/piste_ciclabili">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <owl:sameAs rdf:resource="http://rdf.freebase.com/ns/guid.9202a8c04000641f8000000000428308"/>
  </rdf:Description>

  <rdf:Description rdf:about="http://www.territorio.provincia.tn.it/geodati/resource/piste_ciclabili/529">
    <geontology:length rdf:datatype="http://www.w3.org/2001/XMLSchema#double">1445.8484810675</geontology:length>
    <rdfs:label xml:lang="it">Mori - torbole</rdfs:label>
    <rdf:type rdf:resource="http://www.territorio.provincia.tn.it/geodati/resource/piste_ciclabili"/>
    <rdfs:label xml:lang="it">529</rdfs:label>
    <geo:geometry rdf:resource="http://www.territorio.provincia.tn.it/geodati/resource/piste_ciclabili_529"/>
  </rdf:Description>

  <rdf:Description rdf:about="http://www.territorio.provincia.tn.it/geodati/resource/piste_ciclabili_529">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <geontology:polyline>646339.346896746,5082179.74045936
                         646329.929020191,5082161.84683082
                         ...
                         645576.090351533,5081173.94569307
                         645575.851739799,5081173.68539361
    </geontology:polyline>
  </rdf:Description>
</rdf:RDF>
```

Fig. 3. Fragment of encoding geo-data in RDF

4.3 Linking

With this step we have linked our RDF to some of the most highly connected hub datasets from the linked open data cloud. As it can be seen from Figure 3, this has been done through the OWL owl:sameAs association. To ensure a high accuracy, the links between the resources were established manually and it took one working day.

In line with the "low hanging fruits first" approach that we have followed, we have started with DBPedia[18] and Freebase[19]. In fact, being among those with higher connection with other datasets, they guarantee a high level of reusability and interoperability. Despite they are not domain specific, they also have a broad coverage in our domain of interest.

As the next step we will link the RDF data to geographic specific datasets, such as GeoNames[20]. Also dataset ranking mechanisms, such as in [20], can be employed. As a matter of fact, we did not include GeoNames from the beginning as it lacks of features that were central to the evaluation (§6), such as bicycle tracks that at the moment is also one of our most downloaded datasets.

5 Sharing

The INSPIRE directive indicated quality of service criteria to be respected and monitored by the implementing systems: (i) *performance* - to send one metadata record within 3s.; (ii) *availability* - service available by 99% of time and no more than 15 minutes downtime per day during working hours; (iii) *capacity* - 30 simultaneous service requests within 1s. Additional requirements we have needed to comply with include:

[18] http://dbpedia.org
[19] http://www.freebase.com/
[20] http://www.geonames.org

– coherent view among other geo-related services offered by the PAT,
– centralized user authorization and authentication using standardized mechanisms,
– usage of standard architectures and interfaces for inter-system communications.

In order to satisfy these requirements, the system architecture shown in Figure 4 was implemented. It involves the following main software components:

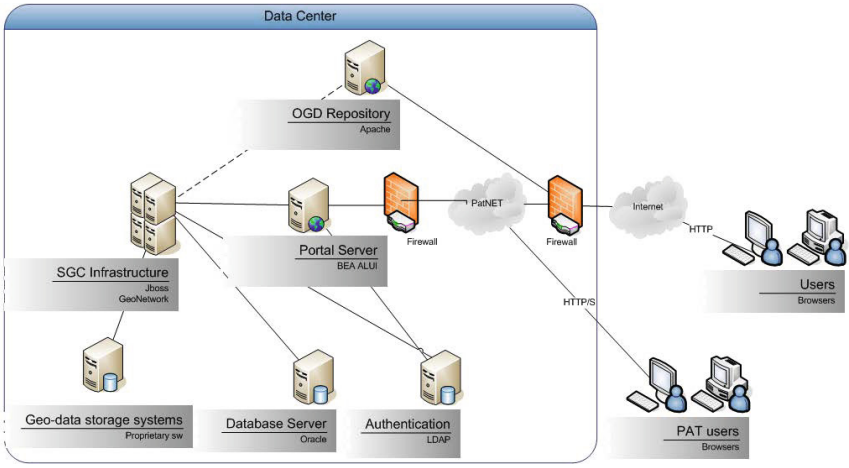


Fig. 4. System architecture

– *OGD repository* is a web-based component responsible for the access to the datasets released. It is based on the Apache web-server.
– *Portal server* is a basis of the geo-portal of the PAT and is an umbrella for all projects of the province dealing with geographical information. It groups them together and serves as a single entry point for citizens and companies. Portal server is based on the BEA ALUI proprietary software solution.
– *Geo-catalogue (SGC) infrastructure* is responsible for the access and management of geo-information (metadata and data). It is based on GeoNetwork open-source software personalized for integration with the existing proprietary software of the PAT.
– *Geo-data storage systems* are back-end systems that store geo-data in various formats (e.g., shape files). These systems are internal systems of the PAT.

With reference to Figure 5 in the following we describe how information can be accessed by using the Trentino geo-portal[21]. First of all, in order to access the geo-catalogue (Ricerca nel Geo-catalogo), the user must select SIAT (Sistema Informativo Ambiente Territorio) from the main menu.

Users can issue queries by typing them in the search box (1) and by clicking on the corresponding search button (2). Queries can be simple, such as bicycle tracks, or

[21] www.territorio.provincia.tn.it

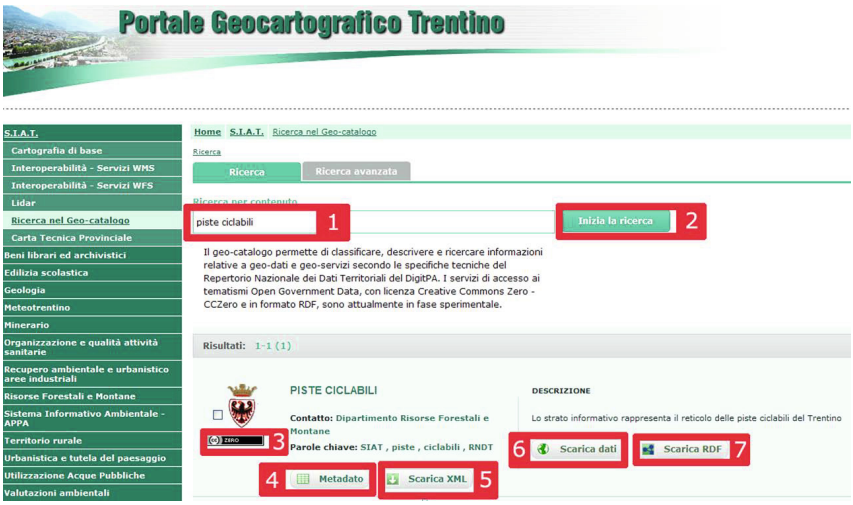


Fig. 5. Search results

more complex ones, such as Trentino mountain hovels reachable with main roads. These are semantically expanded (see [7]) and executed against the existing metadata records. Search results are shown as a list of datasets below the search box. The header on top of the list shows the total number of the datasets found and the number of datasets displayed on the current page. Each dataset is presented on the results page with its title, contact information (e.g., "department of forest resources and mountains"), keywords and description. Possible operations that can be performed on the dataset include: (4) display the geo-metadata; (5) download the geo-metadata in XML format; (6) download the raw geo-data (in a ZIP package); (7) download the dataset in RDF (§4). The icon (3) indicates that the dataset is released under the Creative Commons Zero license (CC0).

6 Evaluation

To evaluate our datasets we have built a mash-up application which is available at `http://sgc.disi.unitn.it:8080/sgcmashup/`. It enabled us to observe the usefulness of the published geo-data in linking and accessing different datasets. The purpose of this application is to support the following scenario:

> *Robert is in a summer trip to Trento cycling along the bicycle path between Trento and Riva del Garda. Once he arrived in the lakefront region of the Mori-Torbole bicycle track, he is fascinated by the splendid natural beauty of the lake and the panoramic beauty of the mountains, which made him interested to know more about the panoramic views of the other parts of the bicycle track and the nearby hotels to stay there for some days. Cycling in the summer noon made him thirsty. Hence he is eager to know the location of the drinking water fountains in the vicinity of the bicycle track.*

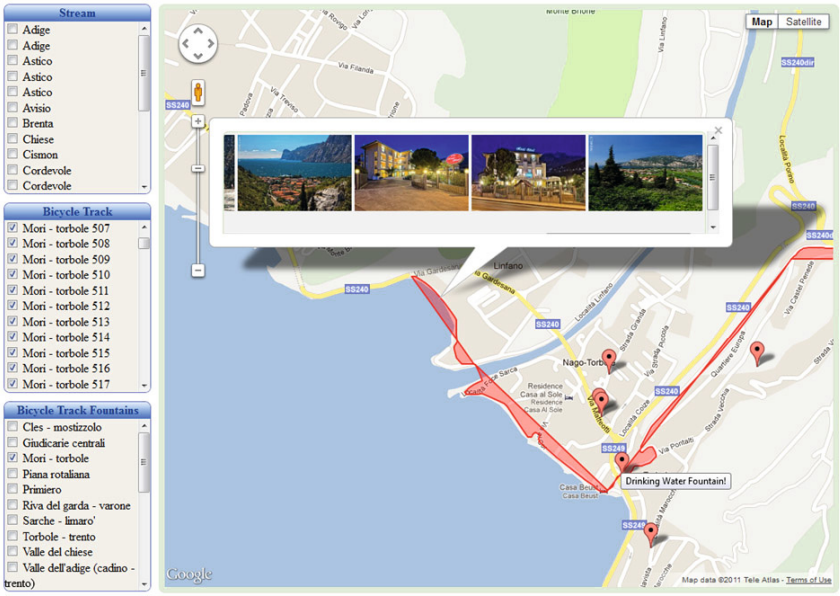


Fig. 6. The mash-up developed to support the cyclist-tourist scenario

Figure 6 provides a snapshot of the mash-up application supporting this scenario. Streams (e.g., Adige), bicycle tracks (e.g., Mori - Torbole 507) and bicycle track fountains are shown on the left as a list of check boxes, where the numbers to the right of the tracks represent the identifiers of the track parts which constitute the whole track. Selected streams, bicycle tracks and fountains are displayed using Google Maps as polygons, polylines and markers, respectively. By clicking on a bicycle track it is possible to visualize a set of images of the nearby hotels and panoramic views. We have collected images from Flickr and we have gathered information about drinking water fountains from Open Street Map through LinkedGeoData[22].

To combine information from different RDF resources, we have used the DERI pipes tool [13]. The development of this mash-up on top of the linked geo-data took a short time (about 4 working days) compared to the time required if we were to develop the same mash-up without using semantic technologies. It has required less time because, among others: (i) it has avoided the need for solving data heterogeneity issue as linked data are published in RDF or RDF compatible format (ii) it has overcome the spatial restriction on data, e.g., necessity to have all data in the same database, as it has worked simply by referring to the dataset URLs and (iii) including a new dataset to an application is less time consuming because of the open (known) data format and ease of access to data through URLs.

Finally, we have asked a local start-up company, SpazioDati.eu, to use the released datasets and in one week the company was able to design a business idea suitable to be presented at the regional workshop[13] dedicated to the release of the datasets. As a

[22] http://linkedgeodata.org/

result, at the workshop they presented the Tindes, a naturalistic index computed for the Trentino restaurants together with a mobile app and widget implementations. Overall, 32 PAT datasets were reused and mixed with 9 Open Street Map datasets. This has provided additional evidence of the usefulness of the released datasets and the possibility to build new business opportunities using them.

7 Related Work

In creating and publishing government data, the contribution of both the public administrations and universities is noticeable. In this section, we review the related work and compare it with the approach we followed along two lines: (i) open government data and (ii) publishing open data.

Open Government Data. Governments are becoming more and more active with respect to OGD. Specifically concerning geospatial data, the UK government has decided to publish them following the INSPIRE Directive using open standards, e.g., RDF for representation, SPARQL Endpoint for exposing, DCMI (Dublin Core Metadata Initiative) vocabulary for annotation and GML (Geography Markup Language[23]) for representing geographic features. Essentially, the use of a SPARQL Endpoint for exposing data allows the Semantic Web search engines - for instance Sindice[24], Swoogle[25] and Watson[26] - to discover, crawl and index the RDF data which in turn helps increasing the visibility of the data itself. Ordnance Survey[27], the national mapping agency in the UK, spearheaded the publishing of geospatial information as part of the linked data [9].

In Portugal, the Geo-Net-PT [11] dataset was created at the University of Lisbon to support applications requiring national geographic information. This dataset is published in RDF and it is linked to Yahoo!GeoPlanet[28]. Standard vocabularies were used including DCMI for metadata and WGS84 vocabulary for geographical coordinates. This dataset is also used as geospatial ontology. A SPARQL Endpoint is provided for querying it. The quality of this work is significant.

In Spain, the GeoLinked Data [3] initiative at the University Politecnica de Madrid has contributed to bringing Spanish geographic and statistical information to the linked data cloud. They have dealt with the data sources owned by the Spanish National Geographic Institute (IGN-E[29]) and Spanish National Statistical Institute (INE[30]). Their dataset is linked to GeoNames and DBPedia. For the representation of the statistical (e.g., unemployment rate), geometrical (e.g., shape) and geo-positioning (e.g., geographical coordinates) information, Statistical Core Vocabulary (SCOVO[31]), GML and

[23] http://www.opengeospatial.org/standards/gml
[24] http://www.sindice.com
[25] http://swoogle.umbc.edu
[26] http://watson.kmi.open.ac.uk
[27] http://www.ordnancesurvey.co.uk
[28] http://developer.yahoo.com/geo/geoplanet
[29] http://www.ign.es
[30] http://www.ine.es
[31] http://vocab.deri.ie/scovo

WGS84 vocabularies were used, respectively. To the best of our knowledge, similarly to Geo-Net-PT, it did not go to production.

In Italy, many communities promote OGD activities. For instance, DataGove.it aims at promoting an open and transparent government in Italy. Trentino Open Data[32] aims to sensitize public awareness of open data issues starting from the Trentino region. Moreover, in Italy many public administrations, for instance, the Piedmont region[33], are working to publish their datasets following the principles stated by OKF. However, at the time of writing, to the best of our knowledge the coverage of their published RDF datasets is quite limited (only 3 features: schools, municipalities and provinces) and no links are provided to any external datasets.

Publishing Open Data. In the following we compare the way in which we have published the open data versus alternative approaches from the state of the art.

- *Conversion:* In [12] data conversion was accomplished with the condition that the dataset had to be published in the Dcat[34] format. This is a strong limitation since in case data is not already in this format there are no tools to automatically convert other formats (e.g., CSV, XML) into Dcat. As a result, here data conversion was not automated.
- *Linking:* In our work the high quality of links was guaranteed by validating them manually. In GovWILD [4] links were established automatically with specifically developed similarity measures. In Midas [14], data about government agencies were matched by using government data extracted from documents. In [12] the alignment was done semi-automatically with Google Refine. Despite some studies show that their accuracy is good, one drawback of this and similar tools stands in the necessity to learn a specific language to handle expressions. These languages are used to specify the information which is necessary to discover the links between source and target datasets. This information includes URLs and candidate entity classes (e.g., river) and it is stored into a link specification file. Another limitation stands in the fact that they only act syntactic matching between the names of the classes. Therefore, they are unable to discover equivalent classes whose names are synonyms (e.g., stream and watercourse) or classes which are more specific (e.g., river is more specific than stream), though some ontology matching techniques can be of help here [6,17,23].
- *Sharing:* We have published our datasets by making them available on a web server. What we have done is similar to what has been done previously with GeoWordNet [8]. Alternative approaches include the usage of a SPARQL Endpoint (see, for instance, in [3,21]). In particular, in [21] along with the experiments on GeoSPARQL and geospatial semantics with the U.S. Geological Survey datasets, they show the corresponding images of the SPARQL output. In [12,5] data sharing is enabled by loading files into CKAN[35].

[32] http://www.trentinoopendata.eu
[33] http://dati.piemonte.it
[34] http://vocab.deri.ie/dcat
[35] http://ckan.org/

– *Evaluation:* We have evaluated the generated RDF linked data with DERI pipes [13] by building a mash-up application. DERI pipes have the advantage of being open source as opposed to the proprietary software alternatives like SPARQLMotion[36].

We did not have to handle enourmous quantities of data. For data intensive applications, Hadoop is often used. For instance, in [14,4], JSON, Jaql query language and Hadoop are used to provide citizens with information about U.S. government spending.

8 Lessons Learned

This section summarizes the lessons learned from deploying and using the application as well as from the release of the datasets. These lessons are articulated along the four steps (§3) of the approach that we have followed:

– *Conversion:* There is still an open question with URIs, namely which patterns to adopt. The geo-catalogue system uses by default universally unique identifiers for its records. For example, bicycle tracks correspond to 7B02F1D1-01C3-1703-E044-400163573B38, while PA would want they were self-explanatory. Thus, an approach to URI design is still to be devised and implemented. The experimentation was useful anyhow to this end, since it has increased awareness in PA that this is not a minor detail, and that URIs enable people and machines to look them up and to navigate through them to similar entities. This is especially important for the core geographic information, which is meant to last in time, and thus, should represent precise and stable reference in order to facilitate its future reuse.

– *Linking:* This is an important process, since it results in connecting the released datasets to the linked open data cloud, and hence, additional information can be discovered and integrated more easily. Experience with existing linking research tools revealed that they are still not yet flexible and precise enough, hence, manual process was preferred.

– *Sharing:* We already had a basic version of a catalogue for geo-data with some metadata conforming to the respective standards (§2.1). We have asked public administration to improve the quality of metadata, that was completed in a reasonable amount of time. This clearly facilitated the process of publishing the selected datasets. Releasing the datasets under the Creative Commons Zero license was well received by various communities with various re-launches of the news[37]. The Trentino geo-portal, being a single point of access to the geographic data, was also perceived as an appropriate place to publish the datasets. However, if this approach worked well in the context of the first experimentation, it does not scale and would create confusion, when other areas, such as statistics, culture, or tourism will start releasing their datasets.

[36] http://www.topquadrant.com/products/SPARQLMotion.html
[37] http://epsiplatform.eu/content/trentino-launches-geo-data-portal

– *Evaluation:* The internal mash-up development and a workshop[13] with PA, academia and industry (§6) has indicated that the approach adopted was a useful tactic. Local companies have perceived the value of data released by PA and would be interested in having a service for the programmatic access to the data with clear service level agreements (e.g., to have up-to-date data). This would allow them to rely on such a service and build their own applications on top of it. Also the possibility of having a feedback loop with citizens or companies in a web 2.0 fashion, signalling that some data is not precise or complete enough have to be respectively treated.

Within this experimentation we have released about 40% of the core geographic datasets of PAT. We have noticed that individuating, understanding them as well as providing metadata for them is an effort requiring collaboration of the departments owning and maintaining the respective data. We think that such datasets are of high importance, since geographic information provides a basic layer for many location-based services. The most downloaded datasets so far are administrative boundaries, bicycle tracks, and monitored rivers. With this "low hanging fruits first" approach we have managed to gain a momentum, such that an overall strategy for releasing linked open government data of Trentino should be devised briefly.

This exercise has also revealed some expectations towards the evolution of the linked open data field. For example, it has emerged the need for technology selection for the production environment to handle RDF. Comparative and convincing surveys with evaluation details are still missing that would allow for informed decision making. There is a need for instruments that support the linked data lifecycle, for example, for monitoring (and improving) the quality of data and on performing in a more automated fashion data linking and reconciliation with quality levels known in advance.

9 Conclusions

We have presented our experimental work on releasing some of the Trentino government geo-data and geo-metadata following the open government data and linked open data paradigms. Creative Commons Zero license was adopted for the release of the datasets identified. RDF has been used for representing fragments of both geo-data and the respective metadata. We have used well-known standards and specifications including Dublin Core for metadata, WGS84 for data and OWL for linking data to external resources, such as DBPedia and Freebase. New terms have been defined only when they were not available in existing vocabularies.

This was a vertical tactical experimentation to gain momentum and engagement with the stakeholders in order to show that practical results can be obtained in a reasonable time and with reduced costs (with a minimal overhead for an ongoing project). We retain that such an approach has been a success and it prepared and has opened the road for a larger transversal initiative.

Acknowledgments. The work has been supported by the Autonomous Province of Trento, Italy. We are thankful to Roberto Bona, Isabella Bressan, Giulio De Petra,

Marco Combetto, Luca Senter, Lorenzino Vaccari, Giuliano Carli, Fausto Giunchiglia, Maurizio Napolitano, Giovanni Tummarello, Michele Barbera, Piergiorgio Cipriano, Stefano Pezzi and the Trentino Open Data (TOD) group members for many fruitful discussions on the various aspects of releasing open government data covered in this paper.

References

1. European Parliament, Directive 2007/2/EC establishing an Infrastructure for Spatial Information in the European Community (INSPIRE) (2009)
2. Berners-Lee, T.: Linked Data. Design Issues for the World Wide Web - W3C (2006), http://www.w3.org/DesignIssues/LinkedData.html
3. Blázquez, L.M.V., Villazón-Terrazas, B., Saquicela, V., de León, A., Corcho, Ø., Gómez-Pérez, A.: Geolinked data and INSPIRE through an application case. In: Proceedings of GIS, pp. 446–449 (2010)
4. Böhm, C., Freitag, M., Heise, A., Lehmann, C., Mascher, A., Naumann, F., Ercegovac, V., Hernández, M.A., Haase, P., Schmidt, M.: GovWILD: integrating open government data for transparency. In: Proceedings of WWW, pp. 321–324 (2012)
5. Ding, L., Lebo, T., Erickson, J.S., DiFranzo, D., Williams, G.T., Li, X., Michaelis, J., Graves, A., Zheng, J., Shangguan, Z., Flores, J., McGuinness, D.L., Hendler, J.A.: TWC LOGD: A portal for linked open government data ecosystems. Journal of Web Semantics 9(3), 325–333 (2011)
6. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2007)
7. Farazi, F., Maltese, V., Giunchiglia, F., Ivanyukovich, A.: A Faceted Ontology for a Semantic Geo-Catalogue. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part II. LNCS, vol. 6644, pp. 169–182. Springer, Heidelberg (2011)
8. Giunchiglia, F., Maltese, V., Farazi, F., Dutta, B.: GeoWordNet: A Resource for Geo-spatial Applications. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part I. LNCS, vol. 6088, pp. 121–136. Springer, Heidelberg (2010)
9. Goodwin, J., Dolbear, C., Hart, G.: Geographical linked data: the administrative geography of Great Britain on the semantic web. Transaction in GIS 12(1), 19–30 (2009)
10. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Morgan & Claypool (2011)
11. Lopez-Pellicer, F.J., Silva, M.J., Chaves, M., Javier Zarazaga-Soria, F., Muro-Medrano, P.R.: Geo Linked Data. In: Bringas, P.G., Hameurlain, A., Quirchmayr, G. (eds.) DEXA 2010, Part I. LNCS, vol. 6261, pp. 495–502. Springer, Heidelberg (2010)
12. Maali, F., Cyganiak, R., Peristeras, V.: A Publishing Pipeline for Linked Government Data. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 778–792. Springer, Heidelberg (2012)
13. Le Phuoc, D., Polleres, A., Hauswirth, M., Tummarello, G., Morbidoni, C.: Rapid prototyping of semantic mash-ups through semantic web pipes. In: Proceedings of WWW, pp. 581–590 (2009)
14. Sala, A., Lin, C., Ho, H.: Midas for government: Integration of government spending data on hadoop. In: Proceedings of ICDE Workshops, pp. 163–166 (2010)
15. Schellong, A., Stepanets, E.: Unchartered Waters: The State of Open Data in Europe. CSC, Public Sector Study Series (2011)

16. Shadbolt, N., O'Hara, K., Salvadores, M., Alani, H.: eGovernment. In: Domingue, J., Fensel, D., Hendler, J. (eds.) Handbook of Semantic Web Technologies, pp. 840–900. Springer (2011)
17. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Transactions on Knowledge and Data Engineering (to appear, 2012)
18. Shvaiko, P., Ivanyukovich, A., Vaccari, L., Maltese, V., Farazi, F.: A semantic geo-catalogue implementation for a regional SDI. In: Proceedings of INSPIRE (2010)
19. Smits, P., Friis-Christensen, A.: Resource discovery in a European Spatial Data Infrastructure. IEEE Transactions on Knowledge and Data Engineering 19(1), 85–95 (2007)
20. Toupikov, N., Umbrich, J., Delbru, R., Hausenblas, M., Tummarello, G.: DING! Dataset Ranking using Formal Descriptions. In: Proceedings of the Linked Data on the Web (LDOW) Workshop at WWW (2009)
21. Lynn Usery, E., Varanka, D.: Design and Development of Linked Data for The National Map. The Semantic Web Journal (2011)
22. Vaccari, L., Shvaiko, P., Marchese, M.: A geo-service semantic integration in spatial data infrastructures. International Journal of Spatial Data Infrastructures Research 4, 24–51 (2009)
23. Vaccari, L., Shvaiko, P., Pane, J., Besana, P., Marchese, M.: An evaluation of ontology matching in geo-service applications. GeoInformatica 16(1), 31–66 (2012)
24. Vickery, G.: Review of recent studies on PSI re-use and related market developments. In: Information Economics, Paris (2011)
25. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and Maintaining Links on the Web of Data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 650–665. Springer, Heidelberg (2009)
26. Wielemaker, J., de Boer, V., Isaac, A., van Ossenbruggen, J., Hildebrand, M., Schreiber, G., Hennicke, S.: Semantic workflow tool available. EuropeanaConnect Deliverable 1.3.1 (2011)
27. Wood, D. (ed.): Linking Government Data. Springer (2011)

Semantic Reasoning in Context-Aware Assistive Environments to Support Ageing with Dementia

Thibaut Tiberghien[1,2], Mounir Mokhtari[1,2],
Hamdi Aloulou[1,2], and Jit Biswas[3]

[1] CNRS, Image & Pervasive Access Lab (IPAL UMI), Singapore
{thibaut.tiberghien,mounir.mokhtari,hamdi.aloulou}@ipal.cnrs.fr
[2] Institut Mines Télécom, France
[3] Institute for Infocomm Research, Singapore
biswas@i2r.a-star.edu.sg

Abstract. Robust solutions for ambient assisted living are numerous, yet predominantly specific in their scope of usability. In this paper, we describe the potential contribution of semantic web technologies to building more versatile solutions — a step towards adaptable context-aware engines and simplified deployments. Our conception and deployment work in hindsight, we highlight some implementation challenges and requirements for semantic web tools that would help to ease the development of context-aware services and thus generalize real-life deployment of semantically driven assistive technologies. We also compare available tools with regard to these requirements and validate our choices by providing some results from a real-life deployment.

Keywords: Ambient Assisted Living, Context Awareness, Knowledge Modelling, Semantic Web, Inference Engine.

1 Introduction

Ambient Assisted Living (AAL) consists of a set of ubiquitous technologies embedded in a living space to provide pervasive access to context-aware assistive services. It can for example enhance ageing in place by helping elderly people with their Activities of Daily Living (ADL). The available solutions in this field are numerous and, in most cases, robust. However, their scope of usability, mostly focused on security aspects, is generally very narrow [7,15]. To help the generalization of such systems, it would be useful to integrate them in an interoperable way. This would decrease their cost by sharing hardware or even software resources. Leveraging the system in place, we could then provide other context-aware services like reminders or ADL assistance at a lower cost and start to tackle the Quality of Life (QoL) aspects [12]. The novelty of our approach lies in the complete redesign of the semantic reasoning engine, able to adapt to people with unpredictable behaviours and evolving needs. This engine aims at providing real-time assistive services in a context-aware manner.

In Sect. 2, we present the potential use of semantic web technologies to drive the interoperability of the system. In a nutshell, semantic descriptions can be

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 212–227, 2012.

used to separate application logic from underlying models in order to avoid writing application specific code [11]. The numerous semantic web tools available have very disparate characteristics and performances. Moreover, benchmarks for such tools have limitations and a more qualitative observation on the requirements is needed to give useful hints to developers [19]. As explained by Luther et al. [11], "choosing the appropriate combination of a reasoning engine, a communication interface and expressivity of the utilized ontology is an underestimated complex and time consuming task". We spent the last year putting in place an appropriate test-case in order to give useful hints to AAL researchers and developers. Sect. 3 describes our conditions on reasoners and ontology/rules formalization to be efficiently integrated in AAL systems. Finally, Sect. 4 provides a comparison of some reasoners with regard to the suggested requirements and some results from our validation process through real-life deployment. The authors recommend to readers who are unfamiliar with AAL systems to read first the description of our prototype in Sect. 4.3 in order to get a good idea of the systems described in the coming sections.

2 Contributions of Semantic Web Technologies to AAL

The Internet of Things (IoT) describes a world where machines and physical objects are seamlessly integrated into the information network, and communicate together to exchange and process information. Tim Berners-Lee's Linked Data [2] is possibly a syntax for this exchange that encloses semantic modelling and annotation in its heart, thus improving the run-time adaptability of the communication. The powerful combination of IoT and Linked Data drives pervasive computing away from predefined bindings and static communication protocols. AAL is only an application-domain of this combination, whose specificities are being analysed here. Semantic technologies are used to perform context-aware service provision in smart environments, and have a multi-faceted role in the platform. Indeed, we referenced four main purpose to using these technologies in our use-case: 1. the modelling of assistance in smart spaces, including non-predictable behaviours, 2. the integration of all entities of the system, with an environment discovery and configuration mechanism, 3. the collaboration between modules of the system based on a shared model and lexical, 4. the reasoning to create the system's intelligence, based on the three previous points. Of course, this paper does not cover all these aspects and if a general introduction is given in this section, it will later focus only on reasoning.

2.1 Enhancing Modularity and Flexibility

To build AAL spaces or smart spaces in general, one must integrate a line-up of entities: sensor network, reasoning engine, environment actuators, interactive devices and services. By enhancing the modularity and flexibility of the system, we could go towards a larger scale deployability without decreasing the customizability of solutions. The Service Oriented Architecture (SOA) has a beneficial

contribution [10] as it provides mechanisms for the deployment and maintenance of entities as well as for the communication between them. We have augmented it with a SOA-based discovery protocol and the automatic generation of bundles (SOA software resources) in order to add a plug & play support for sensors, actuators and devices [1]. However, this only puts in place a *mechanical* plug & play where entities discover each other and start exchanging data. Entities actually do not know about each other's bindings with the environment. E.g. where has this motion detector been deployed? Who is carrying this handphone? Being able to parse data received from a new unknown entity is not enough; you need to be given its semantic. We have imagined, and described in a previous publication [1], a *semantic* plug & play where entities — services, sensors, actuators or devices — provide their semantic profile when "shaking hands" with the platform. This profile can be edited during the development, the deployment, or updated at run-time by users or even other entities. Doing so, a real plug & play behaviour is created where entities are able to genuinely *understand* each other to collaborate. Our solution is represented in Fig. 1.

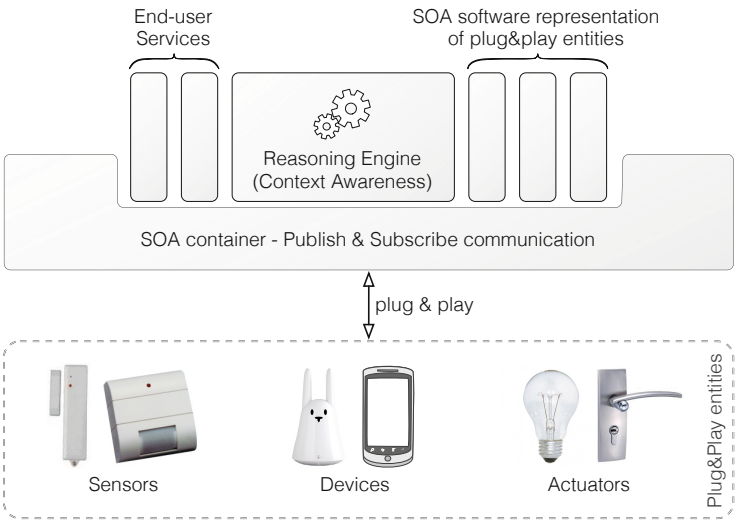


Fig. 1. High-level representation of our context-aware service platform

In the literature, pervasive systems often utilize a layer providing a level of abstraction common to all entities, helping communication, discovery and collaboration using protocols and data formats [10]. Our alternative approach is to use semantic web technologies to bring down to each entity the possibility to understand newly discovered other entities, thus decreasing the overhead on this layer, which is then solely in charge of a higher level system coordination.

2.2 An Adaptable Reasoning Engine for Context-Awareness

With the hardware plug & play and the software modularity in place, we had to reduce our use of specific application code. Indeed, adopting an imperative approach to implement context sensitive applications is very robust and requires only a short design phase. However, it brings deep constraints in term of reusability in personalized environments and adaptability in dynamic environments. As introduced in Sect. 1, a declarative approach allows for a more efficient separation between application logic and underlying models describing the use-case and peculiarities of the environment. Although this choice represents an important trade-off on the system's robustness and the effort to be put at the design phase, it seems unavoidable when targeting a deployment of more than just a dozen of homes. Although reasoners are the heart of AAL solutions, they do not need to be extremely powerful or complex. Their true requirement is to reach a consistent result in a limited time, which can be implemented using semantic reasoners. Moreover, this choice integrates well with the semantic description of the environment and its entities needed for the semantic plug & play presented above. The selection of relevant services and interaction modalities is then performed using semantic matching between the knowledge about users' context derived from sensor events and formalized into an ontology, and respectively services' and devices' semantic profiles. Finally, and as detailed further in Sect. 3.4, semantic technologies are perfectly adapted to model contextual information, along with its specificities.

3 Requirements on Semantic Inference Engines for AAL

In this part, we try to highlight the "must have" features of a semantic reasoner in order to be used efficiently into an AAL deployment.

3.1 Retractability of Knowledge

In assisted living spaces, contextual information is evolving and a detected situation is valid for a short period of time. The most needed feature for a reasoner to be used in AAL is the possibility and ease to retract information, both asserted and inferred. It has not been ignored that removing pieces of knowledge from an ontology is traditionally not a good practise. However, there are several reasons to support this choice in the targeted use-case. Most importantly, we do not want to overload the triple store with deprecated triples having an older time-stamp. We would also prefer to avoid dedicating processing time to select triples with the newest time-stamp. To support this choice, a mechanism has been designed such that the existence of a "thing" is never removed from the ontology. In other words, triples defining a new class, property or individual will never be removed. In an ontological graph, nodes are therefore anchored, while branches can be changed freely to represent the current contextual information available. E.g. if a resident walks to another room, the triple *ns:resident aal:locatedIn ns:kitchen*

is replaced by *ns:resident aal:locatedIn ns:bedroom*, whereas the "existential" triples *ns:kitchen rdf:type aal:Room* and *ns:bedroom rdf:type aal:Room* remain untouched.

We would like as well to retract inferred triples easily, when the conditions necessary to their inference are not fulfilled anymore. Using a graphical analogy, let us consider an asserted piece of knowledge as a node, and the knowledge inferred partly from this node as new nodes branching downwards (unidirectional relation) from it. The expected behaviour is that if a node is removed, which means the represented piece of knowledge is withdrawn, all nodes branching downwards from it should be removed as well. Although it is easy to use SPARQL queries, among others, to update the asserted triples in the ontology, the automatic removal of inferred triples as described above is more complex. Due to the monotonicity assumption of the Resource Description Framework (RDF), and the Semantic Web Rule Language (SWRL) being built on top of RDF, SWRL rules can be written to add new triples into an ontology but not to retract triples from it [14]. If one tries to assert a new value for a property, two values will then be coexisting. Optionally, the property can be characterised as *functional* to indicate that only one value is possible. However, this does not mean that the property will be updated but rather that the knowledge will become inconsistent when the two values are coexisting.

Some reasoners — e.g. Pellet [17], Euler [6] — have a rule syntax that is not expressive enough to allow the retraction of knowledge. One must annotate a part of the knowledge as deprecated and write external queries (e.g. with SPARQL) to filter it out. Others — e.g. Jena [4], RacerPro [8] — use rules that can remove triples. In both cases, it is needed to manually retract knowledge inferred from the asserted-then-retracted "nodes". We did implement some inference rules dedicated to cleaning the ontology after a retraction happened. Although it is working well, this increases the complexity at design level and naturally decreases the performance of the system. We finally realized when experimenting on Euler that even though its expressiveness did not allow retraction of knowledge; the reasoner having *no live state*, the knowledge previously inferred from now-deprecated data is simply not inferred anymore. The live state of a reasoner is the state in which the reasoner remains in between two inference process. It is used so as to keep in memory the inferred state of an ontology, thus inferred knowledge does not need to be inferred again. In our use-case, we prefer to use a rule engine with no live state (i.e. no memory), as it is then only needed to care about information being asserted or retracted, and the rest is handled automatically, similarly to the "downwards branching nodes" approach described above. To summarize, reasoners often implement complex mechanisms to infer knowledge with incremental updates; but we found more suitable, in the AAL use-case, to use a *naive-only inference* like what is provided by Euler.

3.2 Processing Efficiency

Taking into account more common applications of the semantic web in the cloud, one can easily imagine having reasonable resources to process knowledge.

However, in the AAL use-case, it is necessary to embed the reasoner into a low processing power and low power consumption device, so that this device can be easily integrated anywhere in a house. E.g. the reasoner used for our deployment runs on a tiny debian machine whose CPU turns at 500MHz with 500Mb of RAM, and consuming only 5W. Moreover, the data inferred is highly dynamic; unlike web data which is updated by human users with a low frequency, context information is derived from ambient sensors activations representing people's behaviour in real-time, therefore the update rate is way below the minute. Finally, the inference is used to compute which services should be provided in the environment and with which interaction modality; this is decided depending on people's action so users should have the feeling of an almost instant response time. Therefore the minimum inference frequency has to be set very low, which forms the requirement on the processing time, thus on processing efficiency.

3.3 Scalability of Inference

The conceived service platform for assistive living, partly described in this paper, is usually tested in a single room or at most an apartment. But it is difficult to estimate now the extent of the monitored/serviced space in which it will be deployed once AAL technologies get a larger impact. Let us consider that we are deploying at the scale of a whole building: should we plan to have one reasoner per room, per apartment or even per building? With regard to the Linked Data philosophy [2] and due to the interconnection of events inside the building, it makes sense to think of a reasoner per building to be able to draw relations between the data from all apartments. Or even considering the smart cities initiatives — suggesting the extension of smart spaces to the city level [3] — the number of triples to be considered at the reasoning step might suffer a genuine explosion. Thus we have to include a requirement on the scalability of the system, e.g. reasoners inferring with linear cost should be prioritized compared to those running with quadratic cost. Although we expressed in the introduction our reservations towards semantic reasoners' benchmarks, we give in Sect. 4 some figures to compare selected reasoners in this perspective.

3.4 An Opening on Uncertainty and Quality of Information

The main peculiarities of context information lie in its high interrelation, which is leveraged through the linked data approach; and its imperfection, inconsistent or incomplete information being common due to faulty hardware, delays between production and consumption of the information, or even networking problems. Although this is obvious to the engineer, ontological knowledge is naturally processed as an absolute truth if no notion of uncertainty or quality of information is introduced in the semantic model or if the reasoner is not conceived to consider these notions. A semantic modelling language can cope with this by introducing classes of information and associations in accordance with their persistence and source. The adopted description must also allow a range of temporal characterizations as well as alternative context representations at different levels of

abstraction. Introducing such concept into the reasoning engine remains very challenging, this is why we entitled this part *opening on*. Although we do not have a strong contribution here, we could not write about these requirements without mentioning the QoI aspect. The idea of the classifying associations by Henricksen et al. [9] appears to be an important key towards QoI-based semantic reasoning and although they did not explicitly refer to the semantic web paradigm, the model proposed was obviously close to it and its semantic implementation would be straight forward. We also believe that the uncertainty aspect will not be tackled by the engine itself, rather that it is the way the engine is used and wrapped that can ever address this aspect.

4 The Appropriate Reasoner

We have presented in the previous section the requirements we gathered for a practical semantic reasoner in the AAL use-case. Some are immediate necessities like the retractability of knowledge or the processing efficiency, others are key challenges enabling larger scale deployments like the scalability, or a better reliability like the quality of information. Below, we give some feedback on 4 available reasoners that we have selected and tried over the last year.

4.1 Comparing Reasoners' Usability

Jena: The Predominant Reasoner. In the AAL community, the Jena framework [4] is predominantly used. This might partly be due to the unawareness about its alternatives as well as its apparent ease of use compared to other reasoning engines. Indeed, Jena has a few advantages compared to its rivals with probably the most complete Java API for building semantic applications. Unlike most of the other alternatives, Jena has been designed to be used on Java and its way of programming is therefore more natural for a lambda programmer getting a first hand on semantic web technologies. Actually, taking into account the possibility to implement Java built-ins to be called directly from an inference rule, one might not even realize the differences brought by the declarative reasoning paradigm. Moreover, Jena comes fully-featured with, among others, an API to build, populate and modify ontologies, an inference engine using its own rule format, and an integrated SPARQL query point.

Despite all the above, we are having mixed feelings about our experience developing with Jena and would like to express some reservations about it. In fact, without having to load the ontology much, we could observe some inconsistencies in the reasoning when trying to use several rules to collaborate on one decision. When searching for an explanation to this flimsy behaviour, we found out that Jena was actually having an incomplete integrated inference engine [16] and that using Pellet [17] as an external reasoner was advised. Consequently we started to compare the features of available semantic reasoners and their ease of use in our peculiar use-case. Our motivation towards writing this paper grew as we met researchers in the community interested in finding an appropriate reasoner.

Pellet: The Famous Alternative. Since Jena has an option to use Pellet [17] as an external reasoner, it allows to change reasoner while keeping the system infrastructure, like the modules updating the ontology depending on sensors inputs. Moreover, Pellet's rules are using SWRL, the Semantic Web Rule Language, which makes the rules compatible with some other reasoners. Logically we decided to try Pellet but as explained in Sect. 3.1, SWRL does not allow retraction of triples and makes it difficult to be used in AAL use-cases.

RacerPro: The Fully-Featured Commercial Option. We then searched a reasoner able to tackle the knowledge retractability issue and found RacerPro [8], a commercial reasoner with add-ons to the W3C recommendations. Essentially, its expressiveness allows for the retraction of triples from the ontology standing in memory. Though it is necessary to write rules dedicated to retract triples to clean the ontology, the system is at least functional. Other than being a closed-source shareware, RacerPro has its own downsides due to its own powerful rule/query language. This language is actually the most complex one we have used, which did bring a heavier workload on the implementation phase.

Euler: The Lightweight, Naive Reasoner. While facing implementation difficulties with RacerPro, we found an alternative solution with Euler [6], more specifically the EYE implementation by De Roo et al. from AGFA Healthcare. Euler is notably using Notation3 (N3), the most human-readable RDF syntax. It has the advantage to be among the fastest reasoners we found that had a full OWL-DL entailment, and it is also the most lightweight of the reasoners we selected. However, we faced the same retractability limitation as with Pellet. Despite this, we found out that Euler providing a *naive* inference, it was not problematic as explained in Sect. 3.1. Our current choice remains Euler and the validation results presented in Sect. 4.3 have been obtained using EYE.

Synthetic Comparison. We have described above the process we went through and highlighted the pros and cons of each selected reasoner. The Table 1 summarizes the aspects taken into consideration with some key specifications for each of the four reasoners. Its first half provides a good representation of the engines' expressiveness, the ease of use being purely qualitative, subjective, and based on our hands-on experience. It is interesting to note that languages purely implementing the semantic web specifications are the ones we found the most straight forward to use. Based on this first half, Pellet and Euler appear to be the two best options. We refined then our analysis with more quantitative specifications, addressing in a way the concerns raised in Sects. 3.2 & 3.3. Despite the reservations we hold about such benchmarks, we realize with the response time figures the superiority of Euler. Finally, the qualitative characterisation of engines' scalability is given subjectively, taking into account the response time profile, the ease of use and the inference completeness (OWL-DL entailment). These multiple aspects and requirements taken into consideration, this is why we chose to use Euler. To be specific about the scope of this choice, the authors

Table 1. A comparative table of semantic reasoners

	Jena	**Pellet**	**RacerPro**	**EYE**
OWL-DL entailment	incomplete	full	full	full
Rule format	own, basic & built-ins	SWRL	own, powerful	N3 & built-ins
Retractability	yes	can emulate stateless	yes	stateless
Ease of use	average	easy	complex	easy
Response time for 100 triples[1]	783ms	442ms	∼503ms	4ms
Response time for 1,000 triples[1]	29,330ms	38,836ms	∼44,166ms	40ms
Response time for 10,000 triples[1]	out of memory	out of memory	out of memory	436ms
Scalability	Very limited	Average	Limited	Good
Size (download)	22.3Mb	24.3Mb	60.3Mb	12.9Mb
Licensing	freeware, open source	freeware, open source	shareware, closed source	freeware, open source

would like to highlight that Euler has two advantages applicable to any use-case: its scalability, due to its optimized implementation based on YAP-Prolog, and its human readable formalization language using N3. However, Euler is a naive (memoryless) reasoner, which is crucial from our perspective but might be counter-productive in many applications. Here lies the main trade-off in our choice.

At this point, one might also wonder about the level of reasoning chosen in our implementation. Using Euler for the inference, developers are able to use any subset of rules catering to their specific needs. Our implementation is now using a subset of OWL2-RL, but we might choose to use rules from a higher level of reasoning if needed in the future.

4.2 Design: Integration of the Reasoner into a Service Platform

In order to validate our ideas and choices, Euler has been integrated into a context-aware service platform, insuring the selection and provision of appropriate services to end-users depending on their profile and situational needs [18]. As represented in Fig. 2, the platform is based on the OSGi specification, specifically the Apache Felix implementation, which materializes the SOA approach

[1] Figures extracted from [13] for Jena, Pellet and Euler. Cross-integration of RacerPro through a comparison with Pellet [11].

and facilitates the deployment of AAL technologies as explained in Sect. 2.1. Inside the Felix container, the platform is composed of several modules (called bundles in the OSGi lexical) that can be installed, updated and removed at runtime without interrupting the platform's operation. We use this aspect of the OSGi specifications to implement the plug & play behaviour introduced in Sect. 2.1. A Wireless Sensor Management Service (WSMS) bundle has been developed to handle the ZigBee communication between sensors and the platform: once a sensor is turned on in the environment, this bundle automatically generates a new bundle representing and describing the sensor in the platform. A similar mechanism ensuring devices plug & play is currently being developed with a bundle supporting heterogeneous communication layers (e.g. Wi-Fi, Bluetooth, 3G). The dynamic aspect of OSGi bundles also permits the integration of new end-user services (e.g. reminder services, home control) at runtime.

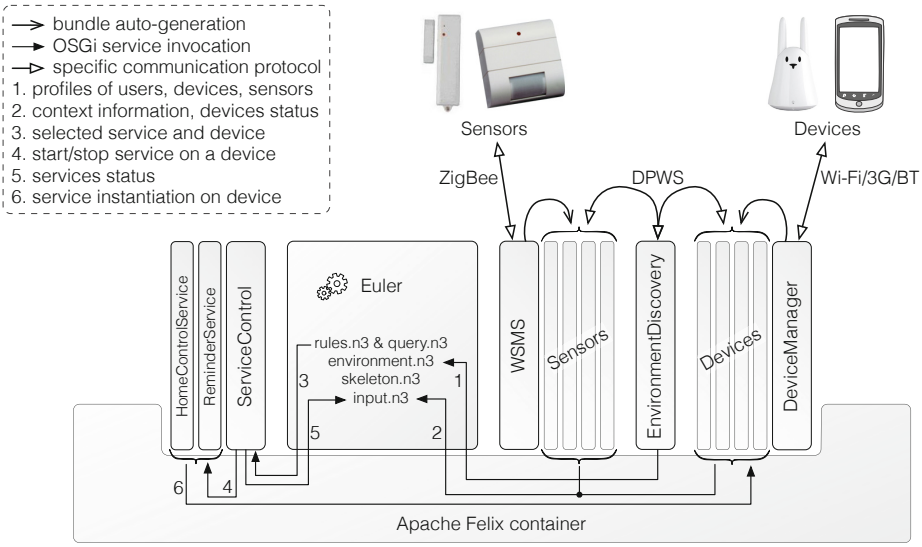


Fig. 2. Detailed architecture of our context-aware service platform

To use Euler as the core reasoner of the platform, it had to be integrated on OSGi. We took the Euler open source Java API and modified it slightly to make it compatible with the dynamic class loading feature overridden by OSGi. In the platform, Euler infers continuously in an independent system thread. The ontology is build from a set of files written in N3 and containing different kinds of information; its update is reduced to files parsing and modification. There is a file (skeleton.n3) constituted of the classes and properties that can be instantiated in the whole system to represent the current contextual information. It is the T-box of our ontological model, the highest level of modelling used in our system, containing notably models of the physical environment, the users and their

behaviour, as well as the available categories of services. Another few files (let us consider a merged example named environment.n3) contain the knowledge coming from the environment discovery phase: e.g. actual users and their profile, or sensors, devices and services along with their semantic profile. Two files (rules.n3 and query.n3) contain the rules and queries necessary for the inference process, thus centralize the application logic, which is the system context-aware decision-making. Finally, a file named input.n3 is updated at run-time through a dedicated interface to reflect the changes in the environment: real-time context information, services or devices status, etc.

In order to ensure discovery and events exchange between the different bundles in the platform, we are using the Device Profile Web Service (DPWS) protocol. DPWS uses several standards from the web services specification — namely WSDL, WS-Discovery, WS-Eventing and SOAP — in order to advertise and discover bundles, as well as for events exchange. Once a bundle representing an entity in the environment is generated, it uses DPWS protocol to advertise itself and send a description of his capabilities. A DPWS client bundle (EnvironmentDiscovery) is in charge of discovering these bundles and updating the environment.n3 file with a semantic description of their corresponding entity. Interested bundles can then subscribe to events coming from the entity, for example to update the input.n3 file. Euler parses all given N3 files, infer a high level representation of users context, and then infer which services need to be started or stopped, as well as on which devices they should be instantiated. This decision is finally executed transparently through the ServiceControl bundle.

4.3 Validation: Prototype and Deployment

After a first implementation of the platform, a validating deployment process was organized in collaboration with Peacehaven nursing home in Singapore. Peacehaven is hosting elderly residents with mild dementia who need of a continuous assistance from caregivers in order to perform their ADLs. The deployment of our platform in the nursing home assists the residents with reminders to increase their independence, as well as the caregivers by raising targeted notifications when an abnormal situation is detected. Initially, a proof of concept deployment was realized in May 2011 ending with a demonstration to the nursing home staff and management. We received good feedback about the features and performance, and filed shortly after this an Ethics Approval application for a real-world deployment with genuine residents. In August 2011, we received the Ethics Approval and put in place a sub-part of the system for technological real-world experimentation. This system's pre-trial period started in November 2011, the specificity being that interaction is instantiated only with caregivers so as to test the system without affecting residents with eventual false alarms.

Prototype Description. The platform, described in Fig. 3, is running on a tiny (115 x 115 x 35 mm, 505g) fanless debian machine, mounted with a 500MB RAM/500Hz CPU, a 4GB Compact Flash drive, and a power consumption of only 5W. Sensors are using the ZigBee communication protocol on a wireless

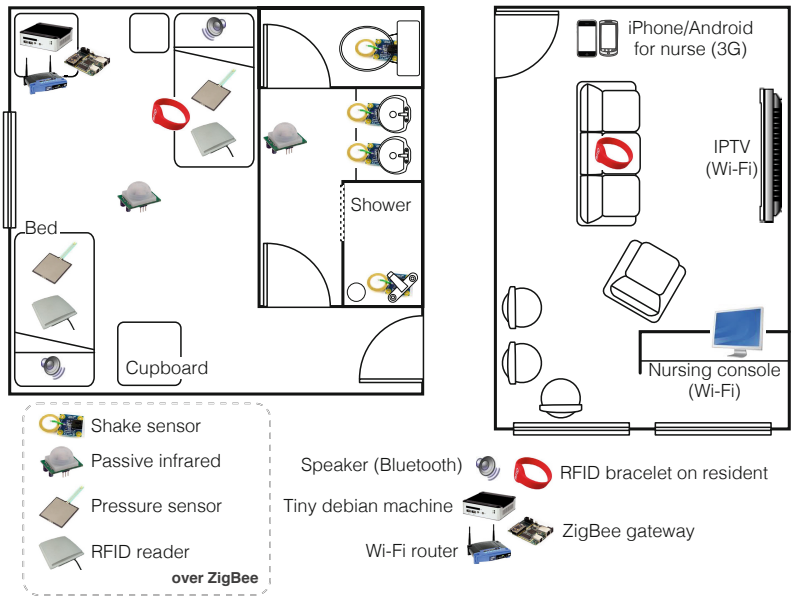


Fig. 3. Hardware infrastructure & use-case of the deployed solution

sensor network based on Crossbow's IRIS mote platform. A Crossbow node is connected via serial port to the debian machine, serving as gateway. The communication with other devices in the environment uses bluetooth for residents' embedded speakers, a client-server communication based on Jabber over Wi-Fi for the residents' IPTV and the nursing console (Windows7 machine with touchscreen) or 3G for the nurses' smartphones (Samsung Galaxy S2 with Android 2.3 and Apple iPhone 4 with iOS 5). In this phase, we monitor two residents wearing an RFID bracelet for identification, bluetooth speakers are deactivated to avoid eventual trouble to the residents in case of a system malfunction. The activities in the bathroom are monitored using shake sensors (accelerometers) placed on the pipes to detect taps/shower usage. A shake sensor is also embedded in the soap dispenser to detect soap usage during the shower. Motion sensors (passive infrared) are positioned on the ceiling of both the bedroom and the bathroom to detect presence and measure the amount of activity. The bedroom also has pressure sensors (force sensing resistors) under the mattresses and an RFID reader allowing the detection of residents and their identification. In the following phases, it is planned to increase the number of rooms monitored to reach a number of 10 residents, there are only 2 now.

In Figure 4, we illustrate our conception with a graphical representation of our model (at T-box level). It is only a part of the whole ontological model in use, simplified for readability and conciseness, thus containing only high-level concepts related to the use-case and not to the internal reasoning process. This model will then be populated with knowledge about the actual environment of

deployment, users and their profile, services activated, hardware deployed and real-time knowledge derived from the sensor data concerning the activities of the residents. In average, the T-box and A-box together constitute around 150 triples to be processed by the reasoner. Depending on the activated features, this processing consists in 10 to 15 rules and queries. The rules used are similar to the examples given below:

$$\forall\ Service\ s,\ Resident\ r,\ Location\ l,\ Device\ dc,\ Activity\ a,\ Deviance\ da$$
$$(r\ hasContext\ da) \wedge (s\ helpsWith\ da) \Rightarrow (s\ runningFor\ r)$$
$$(s\ runningFor\ r) \wedge (r\ locatedIn\ l) \wedge (dc\ deployedIn\ l) \Rightarrow (s\ onDevice\ dc)$$
$$(r\ hasContext\ a) \wedge (a\ needHands\ true) \wedge (dc\ handheld\ true) \Rightarrow (dc\ fitted\ false)$$

The inferred knowledge is not to be used by any front-end application; it is rather used for back-end decision making to provide services seamlessly. For the time-being, context information is inferred from sensor stream by another module using a business rule engine and then used to perform service and device selection semantically. However, the ontological model is currently being extended to infer contextual knowledge using our semantic engine, as defended by Chen et al. [5].

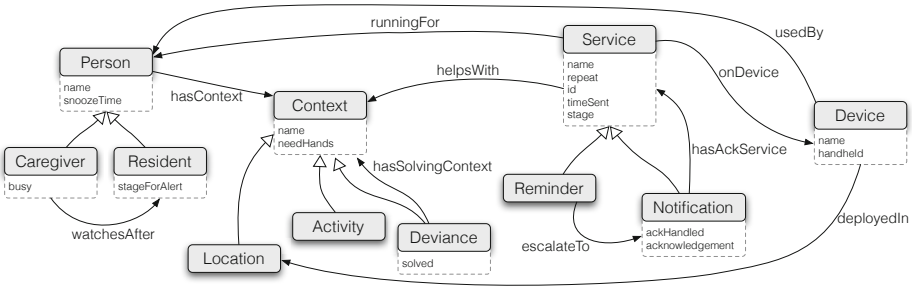


Fig. 4. Ontological sub-model for elderly assistance in a smart space

In Peacehaven nursing home, three services have been running for six months at the time of writing. These services have been designed in collaboration with the nursing staff to respond to the specific needs of the residents who agreed to test the system. These services are monitoring deviances (i.e. problematic behaviours) that are the most likely to lead to a fall. On one side, there are bathroom activities where the space is narrow and ground wet, with notifications being raised when a resident has been showering for an unusual time or when he forgets to turn off the tap of the basin. And on the other side, we raise a notification when a resident is detected to be wandering during the night.

Prototype Performance. The first aspect in which we wish to judge the system performance is regarding its uptime. In this aspect, we learnt a lot from our deployment in Peacehaven and highlighted the main areas, summarized in Figure 5, in which the platform had to be improved. We worked hard on improving these

aspects and were able to improve the average uptime, from 3 days in December 2011 to 11 days in May 2012. The more technical errors were considerably reduced and the challenge today concerns the 12% of reasoning failure, mostly due to our implementation, rather than the reasoner itself. We are currently reimplementing Euler's module, to improve on this aspect.

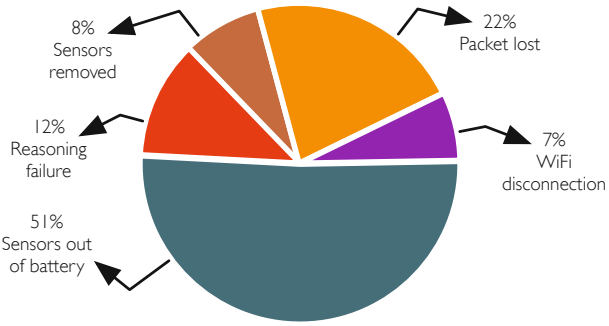


Fig. 5. Pie chart for system crash reasons in Peacehaven

During the trial period, we compute some figures to validate the platform's performance in term of accuracy and timeliness. Results are given based on the analysis of the logs for the two bathroom services described above, during an uptime period of nine days. We consider *atomic* events first — e.g. the use of taps and shower — which happened 34 times a day in average with a recognition accuracy of 71%. This was obtained by comparing the system log files with a manual record of activities provided by the nurses each time they had an intervention in the room. *Complex* events — that correspond to deviances and services provision — happened 7 times a day in average, with an accuracy of service delivery of 70%. This accuracy characterises the ratio between the number of times a service was delivered over the number of times it was needed. As complex events are derived from atomic events, we conclude that little error is introduced by the event mining (reasoning) itself. Finally, the system's reaction time, calculated between the time a service is needed and the time it is delivered in the environment, has an average of 2.713s, which has been refined in 1.226s for Euler module's processing itself, 0.735s for the communication between modules and 0.752s for the processing due to other miscellaneous bundles.

The last aspect considered to estimate the performance is the time needed to set-up the system into a new environment. Indeed, our goal was to build a more flexible system that can adapt to different environments and needs. Therefore, we have analysed the time needed to adapt the operational platform to a new use-case, counting on a team of 2 engineer-researchers. With the imperative approach used before our adoption of semantic technologies, the first reasoner was written in 5 days and its subsequent adaptation took 3 days. We then needed several months to build the first semantic version of the platform. As we were not experienced, we had to discover the existing tools, as well as build the required

models. The subsequent adaptation to a new deployment with its specificities took us only 2 to 3 hours, mostly to adapt the model. In our semantic platform, rules and thus the system logic are kept unchanged.

Of course, the aim here is not to compete with commercial systems on the real-time performance but to validate that more versatile solutions driven by semantic web technologies are an option with a sufficient performance, as observed in a real-life deployment. In this aspect, the results obtained are judged satisfactory for a first, unoptimized implementation.

5 Conclusion

In this paper, we have highlighted how AAL solutions can leverage semantic web technologies in order to enhance their modularity and versatility. AAL can indeed be driven towards a more flexible deployability by using semantic web technologies with a double role of inference engine and integrating abstraction layer. Reflecting on a year of trying out existing inference engines, we have given our take on the requirements for a semantic reasoner to be used efficiently in AAL use-cases. Some reasoners that we tested have then been compared with regard to the mentioned requirements and our choices justified. Finally, the conception and validation phases for the integration of the reasoner have been described and some results provided.

Although the tools available are not always fitting well with the AAL use-case, we observed the important contributions of semantic web technologies to this field. We are still designing and implementing some features made possible through the semantic web in order to enhance the flexibility of our solution. Among others we are planning to create a smart-space composer helping at the deployment step to configure the system, on one hand by making possible a rich semantic characterization of the deployed entities, and on the other hand by generating a specific inference rule-set from a set of abstracted meta-rules and the entire semantic characterization of the specific environment.

Acknowledgements. This work has been funded under AMUPADH project through A*STAR Home 2015 research program (SERC) in Singapore, and by CNRS ModCo project in France. The authors would like to express their gratitude to Peacehaven nursing home in Singapore and the residents involved; to Racer Systems for according us a free educational license of RacerPro; and to Jos De Roo from AGFA Healthcare Belgium for his support on Euler and his inputs to this paper.

References

1. Aloulou, H., Mokhtari, M., Tiberghien, T., Biswas, J., Kenneth, L.J.H.: A Semantic Plug & Play Based Framework for Ambient Assisted Living. In: Donnelly, M., Paggetti, C., Nugent, C., Mokhtari, M. (eds.) ICOST 2012. LNCS, vol. 7251, pp. 165–172. Springer, Heidelberg (2012)
2. Berners-Lee, T.: Design issues: Linked data (2006), http://www.w3.org/DesignIssues/LinkedData.html

3. Caragliu, A., Del Bo, C., Nijkamp, P.: Smart cities in Europe. Vrije Universiteit, Faculty of Economics and Business Administration (2009)
4. Carroll, J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., Wilkinson, K.: Jena: implementing the semantic web recommendations. In: Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters, pp. 74–83. ACM (2004)
5. Chen, L., Nugent, C., Wang, H.: A knowledge-driven approach to activity recognition in smart homes. IEEE Transactions on Knowledge and Data Engineering (2011)
6. De Roo, J.: Euler proof mechanism - eye (2005), http://eulersharp.sourceforge.net/
7. Floeck, M., Litz, L., Rodner, T.: An Ambient Approach to Emergency Detection Based on Location Tracking. In: Abdulrazak, B., Giroux, S., Bouchard, B., Pigot, H., Mokhtari, M. (eds.) ICOST 2011. LNCS, vol. 6719, pp. 296–302. Springer, Heidelberg (2011)
8. Haarslev, V., Möller, R.: Description of the racer system and its applications. Description Logics 49 (2001)
9. Henricksen, K., Indulska, J., Rakotonirainy, A.: Modeling Context Information in Pervasive Computing Systems. In: Mattern, F., Naghshineh, M. (eds.) PERVASIVE 2002. LNCS, vol. 2414, pp. 167–180. Springer, Heidelberg (2002)
10. Hong, J., Landay, J.: An infrastructure approach to context-aware computing. Human-Computer Interaction 16(2), 287–303 (2001)
11. Luther, M., Liebig, T., Böhm, S., Noppens, O.: Who the Heck Is the Father of Bob? In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 66–80. Springer, Heidelberg (2009)
12. Mokhtari, M., Aloulou, H., Tiberghien, T., Biswas, J., Racoceanu, D., Yap, P.: New trends to support independence in persons with mild dementia. International Journal of Gerontology (2012)
13. Osmun, T., De Roo, J.: Deep taxonomy benchmark - sources, http://eulersharp.sourceforge.net/2009/12dtb/
14. Plas, D.J., Verheijen, M., Zwaal, H., Hutschemaekers, M.: Manipulating context information with swrl. Tech. rep., Ericsson Telecommunicatie B.V. (2006)
15. Rougier, C., Auvinet, E., Rousseau, J., Mignotte, M., Meunier, J.: Fall Detection from Depth Map Video Sequences. In: Abdulrazak, B., Giroux, S., Bouchard, B., Pigot, H., Mokhtari, M. (eds.) ICOST 2011. LNCS, vol. 6719, pp. 121–128. Springer, Heidelberg (2011)
16. Singh, S., Karwayun, R.: A comparative study of inference engines. In: Seventh International Conference on Information Technology: New Generations (ITNG). IEEE (2010)
17. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner. In: Web Semantics: Science, Services and Agents on the World Wide Web (2007)
18. Tiberghien, T., Mokhtari, M., Aloulou, H., Biswas, J., Zhu, J., Lee, V.: Handling User Interface Plasticity in Assistive Environment: UbiSMART Framework. In: Abdulrazak, B., Giroux, S., Bouchard, B., Pigot, H., Mokhtari, M. (eds.) ICOST 2011. LNCS, vol. 6719, pp. 256–260. Springer, Heidelberg (2011)
19. Weithoner, T., Liebig, T., Luther, M., Bohm, S.: What's wrong with owl benchmarks? In: Second International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2006) (2006)

Query Driven Hypothesis Generation for Answering Queries over NLP Graphs

Chris Welty[1], Ken Barker[1], Lora Aroyo[2], and Shilpa Arora[3]

[1] IBM Watson Research Center
cawelty@gmail.com, kjbarker@us.ibm.com
[2] VU University Amsterdam
lora.aroyo@vu.edu
[3] Carnegie Mellon University
shilpaa@cs.cmu.edu

Abstract. It has become common to use RDF to store the results of Natural Language Processing (NLP) as a graph of the entities mentioned in the text with the relationships mentioned in the text as links between them. These NLP graphs can be measured with Precision and Recall against a ground truth graph representing what the documents actually say. When asking conjunctive queries on NLP graphs, the Recall of the query is expected to be roughly the product of the Recall of the relations in each conjunct. Since Recall is typically less than one, conjunctive query Recall on NLP graphs degrades geometrically with the number of conjuncts. We present an approach to address this Recall problem by hypothesizing links in the graph that would improve query Recall, and then attempting to find more evidence to support them. Using this approach, we confirm that in the context of answering queries over NLP graphs, we can use lower confidence results from NLP components if they complete a query result.

1 Introduction

Semantic web technologies like RDF and OWL are increasingly becoming desired tools for Natural Language Processing (NLP), for example to store the results of information extraction from documents. As discussed in [3], it is quite natural to store NLP results as a graph, and required elements of an NLP stack to produce RDF graphs are named-entity recognition, relation extraction, coreference resolution, and knowledge integration. Traditionally, the components of an NLP stack simply feed the knowledge integration layer as the last step of a pipeline; we have for several years been exploring the use of semantic technology to influence the NLP stack as well. In this paper, we discuss the particular problem of evaluating SPARQL queries over RDF graphs that store the results of NLP, we call these *NLP graphs*.

The simple step from special-purpose formats and storage of NLP results to using RDF immediately introduces the advantage of a well developed query language (SPARQL), and this itself quickly exposes a problem in the combination: as more complex queries are asked of the results, Recall becomes the dominating factor in performance. In particular, for queries that are a conjunction of terms, the overall Recall can be severely degraded by term Recall. In general we expect conjunctive query Recall to

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 228–242, 2012.

be roughly the product of the Recall of the relations in the conjuncts, in the worst case, query Recall can be zero even for non-zero term Recall if the variable bindings over individual terms do not overlap. For example, consider the conjunctive query to find Jordanian citizens who are members of Hezbollah:

```
SELECT ?p
WHERE {
     ?p mric:citizenOf geo:Jordan.
     mric:Hezbollah mric:hasMember ?p .
}
```

Even if the Recall of each term (`citizenOf`, `hasMember`) is non-zero, their bindings for the variable `?p` must overlap in order for the query Recall to be non-zero. Roughly speaking, we can estimate conjunctive query Recall as the product of the Recall of the relations and the probability that triples share arguments. An NLP graph can be analyzed to produce such estimates, and our empirical analysis supports this model, however our purpose is not to model it accurately, but to provide an explanation for our observations that conjuncive query Recall on NLP graphs is unacceptably low, and to provide a solution.

We describe a system that uses solutions to subsets of a conjunctive SPARQL query as candidate solutions to the full query, and then attempts to confirm the candidate solutions using various kinds of inference, external resources, and secondary extraction results. Previously we showed the hypothesis generation and validation technique is a promising way of incorporating background knowledge and reasoning in NLP graphs [1]. In this paper we extend those results by considering evidence from the secondary hypotheses produced by the components in the NLP stack that are traditionally discarded, and show they can improve Recall while also significantly improving F-measure.

2 Background

Our task is drawn from the evaluation scenarios used for DARPA's Machine Reading program (MRP)[1]. We were given a target ontology in OWL of types and binary relations, a corpus of $10,000$ documents taken from the Gigaword corpus, and a set of 79 documents manually annotated with mentions of the target relations and their argument types. The goal was to find mentions of the types and relations from the ontology in the corpus, and extract them into an RDF Graph. The MRP evaluation scenario consisted of 50 queries in SPARQL that were expected to be answered over this NLP Graph. The query results were evaluated manually by an evaluation team. Each query was supposed to have at least one correct answer in the corpus, some queries had over $1,000$ correct answers.

In order to provide more statistical significance in evaluating our system, we made a few modifications to this evaluation scenario (see section 2.3). In section 2.1 we present our basic NLP pipeline, and in section 2.2 we present our query-answering solution.

[1] http://www.darpa.mil/Our_Work/I2O/Programs/Machine_Reading.aspx

2.1 NLP Stack

The NLP Stack contains tokenizers, parsers, coreference resolution and entity disambiguation and matching modules, and entity and relation extractors trained on the ontology and text in the domain. Its main component is the RelEx relation extractor, an SVM learner with a string-subsequence kernel defined based on the context of each entity mention pair in a sentence [2]. An entity mention pair forms an instance that is *positive* if a given relation exists between them, and *negative* otherwise; there is no representational difference between explicitly negated relations and not stating the relation (e.g. "Joe is not married to Sue" and "Joe is employed by Sue" are both negative examples of the *spouseOf* relation). The output score from the SVM is converted to probabilities/confidences for each known relation that the sentence expresses it between the pair of mentions.

The NLP Stack is run over the target corpus producing an 'extraction graph', in which graph nodes are *entities* (clusters of mentions produced by coreference) and graph edges are either type statements between entities and classes in the ontology, or relations detected between mentions of these entities in the corpus.

The NLP stack produces two extraction graphs, the *primary graph* which contains the single best type, relation, and coreference results for each mention or mention pair, and the *secondary graph*, which contains all possibilities considered by the NLP stack. For example, in the case of relation detection, the system produces, for each pair of mentions in a sentence, a probability that each known relation (and 'no relation') holds, and produces only the highest probability relation in the primary graph. In many cases, the losing probabilities are still quite high, as they are computed independently, and may represent quite useful extraction results. We keep these secondary results for potential exploitation when needed, as discussed later (Section 4).

The graphs produced are special kinds of RDF graphs, which we refer to as *NLP Graphs*. A particular triple in the graph is not an expression of truth, rather it is understood to be a representation of what an NLP component, or a human annotator, read in a document. Attached to each triple is a confidence supplied by system components, and provenance information indicating where the triple was believed to have been stated in natural language. Again, the confidence is not that the triple is true, but reflects the confidence that the text states the triple.

2.2 Query Answering

The queries are a conjunction of terms from the ontology with some relation arguments as variables and some bound. For example, consider the query asking to find all terrorist groups that were agents of bombings in Lebanon on October 23, 1983:

```
SELECT ?t
WHERE {
   ?t rdf:type mric:TerroristOrganization .
   ?b rdf:type mric:Bombing .
   ?b mric:mediatingAgent ?t .
   ?b mric:eventLocation mric:Lebanon .
```

```
    ?b mric:eventDate "1983-10-23" .
}
```

The query answering system must find all bindings for the variable `?t` that satisfy the conjunctive query and also report where in the target corpus the answers are found. In practice, this means finding spans of text ('provenance') expressing the relations in the query.

2.3 Evaluation Queries and Ground Truth

The original MRP evaluation required extensive manual effort to perform because of one basic problem: when RDF graphs are used to store the results of NLP, there is no way to reliably produce identifiers on the nodes in the graph so that they will match the identifiers on the nodes in a ground truth. To address this problem, system results included provenance information in a special format that evaluators would use to find the mention (i.e. document and span) of an entity in a graph. Using the provenance, evaluators semi-automatically mapped the entity IDs from system results to entity IDs in the ground truth. This process proved expensive and error-prone, and was difficult to reproduce, consequently making it difficult to test systems adequately before the evaluation. Due to the cost, only 50 queries were used in the original MRP evaluation, which did not give us a statistically significant way to validate system performance overall, let alone to test parts of the system in isolation.

In order to generate more meaningful experiments over more queries, we had to eliminate the manual entity ID mapping step. To do this, we generated an NLP graph from manually annotated data, which we called a gold-standard graph. From the gold-standard graph, we automatically generated SPARQL queries (this process is described below), ran those queries against the graph and produced query results which we called the gold-standard results. We then measured performance of system results against these gold standard results. It should be clear that since the gold-standard and system NLP graphs both use the same set of node IDs, comparing system query results to gold standard query results is trivial, the only difference between the two graphs (system and gold standard) is the topology of the graph, not the nodes.

The manual annotation took place on a corpus of 169 documents selected from the gigaword corpus, which were completely annotated with types, relations, coreference, and entity names (when they occurred in the text). The corpus was split into two sets of 60 documents for train and devtest, and a 49 document final (blind) test. The split was done in a way that balanced the relative frequencies of the relations in each set (see Table 1). The training set was used to train the relation extraction (RelEx) component on gold standard mentions, their types, and coreference. Gold standard mentions, types, and coreference were also used when applying RelEx to the train, devtest and test sets. Obviously this increases the F-measure of the RelEx output, but our experiments used that output as a baseline. Again, the purpose of this approach was to give us entity IDs from the gold standard in our system output.

The original MRP evaluation queries had at least one answer in the corpus. Our evaluation SPARQL queries were generated from the gold standard NLP graph for each

document set by extracting random connected subsets of the graph containing $2-8$ domain relations (not including `rdf:type`), adding type statements for each node, and then replacing nodes that had no proper names in text with *select* variables. Since the original MRP evaluation queries were not randomly generated, but hand-written ostensibly based on the percieved needs of the domain (national intelligence), we analyzed them and found they tended to be grounded in known (i.e., named) entities, such as "perpetrators of bombings in *Lebanon*" or "spouses of people who attended school in *NY*". To achieve a similar effect in our synthesized evaluation queries, we ensured each query included the id of an entity that was mentioned by name (as opposed to purely nominal or pronominal) at least once in the corpus. We generated 475 test queries for the devtest set and 492 for test. These queries were then run over the same gold standard graph they were generated from – since they had variables the results would be different than what we started with – and the results became our gold standard results for query evaluation.

Although imperfect, our evaluation setup allowed us to run experiments repeatedly over hundreds of queries with no manual intervention. Our choices sacrifice graph size as well as giving up system typing and coreference in the output, which are sources of NLP errors that our hypothesis generation and validation component are capable of correcting.

3 Query-Driven Hypothesis Generation

The primary obstacle for answering conjunctive queries over NLP Graphs is Recall, which degrades geometrically with the number of query terms. To address this problem, we devised a system of hypothesis generation that focuses attention on parts of an NLP Graph that *almost* match a query, identifying statements that if proven would generate new query solutions.

3.1 Generating Hypotheses

The system we developed relaxes queries by removing query terms, then re-optomizes the query and finds solutions to the remaining set of terms, using those solutions to ground the relaxed queries for further analysis. In a sense we are looking for missing links in a graph that, if added, would result in a new query solution. The variable bindings for the relaxed query results give us a place to start. Each solution to a query Q is a single binding over the set of variables V shared among N query terms. We relax Q by removing a query term, leaving a relaxed query Q' having some subset of variables V' shared among the remaining $N-1$ terms. The solutions to Q' provide bindings over V'. We use those bindings to ground all of the original N query terms in Q. These terms of Q grounded using the bindings from solutions to Q' are hypotheses to be tested. Of course, $N-1$ of the hypotheses are already satisfied by the extractions, but since extractions are themselves noisy, we subject them to hypothesis testing, too.

Each set of N query terms is a hypothesis set: if the hypotheses in a hypothesis set can be proven, the hypothesis set constitutes a new solution to Q. If no solutions to subqueries of size $N-1$ are found, we consider subqueries of size $N-2$, and so on

down to subqueries of size $N/2$. (In practice we rarely go lower than $N-2$: the number of hypotheses begins to explode, and they are rarely related enough to be useful). Each relaxed subquery is dynamically optimized based on the number of term solutions and shared variables.

For example, consider again the query from section 2.2 asking to find all terrorist groups that were agents of bombings in Lebanon on October 23, 1983. The system would consider five relaxations at level one, corresponding to the subqueries of length four resulting from ablating each term in turn:

1. find all bombings in Lebanon on 1983-10-23 with agents (hypothesize that the agents are terrorist organizations)
2. find all events in Lebanon on 1983-10-23 by terrorist orgs (hypothesize that the events are bombings)
3. find all bombings in Lebanon on 1983-10-23 (all known terrorist orgs are hypothetical agents)
4. find all bombings by terrorist orgs on 1983-10-23 (hypothesize that the bombings were in Lebanon)
5. find all bombings by terrorist organizations in Lebanon (hypothesize that the bombings were on 1983-10-23).

If no solutions for any of the five subqueries are found, the system will attempt to generate hypotheses for the twenty subqueries of length three, and so on.

One special case is when subquery solutions do not bind a variable. This occurs for subqueries of length $N-k$ for any variables that appear k (or fewer) times in the original query. For example, if the above query produced no solutions to subqueries of length $N-1$, the system would consider subqueries of length $N-2$. At this level, we would produce a subquery in which `?t` does not appear. Solutions to the subquery would provide candidate bindings for `?b`, which would generate `mediatingAgent` hypotheses in which the first argument is bound, but the second is variable. Hypotheses with variable arguments risk generating a large number of query answers improving Recall but degrading Precision.

Finally, we note that generating hypotheses from relaxed subqueries is appropriate for queries that are *almost answerable*. Hypotheses from missing terms will likely only be useful when most of the terms in the query are not missing. Our system does not, however, blindly accept hypotheses. They are only accepted if there is evidence for them and confidence in the evidence above certain thresholds (see 4.3). Nevertheless, this approach is biased toward generating more answers to queries and will likely perform poorly when doing so is not appropriate (for example, queries for which the corpus does not contain the answer).

4 Hypothesis Validation

Each hypothesis set from the hypothesis generator contains hypotheses in the form of RDF statements, which, if added to the primary extraction NLP graph, would provide a new answer to the original query. These hypotheses are not accepted simply because they would provide new answers, rather they are targets for further, deeper processing

that for a variety of reasons was not performed as part of primary NLP stack processing. We call this deeper processing of hypothesis *hypothesis validation*. Only validated hypotheses are added to the query result sets.

4.1 Architecture

All hypotheses are passed to a stack of hypothesis checkers. Each hypothesis checker reports its confidence that the hypothesis holds and, when possible, gives a pointer to a span of text in the target corpus that supports the hypothesis (the provenance). In the current architecture, hypothesis checkers may also choose whether to allow a hypothesis to flow through to the next checker. A significant property of the architecture is that an individual hypothesis checker can be any arbitrary module capable of reporting support for a hypothesis with confidence and provenance, including external IR systems, IE systems, knowledge bases, reasoners or even independent question answering systems.

One of the prime motivations for this architecture is to circumscribe complex or problematic computational tasks, such as formal reasoning or choosing between multiple low-confidence extractions. When isolated to proving, supporting, or refuting individual RDF statements, these tasks can be made more tractable by providing a goal. For example, if we ran a tableau-based reasoner over the primary extraction NLP graph, the reasoner would report inconsistency (see for example [3]) and fail to do anything useful. However, when constrained to only the part of a graph that is connected to a hypothesis, the reasoner may be used effectively. We did not use a tableau reasoner in the experiments described here, it remains future work to put that together with the individual hypothesis checkers discussed below, but the point is the same: complex computational tasks are made more tractable by using hypotheses as goals.

4.2 Hypothesis Checkers

We have tried many hypothesis checking components. For the experiments described here, we used only three:

Primary Extraction Graph. The original primary extraction NLP graph is itself just one of the hypothesis checkers in our stack. By default, this means that the hypotheses corresponding to the $N - 1$ terms not ablated are guaranteed to be supported, with the original extraction confidence and provenance. By including these terms as hypotheses and rechecking them, our system could adjust their confidence using other hypothesis checkers and even reject them if their confidence is low. Other extraction graphs could also be plugged in as checkers (extraction graphs over other corpora, or even a new extraction graph over the same corpus with different NLP stack configurations). For these experiments, however, it was only possible for a hypothesis supported by the primary graph to increase in confidence through support from the knowledge base.

Secondary Extraction Graph. Many existing NLP components consider multiple interpretations of the text during processing and score or rank these interpretations. When used in a pipeline, it is very common to simply take the top interpretation and pass that to the next stage. The individual components are typically tuned to a performance tradeoff that maximizes F-measure. In the case of answering conjunctive queries over NLP

graphs, we needed to explore options that would emphasize Recall, as discussed above. The secondary extraction graph is an RDF NLP Graph generated from *all the interpretations considered by the NLP Stack.* In practice this can include multiple mentions, types on those mentions, multiple entities, multiple types on the entities, and multiple relations between them, and can obviously be quite large. In these experiments we explored only the interpretations considered by the RelEx component, which generates a probability of every known relation holding between every pair of mentions within a sentence. Our secondary graphs are pruned at a particular confidence threshold (or rank), and in our experiments we explored different confidence thresholds. Clearly the secondary graph can have very high Recall, but extremely low Precision: for RelEx the Recall with no threshold should approach 1 and the Precision should approach $1/|R|$ ($|R|$ is the number of known relations), since for each mention pair a probability is generated for each relation and no more than one will be correct in the ground truth. The central hypothesis of this paper is that the secondary graph is a useful source of information when it is used in conjunction with the primary graph to answer a query.

Knowledge Base. A third hypothesis checker is a knowledge base allowing taxonomic inference as well as more complex rules. For our evaluation, the knowledge base contained rules derived directly from the supplied domain ontology. It also contained a small number of general, domain-independent rules (such as family relationships). Rules derived directly from the ontology include:

- simple superclass-subclass rules ($Bombing(?x) \rightarrow Attack(?x)$)
- simple relation-subrelation rules ($hasSon(?x, ?y) \rightarrow hasChild(?x, ?y)$)
- simple relation inverse rules ($hasChild(?x, ?y) \leftrightarrow hasParent(?y, ?x)$)

We also derived about 40 more complex rules automatically from the ontology, based on specialization of the domain or range of subrelations, e.g.

$$(hasSubGroup(?x, ?y) \& HumanOrganization(?x) \rightarrow hasSubOrganization(?x, ?y)).$$

In other experiments [1] we used the knowledge base as a hypothesis checker to confirm, for example, the hypothesis $eventLocation(?b, Lebanon)$ when the extraction graph contains $eventLocation(?b, Beirut)$. In the experiments described here, however, we turned this off as it requires an entity matching step (that maps text strings to knowledge-based IDs in the background geographic knowledge base [5]), which would have had to be run on both the ground truth and the extraction graphs. This step has its own kinds of errors, which we wanted to remove from these experiments.

4.3 Accepting Support for Hypotheses

Each hypothesis checker provides its measure of confidence that a hypothesis holds, making it possible to learn thresholds for accepting or rejecting hypothesis support. This was one of the main reasons for our experimental design: the 50 manually scored queries provided by the program did not give us a satisfactory space to tune the threshold parameters. In the experiments described below, we explored this space over hundreds of queries.

With a secondary graph to support hypotheses, we need two thresholds: one for the primary graph and one for the secondary.

Accepted hypotheses are added to the extraction graph and become part of query results. For hypotheses generated from subqueries of length $N - 1$, each accepted hypothesis is an RDF triple that results in a new answer to a query. For hypotheses generated from subqueries of length $N - 2$ or shorter, hypothesis additions may combine to produce new query answers.

Extending the knowledge in the extraction graph with supported hypotheses may also produce answers to other queries that had no solutions, or produce more answers to queries that did. This does not mean an increase in Recall of course, as the hypotheses, even if supported, might be incorrect. For the current experiments we used fully bound hypothesis sets as complete solutions to queries, and not as additions to the primary extraction graph.

5 Experiments

The main purpose of the paper is to present the hypothesis generation and validation framework for NLP Graphs. The central hypothesis is that within this framework, there is value in the secondary extraction graph for conjunctive query answering. More precisely, it is our hypothesis that the probabily of a secondary graph statement being correct increases significantly when that statement generates a new result to a conjunctive query over the primary graph.

5.1 Experimental Setup

As discussed in 2.3, a manually annotated corpus of 169 documents was split into train, development, and test sets of 60, 60, and 49 documents, split in a way to balance the distribution of the domain relations (see below, however, for notes on this balance). In total there are 48 relation types. Table 1 lists the number of instances of these relations in the train, development and test sets. Since documents contain multiple relations, it was not possible to balance the relation distribution perfectly.

The RelEx component was trained using gold standard mentions and types, and was applied to the test and devtest documents also using gold mentions and types. The NLP Graphs were generated from the RelEx output as well as gold standard coreference information, the latter forming gold standard entities that were represented as NLP Graph nodes, with the selected relations representing links between them.

The NLP graphs can be generated by selecting relations from the NLP output in two dimensions:

- **Top vs. All:** Whether to take the top scoring relation between any mention pair (top) or all relations (all) between the pair.
- **Thresh:** A confidence threshold (strictly) above which relations are selected.

Clearly the same dimensions could have applied to the selection of type triples for the NLP graph, but the type information in these experiments is from the ground truth, so all type information from the NLP output was selected.

Table 1. Distribution of relations in train and test sets

Relation	Train	Dev	Test	Relation	Train	Dev	Test
affectedBy	317	342	204	hasMember-HumanAgent	37	45	46
agentOf	812	847	596	hasMember-Person	29	35	19
attendedSchool	2	3	5	hasSibling	16	9	4
awardedBy	1	3	1	hasSpouse	8	7	15
awardedTo	1	3	1	hasSub-Organization	8	6	4
basedIn	122	93	51	instrumentOf	4	2	1
before	14	21	10	isLedBy	139	236	118
building-Physically-Destroyed	21	7	9	killingHuman-Agent	26	17	13
capitalOf	7	4	5	locatedAt	396	410	264
clientOf	23	14	14	mediating-Instrument-WeaponClass	15	12	4
colleague	29	23	16	near	31	22	8
diedAt	8	2	8	overlaps	4	0	3
diedOf	67	34	38	ownerOf	82	49	62
diedOn	14	8	11	participantIn	39	71	36
employedBy	201	216	170	partOf	188	236	132
eventDate	196	175	143	partOfMany	289	230	142
eventLocation	82	42	63	personKilled	2	7	1
founderOf	12	7	2	personGroup-Killed	0	1	0
hasAgeInYears	24	24	24	populationOf	4	1	2
hasBirthDate	1	2	0	quantityOf	204	176	114
hasBirthPlace	12	6	3	rateOf	1	2	2
hasChild	28	27	31	relative	21	13	9
hasCitizenship	9	10	7	residesIn	67	37	26
hasDisease	18	28	18	spokespersonFor	17	27	4
				timeOf	0	1	1

We refer to the graphs by document set (dev or test), $top/all@threshold$. Thus $devTop@.2$ is the NLP Graph from the development set using top relations above a .2 confidence, and $testAll@0$ is the NLP graph from the test set using all relations above 0 confidence.

For development and test sets, we generated 3 NLP graphs for use as primary extraction graphs, in all cases using top, and selecting relations at thresholds 0, .1, and .2. We also generated a single NLP graph for use as secondary for each set, using the $all@0$ setting.

Precision and Recall are determined by comparing system query results to the ground truth query results. The confidence of a query result set is the minimum confidence of the statements in the result, which performed better in experiments than using the product. Therefore for experiments with validation in a secondary graph the confidence thresholds below the primary graph threshold indicate the confidence of the secondary graph statements added to the result.

We generate a primary graph at different confidence levels when a secondary graph is being used for validation, because it is too complex to vary the thresholds for both graphs. For each primary graph threshold, we show the performance of different secondary graph thresholds, focussing at the space below the primary threshold (as at the primary graph threshold the secondary and the primary graph are relatively similar, i.e. it is rare to have more than one relation above threshold between two mentions).

We chose .2 as it was the threshold producing the max F1 score on the devset for RelEx when measured at the mention level (as opposed to the entity level in the NLP Graph), which was .28. We chose .1 as it was the threshold we had guessed at in the program evaluation, before having any data to back it up.

After performing many tests and experiments tuning the system on the development set, we ran a single experiment on the test set.

5.2 Results and Analysis

We show the results of the six experiments (Fig. 1 - 4), each one comparing a baseline, using the primary graph only, to using the primary graph with the secondary graph for hypothesis validation (Fig. 1, 2 and 3). The experiments are named after the primary graph with a "+s" indicating validation in a secondary graph.

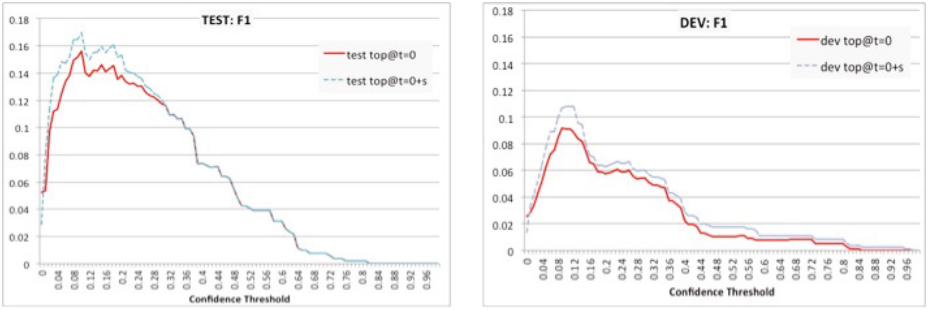


Fig. 1. F-Measure for the 0-threshold primary graph (with and without secondary graph for hypothesis validation) on test and dev sets

The Recall using the secondary graph for validation is always higher than without it, with a corresponding drop in Precision. F-measure improves across the confidence curve, with dramatic improvement in the interesting range below the primary graph threshold. Of particular interest is that the max F1 confidence point on the $top@.1 + s$

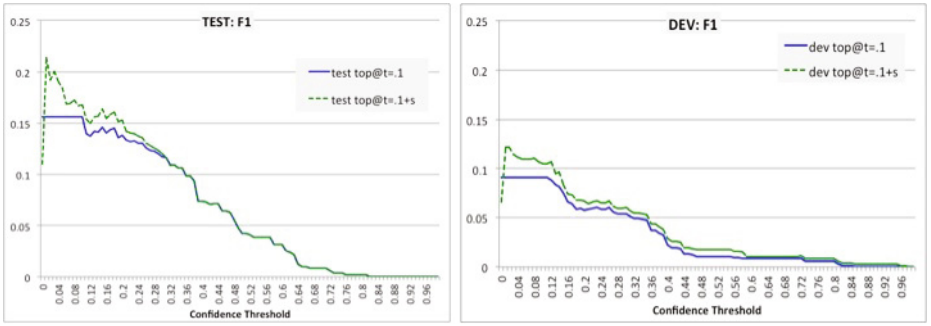


Fig. 2. F-Measure for the .1-threshold primary graph (with and without secondary graph for hypothesis validation) on test and dev sets

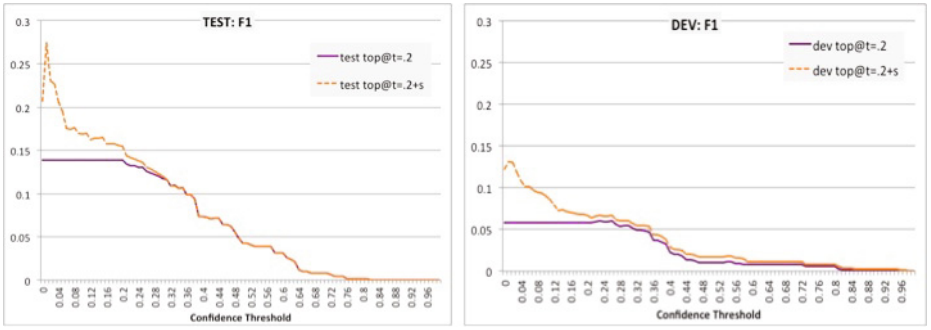


Fig. 3. F-Measure for the .2-threshold primary graph (with and without secondary graph for hypothesis validation) on test and dev sets

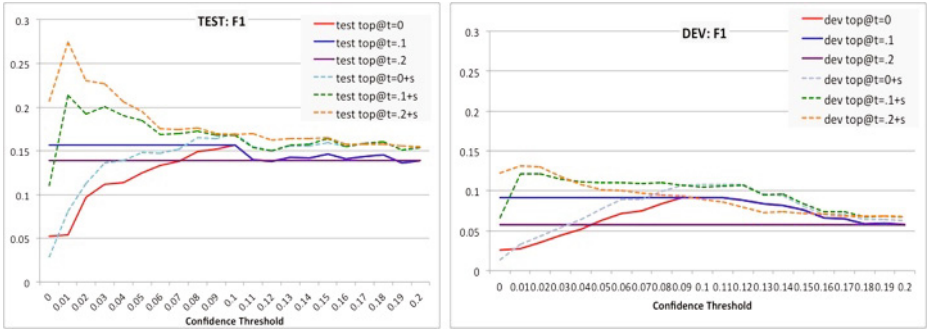


Fig. 4. A comparison of the six configurations on test and dev sets for the confidence range $0 - .2$

and $top@.2 + s$ experiments in both sets occur at a threshold of .01 (note again that below the primary threshold, the threshold value is only for statements added from the secondary graph) .

The threshold values themselves are artifacts of how the RelEx component computes its probabilities and was trained, but the result does strongly support our hypothesis that the probability of a relation holding between two mentions increases significantly if that relation would complete a conjunctive query result. The low max F1 threshold has certainly given us cause to more seriously consider the "cruft" produced at very low confidence levels that is normally discarded.

The results for $top@0$ in both sets also demonstrates that selecting a primary graph at some optimal threshold is better than a high Recall graph.

In the following table we summarize our results by choosing a configuration based on dev set performance and applying that configuration to the test set. The best performing configuration on the dev set was for $top@.2+s$ with a threshold of 0.01 on the secondary graph hypotheses. We can see that the difference at the chosen threshold on the test set significantly outperforms the baseline on the same set.

Table 2. Performance summary at chosen threshold

Doc Set	Configuration	P	R	F
Dev	top@.2 (baseline)	0.10	0.04	0.06
Dev	top@.2+s@.01	0.16	0.11	0.13
Test	top@.2 (baseline)	0.28	0.09	0.14
Test	top@.2+s@.01	0.32	0.24	0.27

All the differences from baselines to results with secondary graph validation, in the figures and tables, are statistically significant to six decimal places (paired t-test with one-tailed distribution).

Overall performance (P, R, F) of the test set is also significantly higher than overall performance of the development set. Upon inspection, this is due to an uneven split of low-performing relations, in particular the *locatedAt, isLedBy* and *employedBy* relations, between the two sets. We did not look at per-relation performance in balancing the relations, only at the total counts, since absolute performance was not our focus.

6 Related Work

Modifying queries to control Precision and/or Recall is an old and well-explored idea. On the one hand, the area of *Query Expansion* [6] investigates adding terms to queries as either conjuncts (often to improve Precision by disambiguating original query terms) or disjuncts (to improve Recall by complementing original query terms with related terms). Terms to add may be found through corpus analysis (for example, finding terms that co-occur with query terms), through relevance feedback or through existing semantic resources, such as thesauri or ontologies. On the other hand, *Ontology-based Query Relaxation* [7] seeks to improve Recall not by deleting query terms, but by relaxing

them through generalization in a taxonomy of relations and types. For example, an original query to find all people married to actors could be relaxed via *superrelations* (all people related to actors) or via *supertypes* (all people married to entertainers) or both. In our system, the *Knowledge Base hypothesis checker* allows more specific relations and types to satisfy hypotheses based on taxonomic rules from an ontology. It could also easily reverse those rules to allow evidence of more general relations and types to support hypotheses. Alternatively, in *Query Approximation* [8], a complex, conjunctive query is evaluated by starting with a simpler query (typically a single conjunct) and successively adding query terms. At each step, the set of solutions includes the solutions to the original, complex query, along with (successively fewer) false positives. The goal is to produce these approximate solutions quickly, while converging on exact solutions given more time.

In *Question Answering*, document retrieval is used to produce a smaller candidate subset for subsequent processing. Recall is preferred over Precision in performance tradeoff for document retrieval, and incorrect answers are filtered by downstream modules [9]. To improve Recall of document retrieval in question answering, Bilotti et al. [9] use inflectional variants of the terms in the question, to expand the query for retrieving relevant documents. The final query is a conjunction of disjunct clauses, consisting of original query term and its inflectional variants. Query expansion with inflectional variants improves Recall of the document retrieval. Mollá et al. [10] and Mcnamee et al. [11] also highlight the importance of high Recall information extractors in question answering. They focus on a high-Recall Named Entity recognizer that allows allocation of multiple Named Entity tags to the same string, to handle ambiguous entities by not committing to a single tag; leaving it to the final answer extraction and scoring module to filter out incorrect answers.

7 Conclusions

We have presented a framework for hypothesis generation and validation in RDF NLP graphs. The hypotheses are generated given a query, and propose new statements to add to the graph that would increase the number of results for the query. We have shown that a useful way to validate such generated hypotheses is to look in the secondary interpretations of the text considered by NLP components and ultimately discarded.

In previous work [1] we demonstrated the value of hypothesis generation and validation as a way to circumscribe reasoning tasks when employing rules and background knowledge. In future work we plan to combine background knowledge and reasoning with secondary graphs for validation. Another promising direction is to consider making or breaking coreference decisions as hypotheses, and validating them with a secondary graph and other knowledge.

One way of viewing our result is to change the role of NLP and semantic technologies in end-to-end systems. A traditional view of NLP is as a layer below that of knowledge representation from which information flows strictly upwards. In this view, NLP components must therefore make their best guess, without any knowledge of the specific task at hand. In this paper, we have presented an approach that improves on this traditional view, by allowing the knowledge layer to “go back” to the NLP layer

and ask for more targeted information about certain missing information that will help it complete its task. Specifically, we are able to target further consideration of discarded interpretations when they will meet some user need (i.e. provide new results to queries).

Acknowledgments. The authors gratefully acknowledge the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0172. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government. We would like to also thank Hideki Shima, Eric Riebling and Eric Nyberg from Carnegie Mellon University, as well as Sid Patwardhan and James Fan from IBM Research for their contribution to the implementation.

References

1. Barker, K., Fan, J., Welty, C.: Query driven hypothesis generation for answering queries over nlp graphs. In: IJCAI 2011 Workshop on Learning by Reading and its Applications in Intelligent Question-Answering (2011)
2. Bunescu, R., Mooney, R.: Subsequence kernels for relation extraction. In: Weiss, Y., Schölkopf, B., Platt, J. (eds.) Advances in Neural Information Processing Systems 18, pp. 171–178. MIT Press, Cambridge (2006)
3. Welty, C.A., Murdock, J.W.: Towards Knowledge Acquisition from Information Extraction. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 709–722. Springer, Heidelberg (2006)
4. Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A.A., Lally, A., Murdock, J.W., Nyberg, E., Prager, J., Schlaefer, N., Welty, C.: Building watson: An overview of the deepqa project. AI Magazine 31, 59–79 (2010)
5. Kalyanpur, A., Murdock, J.W., Fan, J., Welty, C.A.: Leveraging Community-Built Knowledge for Type Coercion in Question Answering. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 144–156. Springer, Heidelberg (2011)
6. Efthimiadis, E.N.: Query expansion. Annual Review of Information Systems and Technology 31, 121–187 (1996)
7. Hurtado, C.A., Poulovassilis, A., Wood, P.T.: Query Relaxation in RDF. In: Spaccapietra, S. (ed.) Journal on Data Semantics X. LNCS, vol. 4900, pp. 31–61. Springer, Heidelberg (2008)
8. Stuckenschmidt, H., van Harmelen, F.: Approximating Terminological Queries. In: Andreasen, T., Motro, A., Christiansen, H., Larsen, H.L. (eds.) FQAS 2002. LNCS (LNAI), vol. 2522, pp. 329–343. Springer, Heidelberg (2002)
9. Bilotti, M.W., Katz, B., Lin, J.: What works better for question answering: Stemming or morphological query expansion. In: Proceedings of the Information Retrieval for Question Answering (IR4QA) Workshop at SIGIR 2004 (2004)
10. Mollá, D., van Zaanen, M., Cassidy, S.: Named entity recognition in question answering of speech data. In: Colineau, N., Dras, M. (eds.) Proc. ALTW, vol. 5, pp. 57–65 (2007)
11. Mcnamee, P., Snow, R., Schone, P.: Learning named entity hyponyms for question answering. In: Proceedings of the Third International Joint Conference on Natural Language Processing (2008)

A Comparison of Hard Filters and Soft Evidence for Answer Typing in Watson

Chris Welty, J. William Murdock, Aditya Kalyanpur, and James Fan

IBM Research
cawelty@gmail.com, {murdockj,adityakal,fanj}@us.ibm.com

Abstract. Questions often explicitly request a particular type of answer. One popular approach to answering natural language questions involves filtering candidate answers based on precompiled lists of instances of common answer types (e.g., countries, animals, foods, etc.). Such a strategy is poorly suited to an open domain in which there is an extremely broad range of types of answers, and the most frequently occurring types cover only a small fraction of all answers. In this paper we present an alternative approach called TyCor, that employs soft filtering of candidates using multiple strategies and sources. We find that TyCor significantly outperforms a single-source, single-strategy hard filtering approach, demonstrating both that multi-source multi-strategy outperforms a single source, single strategy, and that its fault tolerance yields significantly better performance than a hard filter.

1 Introduction

Natural Language Question Answering (QA) systems allow users to ask questions in their own natural language, using their own terminology, and receive a concise answer [1]. While the success of Waston [2] was an important landmark, research in this area continues in the semantic web, as there is a clear gap between the complexity of logic-based semantic web representations and the capabilities of web users on a large scale [3]. A comprehensive survey can be found at [4].

Since the late 1990s, approaches to QA have always included a strong focus on typing. A common technique, possibly influenced by the way people answer questions, is to analyze the question for the type of thing being asked for, and then to restrict answers to that type. Many experimental systems have used such a *type-and-generate* approach, and rely on a process of *Predictive Annotation* [5], in which a fixed set of expected answer types are identified through manual analysis of a domain, and a background corpus is automatically annotated with possible mentions of these types before answering questions. These systems then map answer types into the fixed set used to annotate the corpus, and restrict candidate answers retrieved from the corpus to those that match this answer type using semantic search (IR search augmented with the ability to search for words tagged with some type). More recent semantic web QA systems similarly rely on the ability to map questions into ontology terms, and assume the ontology types cover all the questions and answers.

Most existing open-domain QA systems also use a single source of typing information, or in some cases two sources (where one is WordNet [6]), and a single strategy for

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 243–256, 2012.

analyzing questions for their answer type. While it is well known that multiple sources would provide more coverage, most systems avoid exploiting multiple sources because they believe it is necessary to perform ontology alignment between them.

In this paper we compare the single source, single strategy, type-and-generate approach to a multi-source, multi-strategy *generate-and-type* approach called TyCor, in which candidate answers are initially produced without use of answer type information, and subsequent stages score the degree to which the candidate answer's type can be coerced into the Lexical Answer Type (LAT) of the question. To isolate the comparison, we reduce the type-and-generate approach to hard filtering of answer candidates by type, and demonstrate that type coercion (TyCor) outperforms it on a large set of questions from the Jeopardy! quiz show, without requiring ontology alignment.

2 Background

An important part of Watson, the QA system that defeated the best human players on the American television quiz show *Jeopardy!*, is the way it identifies the type of the answer from the question and evaluates candidate answers with respect to that type. At the very beginning of the Watson project, we used a type-and-generate approach for generating candidate answers, simply because that is what we had. However, in our early analysis of the domain of questions from the TV quiz show Jeopardy!, we found three main problems that led us to consider an alternative.

To begin with, the design of the *Jeopardy!* game makes confidence an important part of performance, experts only answer when they think they know the answer and have to compete with the other players to get to be the one that answers. In order to account for the impact of confidence on performance, we adopted a metric of precision at 70% answered (P@70) with an expert level performance target of 85% P@70. In other words, our target was to perform at 85% precision on the 70% of answers in which the system was most confident.

The first problem we had with the type-and-generate approach resulted from analysis of 20K questions from our domain: we observed a very long tail of types (see 2). While the type system for our predictive annotation engine was among the largest in the community (over 100 types), these did not cover even half the questions. There were roughly 5K different type words used in the 20K questions, more than half of these occurred fewer than three times in the question set, and roughly 12% occurred once. As we continued to evaluate on hidden data, we found the 12% number to be constant: new types were being introduced at this rate (one in eight questions on average). In addition, 15% of questions did not identify the answer type with a specific word. This data seems immediately to contraindicate the use of a predetermined set of answer types.

The second problem with type-and-generate is the reliance on question analysis to map type *words* in the question to the predetermined set of recognized answer types. Human language is remarkably rich when it comes to describing types; nearly any word can be used as a type, especially in questions, e.g.:

– Invented in the 1500s to speed up the game, this *maneuver* involves 2 pieces of the same color. (Castling)

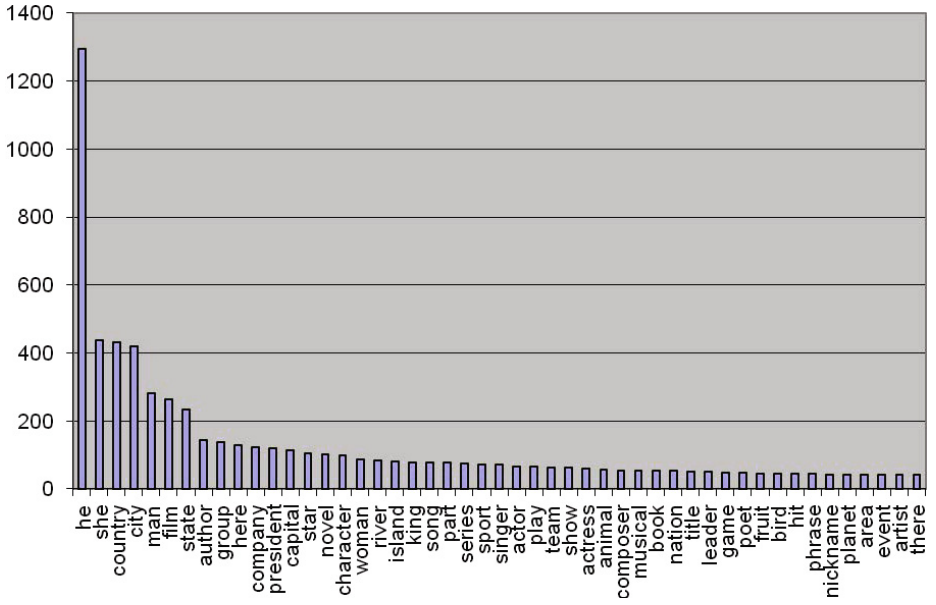


Fig. 1. The distribution of the 30 most frequent LATs in 20K Jeopardy! questions

- The first known air mail service took place in Paris in 1870 by this *conveyance*. (Hot-air balloon)
- In 2003 this Oriole *first sacker* was elected to the Baseball Hall of Fame. (Eddie Murray)
- Freddy Krueger was introduced in this 1984 *scarefest*. (A Nightmare on Elm Street)
- When hit by electrons, a phosphor gives off electromagnetic energy in this *form*. (light)
- Whitney's patent for *this* revolutionized the garment industry. (the cotton gin)

Given this variability, one of the intuitive problems with relying on predictive annotation is that we cannot reliably map from these words, which we call the *lexical answer type* (LAT), to the predetermined set of known answer types. Even in the cases where there should be a mapping, as in the case of scarefest, or conveyance, there is no way to accurately predict for unseen questions what LATs might be used for the known types.

The third and final problem was that the type-and-generate approach itself was a single point of failure that quite simply prevented us from reaching human expert performance. The basic idea (analyze the question and determine the answer type from a predetermined set, find answers in the background corpus that have been labelled with that type, *then* process those candidates with other forms of evidence) limits the ultimate performance of the end to end system to the recall of the question analysis times the recall of the predictive annotation. In our case, the recall of our predictive annotator was estimated at about 70% *for the types it knew*, and the recall of the question analysis was similar, leaving us with a maximum performance of under 50% absolute for less that half the questions, and then we had to have an approach for the other half.

While we may have had some hope of improving the recall of question analysis and predictive annotation, these problems and the need to implement some way to handle questions in the long tail of LATs anyway, forced us to rethink the approach. It seemed to us that we needed to be open and flexible about types, treating them as a property of a question and answer combined. In other words, instead of finding candidates of the right type, we should find candidates (in some way) and judge whether each one is of the right type by examining it in context with the lexical answer type from the question. Furthermore, we needed to accommodate as many sources as possible that reflect the same descriptive diversity as these questions. Our hypothesis was that, to reach expert human performance, we needed a system design in which any part of the system could fail to produce the proper result, without preventing the overall system from succeeding.

3 TyCor

We use the term *Type Coercion* to refer to a process of determining whether it is possible to interpret a candidate answer that is consistent with the type requested by the question. Note that this term is also used by Pustejovsky [7] as a linguistic phenomenon in which a speaker imposes an abstract type onto a word by context. For example, saying that someone finished a book coerces an interpretation of "book" as an event, while the word "book" might be interpreted as a physical object in other contexts. Our use of the same term is largely coincidental; we describe a process of interpretation not a generative theory of language. However, we share with Pustejovsky the perspective that types can be dynamic and influenced by context.

3.1 Question Analysis

TyCor depends on getting the type word from the question, and gathers evidence for each candidate answer that it is of that type. The type word is not interpreted according to some predefined type system or ontology, unlike with type-and-generate approaches to question answering. We call the uninterpreted type word from the question the *lexical answer type* (LAT). If the LAT has been mapped into some predefined type system, we refer to the resulting type as the *semantic answer type* (SAT).

LAT recognition is easier than mapping to a semantic type, and though imperfect our LAT detection measures above .9 F1 on 20K randomly selected questions [8]. Like all parts of our system, LAT detection includes a confidence, and all type scores are combined with LAT confidence.

3.2 Candidate Generation

Our approach to candidate generation, driven by the observations discussed above, was to raise recall well above .8, and expend more effort on answer scoring to promote correct answers. A full discussion of candidate generation is beyond the scope of this paper, and can be found elsewhere [9], but improving this step to .85 recall for the top 500 answers was significant.

3.3 Answer Scoring

One of the significant differences between our approach and that of many previous QA systems is the manner in which candidate answers are generated, and that the processing of type information has been moved from candidate generation and search to answer scoring. Answer scoring is the step in which all candidates, regardless of how they were generated, are evaluated. During the answer scoring phase, many different algorithms and sources are used to collect and score evidence for each candidate answer with respect to the question. Type information is just one kind of evidence that is used for scoring, there are over 100 other dimensions of evidence including temporal and spatial constraints, n-grams, popularity, source reliability, skip-bigrams, substitutability, etc. Each answer scoring method is independent from all others, and may fail on any particular question-answer pair. The general idea is that across all methods, more will succeed for the correct answer than any other. In the end, this idea turned out to be true often enough to win.

3.4 Final Answer Merging and Ranking

With over 100 different methods for retrieving and scoring evidence for candidate answers, an overall determination of the final answer must combine the scores from each scoring algorithm for each answer in a way that weights each score as appropriate for the context given by the question [10]. In general, using a large number of blind training examples (roughly 3K questions with answers), we learn a set of context-dependent vector models in which each score corresponds to a dimension that is assigned a weight and combined using a logistic function. The contexts are mainly based on properties of the question: is there a LAT at all, is the question decomposed, etc.

An important quality of answer scores, due to the way the scores are combined, is that they exhibit monotonic behavior. That is, they should consistently increase (or decrease) with the probability of the answer being correct. For TyCor components, this requirement meant special attention had to be paid in the framework for estimating and modeling error. To accomplish this, we broke all TyCor components into four basic steps, and collected error rates for these steps. This allowed us to make more fine grained predictions of confidence in typing.

3.5 TyCor Framework

TyCor is a class of answer scoring components that take a LAT and a candidate answer, and return a probability that the candidate's type is the LAT. Note that since language does not distinguish between instantiation and subclass, the TyCor components must allow for this, and given a candidate answer that refers to a class, TyCor should give it a high score if it can be interpreted as a subclass or instance of the LAT. As mentioned above, LAT detection produces a confidence, so each (answer,LAT) score is modified by the LAT confidence.

Each TyCor component uses a source of typing information in some cases (see e.g. Lexical TyCor in the next section) this source is the answer itself and performs four

steps, each of which is capable of error that impacts its confidence. A more complete description of the framework can be found in [11] and [12], but briefly:

Entity Disambiguation and Matching (EDM): The most obvious, and most error-prone, step in using an existing source of typing information is to find the entity in that source that corresponds to the candidate answer. Since the candidate is just a string, this step must account for both polysemy (the same name may refer to many entities) and synonymy (the same entity may have multiple names). Each source may require its own special EDM implementations that exploit properties of the source, for example DBpedia encodes useful naming information in the entity id. EDM implementations typically try to use some context for the answer, but in purely structured sources this context may be difficult to exploit.

Predicate Disambiguation and Matching (PDM): Similar to EDM, the type in the source that corresponds to the LAT found. In some sources this is the same algorithm as EDM, in others, type looking requires special treatment. In a few, especially those using unstructured information as a source, the PDM step just returns the LAT itself. In type-and-generate, this step corresponds to producing a semantic answer type (SAT) from the question. PDM corresponds strongly to notions of word sense disambiguation with respect to a specific source.

Type Retrieval (TR): Once an entity is retrieved from the source, if applicable the types of that entity must be retrieved. For some TyCors, like those using structured sources, this step exercises the primary function of the source and is simple. In others, like unstructured sources, this may require parsing or other semantic processing of some small snippet of natural language.

Type Alignment (TA): The results of the PDM and TR steps must then be compared to determine the degree of match. In sources containing e.g. a type taxonomy, this may include checking the taxonomy for subsumption, disjointness, etc. For other sources, alignment may utilize resources like wordnet for finding synonyms, hypernyms, etc. between the types.

3.6 Multi Source Multi Strategy TyCor

We have implemented more than ten different TyCor components for scoring candidate answers. Some of our TyCor components share algorithms but use different sources of evidence, others use different algorithms on the same sources. The algorithms mainly involve different approaches to the four TyCor steps as appropriate for the source, with an eye towards accurately accounting for the error in the steps (most notably EDM and alignment) to produce a meaningful score. The sources we use range from DBpedia, WordNet, Wikipedia categories, Lists found on the web (e.g. list of nobel prize winners), as well as the first sentence of Wikipedia articles, lists of common male and female names, specially mined databases of people and their genders, and mined results of is-a patterns [13] from large corpora.

Generally speaking, TyCor scores range from [-1,1]; negative scores are interpreted as evidence that a candidate is not of the right type, positive scores that it is, and a 0 score is interpreted as unknown. For example, if a candidate is simply not known by a source, this doesn't constitute evidence that the answer is not of the right type.

Most TyCor methods do not even include the ability to collect negative evidence, those that do are indicated below.

A full discussion of all the components is beyond the space limitations of this paper. A brief overview follows.

Yago: Many candidate answers in our domain are titles of Wikipedia articles. Each of these is an entity in DBpedia, a linked open data source compiled automatically from Wikipedia infoboxes and article templates. Entities (articles) in DBpedia have types represented in RDF from Yago [14], a semi-automatically constructed type taxonomy based on WordNet, corpus analysis, and Wikipedia. In addition, we have added roughly 200 disjointness constraints (e.g. a Person is not a Country) at high levels in the taxonomy. Using a special purpose reasoner to check for subsumption and disjointness, Yago Tycor can produce negative evidence when a candidate matches only types that are disjoint from all the types matching the LAT.

Intro: The first sentence of all Wikipedia articles identifies some types for the entity described in the article, e.g. Tom Hanks is an American actor, producer, writer, and director. Intro TyCor utilizes a special source mined from these intro passages and scores LAT matches, using WordNet synonyms and hypernyms for Type alignment.

Gender: Uses a custom source of data mined from articles about people by determining which pronouns are most commonly used to refer to the person, scores the degree to which a candidate answer is of the appropriate gender, or not. Can produce negative evidence if the LAT indicates one gender and the answer is found to be of the other.

ClosedLAT: Certain LATs identify types with enumerable lists of instances, such as Countries, US States, US Presidents, etc. When such a list is available, this TyCor component is capable of producing a negative type score for candidate answers that are not in the list. Of course, as with everything described here, confidence is never perfect due to name matching issues and the possibility that the LAT is used in a non-standard way. For example, the mythical country of Gondor is not on our closed list, but could conceivably be the answer to a country-LAT question. Based on the domain analysis in Figure 1, we selected the 50 most frequent closed LATs and developed lists of their instances.

Lexical: Occasionally LATs specify some lexical constraint on the answer, like that it is a verb, or a phrase, or a first name, etc. Lexical TyCor uses various special-purpose algorithms based on the LAT for scoring these constraints. Passage: When a candidate answer is extracted from a passage of text, occasionally that passage includes some reference to the candidate's type, e.g. Actor Tom Hanks appeared at the premier of his new film. Passage Tycor uses WordNet for type alignment to match the LAT to the type word in the passage.

Identity: Some candidate answers contain the LAT as part of their name it can be quite embarrassing for a system to miss King Ludwig as a king just because he isn't on a list of Kings. Identity TyCor uses WordNet to match the LAT to any part of the candidate string itself.

NED: The NED TyCor uses the engine previously used for predictive annotation. Although our approach supercedes pure predictive annotation for candidate generation, it does not throw it away. The NED engine recognizes over 100 SATs, most of which are among the top 100 LATs. It uses a special purpose rule base PDM to map from

LATs to SATs, and recognizes candidates as being of the right SAT mainly through large manually curated list and patterns.

WordNet: WordNet is used primarily in other TyCor components to assist in the type alignment phase, however it does contain some limited information about well known entities such as famous scientists, a few geographic and geopolitical entities, etc. It also has high coverage of biological taxonomies. This TyCor implementation has very high precision but low recall.

WikiCat: Wikipedia articles are frequently associated with categories that more or less match, in style and content, the linguist ability to say that one thing has some topic association. All these categories are stored in DBpedia. The WikiCat TyCor uses primarily the headword of all the category names for an entity, and performs type alignment using WordNet synonyms and hypernyms. Wikicat does not use the category structure (e.g. subcategory) at all, as this adds too much noise.

List: Wikipedia and many other web sources contain lists of things associated in some way, like List of Argentinean Nobel Prize Winners. We collect these lists, and like WikiCat map only the headwords to the LATs.

Prismatic: Prismatic [15] is a repository of corpus statistics such as all subject-verb-object tuples, or all subject-verb-preposition-object tuples, that allows counting queries (e.g. how many SVPO tuples with stars as the verb). Prismatic Tycor measures the frequency with which the candidate answer is directly asserted to be an instance of the lexical answer type using is-a patterns [13].

4 Experiment

We performed a series of experiments to validate our hypothesis that the type-and-generate (TaG) approach exemplified by TyCor would lead to improved performance over a generate-and-type (GaT) approach to open-domain QA. In particular, we were interested in validating that generating only candidate answers believed to be of the right semantic type before seeing the question would limit system performance.

4.1 Experimental Setup

The full QA system we ran this experiment with is the version of Watson that participated in the televised exhibition match. It performs at 71% accuracy overall and with a confidence estimation capability that allows it to perform at over 85% P@70 (precision at 70% answered). This combination of overall accuracy and confidence estimation is unprecedented in the community. The system was tested in a public display on blind questions which were held by independent auditors to ensure the test was fair and truly blind.

We ran a second set of experiments using a smaller “lite” version of Watson that removes all the answer scoring components (except, where applicable, the TyCor component being tested), relying on only the scores resulting from candidate generation methods, primarily (though not exclusively) search. This set of experiments was intended to demonstrate that the relative difference between TaG and GaT changes with the overall performance.

While the performance numbers are interesting in themselves (the lite version of the system already performs at levels that would make it one of the top 2 open domain QA systems at TREC [16]) it is simply beyond the scope of this paper to describe how they were obtained. For the purposes of this paper, within each set of experiments (full system and lite system) all aspects of the system were held constant except for the approach to answer typing. Thus it is the relative performance numbers we focus on.

For the full and lite versions of the system, we ran four experiments:

No Typing: Uses no typing components in scoring answers, although since the LAT is a search keyword, some type information does make it into candidate generation and scoring.

Tycor: Uses the full set of TyCor components (described above) as separate answer scorers.

NED Tycor: Uses only the NED based TyCor component as an answer scorer, to illustrate the effectiveness of predictive annotation as a source of evidence.

TaG: We simulate the type-and-generate approach by using the NED TyCor component as a hard filter on candidate answers that are scored. Candidates are generated in the same way as other versions of the system, but all those that do not match the SAT according to the NED annotator are filtered out. We chose this approach over using actual semantic search for candidate generation in order to hold the search engine itself constant, since the one we used in Watson did not support semantic search [9]. As can be seen from the lite version of the system, the search component alone produces 50% accuracy and 64% P@70, which is significant.

All experiments were run on 3500 blind questions from past Jeopardy! games. Candidate generation recall for all versions is the same, at 87% in the top 500.

4.2 Results

Table 1 shows the relative performance of the four configurations of the typing components described above on the lite and full versions of the system. The lite version has no other answer scoring except search and candidate generation. The full version uses all answer scoring in the system that performed in the exhibition match. Accuracy (precision @ 100% answered), is shown along with P@70. All results within columns are different with statistical significance at $p < .01$.

The main comparisons are between the TaG and TyCor versions of the system, and between the NED TyCor and TaG, in both the lite and full systems, although it is clearly worth noting that TaG draws down the performance of the system even with no typing at all.

Table 1. Relative Performance on Lite and Full QA Systems

	Lite System		Full System	
	Accuracy	P@70	Accuracy	P@70
No Typing	50.0%	63.4%	65.5%	81.4%
NED TyCor	53.8%	67.8%	67.9%	84.2%
TaG	45.9%	62.1%	55.3%	74.9%
TyCor	58.5%	75.0%	70.4%	87.4%

4.3 Analysis and Discussion

Throughout this paper we have been careful to distinguish the type-and-generate approach (TaG) from predictive annotation itself. Predictive annotation refers only to the process of annotating a background corpus with types from a predefined set of possible semantic answer types. The TaG approach is one in which questions are analyzed to produce the semantic answer type, and candidate generation produces only candidate answers which were identified as being of the right type during the predictive annotation step. Note that while there is a clear similarity between predictive annotation and named entity detection (NED), predictive annotation tends to favor recall over precision, and is generous in labeling mentions in a corpus; often mentions will have multiple type labels.

In the full system, the TaG configuration performs at 55.3% accuracy and 74.9% P@70, compared to 70.4% and 87.4% resp. for TyCor. It could be argued that the multi-source multi-strategy (hereafter, multi-strategy) aspect of TyCor accounts for this difference and not the TaG vs. generate-and-type (GaT) approaches, and that a multi-strategy approach could have been used during predictive annotation itself. If true, this would mean the only difference between the two is one of efficiency at question answering time. In reality, aspects of the multi-strategy approach make it computationally impractical to annotate the entire corpus this way, but to further isolate the comparison we ran the system with only the NED TyCor component, which is the same component used to hard filter answers in the TaG configuration.

The results demonstrate that the TaG approach is a hindrance to a high performing QA system. The idea of predictive annotation itself, however, is a helpful one, as shown by the NED TyCor rows. In both cases, having some evidence for typing, even when (as in this case) it covers roughly 60% of the questions, is more helpful than no typing information at all, provided that evidence is used in answer scoring, not as a filter. Indeed, the NED TyCor component is consistently among the top 4 performing TyCor components over large sets of questions. The main issue it has, which is more than made up for by the full complement of TyCors, is that it works only on types in the head of the LAT curve (see 2).

The main reason the GaT approach outperforms TaG is the fault tolerance of the answer scoring phase in DeepQA. It is possible for a correct answer to "win out" and become the top answer even when the NED TyCor believes the answer is of the wrong type. In TaG, when question analysis or predictive annotation make mistakes, there is no way to recover from them. This is borne out further by the difference between the full system and the lite, where there are more kinds of answer scoring and thus more ways in which other evidence can overcome typing failures; the relative and absolute differences in the full system for TaG vs. GaT are more. Interestingly, until our system reached a performance level of about 50% overall accuracy, we were not able to validate the TaG vs. GaT hypothesis experimentally, due mainly to the lack of sufficient other evidence to overcome the typing failures. This seems to indicate that, for systems performing at 50% or less, TaG is a reasonable approach. This performance level characterized all but the top performing system at the TREC QA evaluations [16].

The results also show the relative impact of the multi-strategy TyCor approach on the system. In the lite system, full TyCor adds 15.7% relative (8.5% absolute) to accuracy

and 16.8% relative to P@70 (11.6% absolute). For the full system, TyCor adds 7.2% relative accuracy improvement (4.9% absolute), and 7.1% relative P@70 (6% absolute). All components in our system (there are over 100) see the same sort of diminished relative impact in the full system compared to the lite, this is due to the fact that many of the scoring components overlap. For example, another scoring component counts the n-gram frequency of terms in the question with the candidate answer. Since the LAT is one of the terms in the question, the ngram score can provide partial information about typing.

That the multi-strategy approach outperforms a single strategy for typing candidate answers should come as no surprise, but it is interesting to note that the DeepQA architecture facilitates the combination easily. Training the system only against a question-answer ground truth (ie as opposed to a task-specific ground truth of candidate answers and types), DeepQA is able to effectively combine the 14 different TyCor implementations to produce significantly better results than against any of them in isolation. Here we show only the comparison to the NED TyCor in isolation, a full set of experiments showing the relative performance of each in isolation can be found in [12].

5 Related Work

QUARTZ [17] is a QA System that uses WordNet as the background knowledge base for mapping answer types expressed in the question. This approach mitigates the type coverage issue in earlier QA systems due to the conceptual breadth of WordNet. The mapping from answer type to WordNet synset, which is essentially a Word Sense Disambiguation (WSD) problem, is done using statistical machine learning techniques. Having obtained a WordNet synset T for the answer type, the system estimates a set of complementary-types C(T) in WordNet (typically considering siblings of T). A given candidate answer is then determined to be of the correct type if it has a stronger correlation to type T than to the types in C(T), where the correlation is computed using Web data and techniques like mutual information (MI) e.g. how often does the candidate answer co-occur with the type across a collection of Web documents. In [18] the approach has been taken a step further by combining correlation-based typing scores with type information from resources such as Wikipedia, using a machine-learning based scheme to compute type validity.

Both [18] and [17] use a similar approach to type-coercion in DeepQA in that they defer type-checking decisions to later in the QA pipeline and use a collection of techniques and resources (instead of relying on classical NERs) to check for a type match between the candidate and the expected answer type in the question. However, that is where the similarity ends. A fundamental difference in our approach is that the type match information is not used as a filter to throw out candidate answers, instead, the individual TyCor scores are combined with other answer scores using a weighted vector model. Also, our type-coercion is done within a much more elaborate framework that separates out the various steps of EDM, PDM, Type Alignment etc, and the intermediate algorithms (and resources) used in these steps are far more complex and varied – having either much more precision, and/or much broader scope compared to existing work, and a precise model of error. For example, the only use of Wikipedia content for

type inference in [18] is through a shallow heuristic that searches for the mention of the expected answer type on the Wikipedia page of a candidate answer (mapped using an exact string match to the page title) and makes a Yes/No decision for the type validity based on this finding. In contrast, in our Wikipedia-based TyCors we use an EDM algorithm to map the candidate answer string to a Wikipedia page using a variety of resources such as Wikipedia redirects, extracted synonym lists, link-anchor data etc, and then use different kinds of type information expressed in Wikipedia, such as lexical types in the introductory paragraphs, Wikipedia categories etc. Similarly, while [17] uses the notion of complement-type sets, which are approximated using heuristics such as sibling-types, we define explicit disjoint types in the Yago Ontology and use disjointness information to down weigh candidate answers whose types are disjoint with the LAT.

An approach that combines slightly softens the type and generate approach by applying semantic answer type constraints to passage ranking is presented in [19]. Like our system, search terms are extracted from questions for a search engine which returns ranked sets of passages. These passages are then pruned by removing all passages that do not contain terms labeled with the semantic answer type detected in the question. The approach shows improvement in passage ranking metrics, but QA performance is not evaluated. Such an approach could be used in our system, as removing passages without detected answer types in them is subtlety different than removing answers themselves; candidate answers can be generated from passages by e.g. extracting all noun phrases, whether they have the right annotation labels or not. Still, our analysis here suggests that this would probably only help for lower performing QA systems.

A similar approach to our combination of NED and WikiCat is presented in [20]. The traditional type-and-generate approach is used when question analysis can recognize a semantic answer type in the question, and falls back to Wikipedia categories for candidate generation, using it as a hard filter instead of predictive annotation. In our approach we assume any component can fail, and we allow other evidence, both from other TyCor components and from other answer scoring components, to override the failure of one particular component when there is sufficient evidence.

6 Conclusion

Answer typing is an important component in a question answering (QA) system. The majority of existing factoid QA systems adopt a type-and-generate pipeline that rely on a search component to retrieve relevant short passages from the collection of newswire articles, and to extract and rank candidate answers from those passages that match the answer type(s) identified, based on the question, from a pre-constructed and fixed set of semantic types of interest. For example, the semantic answer type for the question "What city is was 2008 World Sudoku Championship held in?" is *City*, and the candidate answer set for this question typically consists of all cities extracted from a relevant passage set by a named entity recognizer (NER). This approach suffers from two main problems.

First, restricting the answer types to a fixed and typically small set of concepts makes the QA system brittle and narrow in its applicability and scope. Such a closed-typing

approach does not work for open-domain Question Answering, and in particular the Jeopardy! problem, where answer types in questions span a broad range of topics, are expressed using a variety of lexical expressions (e.g. scarefest when referring to the semantic type horror movie) and are sometimes vague (e.g. form) or meaningless (e.g. it).

Second, the QA system performance is highly dependent on the precision and recall of the NERs used, as they act as candidate selection filters, and the system has no way to recover from errors made at this stage.

We have presented an approach to handling answer types in open domain question answering systems that is more open and flexible than the commonly used type-and-generate approach. Our generate-and-type approach does not rely on a fixed type system, uses multiple strategies and multiple sources of typing information, gathers and evaluates evidence based on the type words used in the question, and is not a hard filter. Our approach is broken into four basic steps, which have allowed us to more accurately model and predict the error of typing statements, which increases the ability of the typing system to inform the confidence in final answers. We compared our approach to type-and-generate within a high performance QA system and found a significant difference in performance, both in the overall accuracy and the ability to estimate confidence.

Acknowledgements. Numerous people contributed to Watson.

References

1. Hirschman, L., Gaizauskas, R.: Natural language question answering: the view from here. Nat. Lang. Eng. 7(4), 275–300 (2001)
2. Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A., Lally, A., Murdock, J.W., Nyberg, E., Prager, J., Schlaefer, N., Welty, C.: Building watson: An overview of the deepqa project. AI Magazine, 59–79 (2010)
3. Kaufmann, E., Bernstein, A., Fischer, L.: NLP-Reduce: A "naive" but Domain-independent Natural Language Interface for Querying Ontologies (2007)
4. Lopez, V., Uren, V., Sabou, M., Motta, E.: Is question answering fit for the semantic web? a survey. Semantic Web? Interoperability, Usability, Applicability 2(2), 125–155 (2011)
5. Prager, J., Brown, E., Coden, A., Radev, D.: Question-answering by predictive annotation. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2000, pp. 184–191. ACM, New York (2000)
6. Miller, G.A.: Wordnet: a lexical database for english. Commun. ACM 38(11), 39–41 (1995)
7. Pustejovsky, J.: Type coercion and lexical selection. In: Semantics and the Lexicon. Kluwer Academic Publishers, Dordrecht (1993)
8. Lally, A., Prager, J.M., McCord, M.C., Boguraev, B.K., Patwardhan, S., Fan, J., Fodor, P., Chu-Carroll, J.: Question analysis: How watson reads a clue. IBM Journal of Research and Development 56(3.4), 2:1 –2:14 (2012)
9. Chu-Carroll, J., Fan, J., Boguraev, B.K., Carmel, D., Sheinwald, D., Welty, C.: Finding needles in the haystack: Search and candidate generation. IBM Journal of Research and Development 56(3.4), 6:1 –6:12 (2012)
10. Gondek, D., Lally, A., Kalyanpur, A., Murdock, J., Duboue, P., Zhang, L., Pan, Y., Qiu, Z., Welty, C.: Finding needles in the haystack: Search and candidate generation. IBM Journal of Research and Development 56(3.4), 14:1 –14:12 (2012)

11. Kalyanpur, A., Murdock, J.W., Fan, J., Welty, C.A.: Leveraging Community-Built Knowledge for Type Coercion in Question Answering. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 144–156. Springer, Heidelberg (2011)
12. Murdock, J.W., Kalyanpur, A., Welty, C., Fan, J., Ferrucci, D.A., Gondek, D.C., Zhang, L., Kanayama, H.: Typing candidate answers using type coercion. IBM Journal of Research and Development 56(3.4), 7:1 –7:13 (2012)
13. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th Conference on Computational Linguistics, COLING 1992, vol. 2, pp. 539–545. Association for Computational Linguistics, Stroudsburg (1992)
14. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, WWW 2007, pp. 697–706. ACM, New York (2007)
15. Fan, J., Kalyanpur, A., Gondek, D.C., Ferrucci, D.A.: Automatic knowledge extraction from documents. IBM Journal of Research and Development 56(3.4), 5:1 –5:10 (2012)
16. Voorhees, E. (ed.): Overview of the TREC 2006 Conference, Gaithersburg, MD (2006)
17. Schlobach, S., Ahn, D., de Rijke, M., Jijkoun, V.: Data-driven type checking in open domain question answering. J. Applied Logic 5(1), 121–143 (2007)
18. Grappy, A., Grau, B.: Answer type validation in question answering systems. In: Adaptivity, Personalization and Fusion of Heterogeneous Information, RIAO 2010, Paris, France, France, Le Centre De Hautes Etudes Internationales D'Informatique Documentaire, pp. 9–15 (2010)
19. Aktolga, E., Allan, J., Smith, D.A.: Passage Reranking for Question Answering Using Syntactic Structures and Answer Types. In: Clough, P., Foley, C., Gurrin, C., Jones, G.J.F., Kraaij, W., Lee, H., Mudoch, V. (eds.) ECIR 2011. LNCS, vol. 6611, pp. 617–628. Springer, Heidelberg (2011)
20. Buscaldi, D., Rosso, P.: Mining Knowledge from Wikipedia from the question answering task. In: Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006), Genoa, Italy (2006)

Incorporating Semantic Knowledge into Dynamic Data Processing for Smart Power Grids

Qunzhi Zhou, Yogesh Simmhan, and Viktor Prasanna

University of Southern California, Los Angeles, USA
{qunzhizh,simmhan,prasanna}@usc.edu

Abstract. Semantic Web allows us to model and query time-invariant or slowly evolving knowledge using ontologies. Emerging applications in Cyber Physical Systems such as Smart Power Grids that require continuous information monitoring and integration present novel opportunities and challenges for Semantic Web technologies. Semantic Web is promising to model diverse Smart Grid domain knowledge for enhanced situation awareness and response by multi-disciplinary participants. However, current technology does pose a performance overhead for dynamic analysis of sensor measurements. In this paper, we combine semantic web and complex event processing for stream based semantic querying. We illustrate its adoption in the USC Campus Micro-Grid for detecting and enacting dynamic response strategies to peak power situations by diverse user roles. We also describe the semantic ontology and event query model that supports this. Further, we introduce and evaluate caching techniques to improve the response time for semantic event queries to meet our application needs and enable sustainable energy management.

Keywords: Semantic Web, complex event processing, smart grid.

1 Introduction

The power grid is undergoing rapid modernization into a Smart Grid through the integration of digital and information technologies. This trend is worldwide [1]. Conventional meters which record the accumulative power usage at monthly base are being replaced by smart meters which report power consumption at minutes interval. In Europe, for example, Italy and Sweden are approaching 100 percent deployment of smart meters for consumers. In U.S., the largest municipal utility, the Los Angeles Department of Water and Power (LADWP), has begun to expand its advanced metering infrastructure. At the building level, ambient sensors and smart appliances, such as HVAC sensors, occupancy sensors and plug-in electric vehicles (PEVs) are being integrated into building control systems. These various information sources provide fine-grained monitoring and control capability of power use activities, in both spatial and temporal scales. However, transforming this capability to actionable knowledge is challenging, due to the complexity of both information and interactions.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 257–273, 2012.

Demand response (DR) is a cornerstone application in Smart Grids that aims to curtail power load during peak load periods. This reduces the captive power generation required by a utility for reliable operations by shaping the power usage to remain relatively constant over time. Existing DR approaches are statically planned through time-based pricing incentives for voluntary curtailment by customers. But the intermittent nature of renewable generation like wind power and changes in energy use pattern hampers this static approach.

Dynamic DR [18] supplements traditional DR approaches by leveraging real-time information for online decision making, enabling opportunistic curtailment based on the current situation. However, as an application layer software, a dynamic DR system has to deal with the heterogeneous and constantly evolving Smart Grid infrastructure. Further, the curtailment decision and enactment is distributed, engaging participants like utility operators, facility managers, building occupants and household customers. No single person has a holistic sense of the information space and concepts used for DR decisions.

Semantic Web provides an ontology-based extensible framework that allows information to be shared and reused across application and domain boundaries. It has been used for information integration in domains such as health care [10,20], biology [21,16] and transportation [11]. However these approaches mostly process time-invariant or slowly evolving semantic data.

We provides a framework that adopts semantic knowledge in stream processing and applies it to dynamic DR in Smart Grids, specifically the *USC Campus Micro-Grid*. We combine complex event processing (CEP) with Semantic Web to facilitate high level user application design. The semantically enhanced CEP system ties in with our semantic repository that hosts comprehensive information on the campus micro-grid. These are intended to be leveraged by the campus energy center, building managers, staff and students, and their applications, for campus energy management. Semantic technologies are central to meet the organic growth of information and infrastructure diversity and keep them accessible for easy use. Our key contribution in this paper are:

Semantics in Use in Micro-grid. We discuss specific uses and benefits of semantic technologies for micro Smart Grid applications and the participants.

Semantics for Complex Event Processing. We describe unique benefits offered by semantics for complex event processing.

Semantic CEP (SCEP) Optimizations. We introduce caching techniques for efficient semantic query processing over event data streams, and evaluate them using semantic CEP queries used for dynamic DR on campus.

The rest of the paper introduces dynamic DR and semantic information model for the campus Micro Grid (§ 2), describes the uses of semantic technologies in the Micro Grid (§ 3), presents our semantic CEP model and DR patterns (§ 4), discusses and evaluates our caching optimizations for query processing (§ 5), reviews related work (§ 6) and presents our conclusions (§ 7).

2 Background

Our work is done as part of the Los Angeles Smart Grid Demonstration Project, where the USC Campus is a *Micro-Grid* testbed for evaluating Smart Grid technologies and software tools. Smart Grids have two characteristics relevant to Semantic Web: Diversity and Evolution. Smart Grid applications need to support **diverse information sources and users**. Besides sensors and instruments monitoring the infrastructure to produce an avalanche of data, information on electrical equipment, organizations, class schedules and weather are also used for decision making [22]. The managers and consumers of this data also vary. At USC, the facilities management services (FMS) deploys and manages sensors and meters in campus buildings for energy monitoring and control. Research and service groups such as the sustainability center, energy club, and energy forecasting models consume this data for analysis. Users like FMS operators, department coordinators and students need an integrated and easy to use view of the complex data to support their individual needs.

Another feature of Smart Grids is their **continuous evolution**, given the emerging nature of technologies and deployment. For e.g., USC, as the largest private power consumer in Los Angeles, has over 60,000 students, faculty and staff spread over 170 buildings [23]. This means that infrastructure is constantly being upgraded and consumers change every year. Of late, an average of two new buildings are built each year on campus, each with hundreds of sensors and equipment. Ambient sensors such as temperature, airflow and CO_2 sensors are deployed at the room-level to monitor conditions. Likewise, around 19,000 new students enroll in USC each year which induces changes in power usage profiles in dormitories and classrooms. Smart Grid applications need *sustainably adapt* to these changes in the information space with low overhead.

2.1 Online Strategies for Dynamic Demand Response

Traditional DR approaches are static: the decision is global and made in advance [1]. Dynamic DR as introduced in [18] supplements traditional DR by offering a more fine-grained approach that is responsive to dynamic power usage changes. The pervasive sensing capability enables us to monitor power consumption and its indirect influencers, such as weather and occupancy, in near real-time. Dynamic DR adopts a data driven approach that detects the occurrence of specific information patterns by examining hundreds or thousands of online data streams. Such analysis offers deeper situational awareness on power usage behavior for timely and opportunistic curtailment strategies.

Our objective is to provide a framework where multi-disciplinary users can *define* DR situations at higher level abstractions, and *detect* these situations over dynamic Smart Grid data streams for timely decision making. Sample dynamic DR situations are listed below, with more details in [18].

Situation 1. The *space temperature* in an *office* of *EE department* is lower than the *green building temperature*.

Situation 2. The *power use* of a *teaching building* exceeds its *pre-peak demand*.

Situation 3. The *space temperature* in a non-occupied *meeting room* is lower than the *green building temperature*.

Situation 4. *Fan coils* in building *MHP* peak concurrently.

Situation 5. The *temperature* in a *meeting room* is above setpoint by 5 °F.

2.2 Semantic Information Model for Smart Grid Applications

Earlier, we have designed an integrated, modular Smart Grid semantic information model for dynamic DR applications [24,22]. In summary, this captures,

Data Sources. We model Smart Grid data sources and the information they emit, including smart meters which measure the power use of buildings, sensors which detect room occupancy, temperature and airflow sensors which measure HVAC status, weather reporting services. These sources are linked to concepts of physical and virtual spaces where they monitor.

Infrastructure. We model both the campus power grid infrastructure, such as the distribution network, and the physical environment. This includes concepts and relationships between building, rooms, and energy sinks like appliances and equipment. These relate to power usage behaviors and curtailment capabilities. For example, a *meeting room* may need to be cooled only when occupied.

Organization. Campus organizations including schools, departments, laboratories and so on are also modeled. These can help users to define organization-specific DR strategies even though it may span physical locations. For example, a monitoring pattern can alert the department coordinator when consumption exceeds a certain threshold.

Other Information. Other information including scheduling and weather also help DR applications, and are modeled using existing domain ontologies.

3 Semantics in Action on Campus Micro Grid

The semantic Smart Grid information model forms the center piece of many applications in the USC campus Micro Grid. These range from asset management, information diffusion and data analysis, allowing different participants to cooperate on campus-wide DR operations. In the following we discuss these applications and their use of semantics.

3.1 Sustainable Asset Management

Asset management is a basic power grid operation. In a Smart Grid, asset management must shift from a dependence on domain experts' experiences to a reusable knowledge base. This is necessary to deal with an aging (but experienced) workforce whose retirement can lead to less experienced staff without

holistic knowledge of the evolving grid. Another concern is the need for seamless and rapid integration of new resources deployed in the power grid.

As a Smart Grid testbed, the USC campus Micro Grid is pioneering novel infrastructure and its management. The USC FMS deploys and manages assets on campus, replacing electromechanical devices by digital instruments, introducing power efficient equipment and enabling bi-directional grid communications. At the time of writing, smart meters were deployed to monitor over 170 buildings on campus, and over 50,000 sensors installed to monitor room temperature, HVAC airflow and fan speed, and even CO_2 levels. As equipments are upgraded, ensuring transfer of knowledge and its accessibility to relevant users is a challenge.

Semantic technologies enable sustainable asset management for the campus Micro Grid. We worked with USC FMS to build the Smart Grid ontology model which captures relevant Micro Grid aspects ranging from electrical equipments, buildings, participants, and departments [24]. Using this model, asset management merely involves performing model queries that are synchronized with field operations. For example, when a new occupancy sensor is installed, the facility operator inserts the sensor entity into the semantic repository, describing it and its relations to existing domain entities using properties such as "ee:hasID" and "ee:hasLocation". A new sensor type can be intuitively introduced by creating a concept such as "ee:OccupancySensor" in the model, defining its properties and classifying it under an existing parent category such as "ee:Sensor". Compared to relational model, semantic ontologies support property inheritance and reasoning while ensuring that introduction of new concepts, or in other word enriching the schema, does not affect legacy data. Further, external models (such as Weather) can be easily integrated. Semantic model based asset management hence ensures rapid and extensible knowledge transference and integration.

3.2 Accessible Information Diffusion

Another key Smart Grid activity is to promote energy awareness and participation by delivering interpretable energy use information to power consumers and end-use applications. We use a web portal as the primary vehicle for this task. Incorporating semantics in these applications considerably improves the process of information dissemination in a heterogeneous power grid environment. It provides ubiquitous data and query representation that hides complexities associated with multi-disciplinary users and distributed asset management.

Our prototype web portal for campus Micro Grid information exploration, *eScope*, is hosted at `smartgrid.usc.edu`. It provides not only static asset information but also dynamic energy use "heatmap" for the campus. The portal supports the needs of both domain experts for easy exploration and end consumers wishing to learn about the energy footprint. It uses SPARQL queries to extract and present this information from a semantic repository; we use *4store*. A sophisticated information integration pipeline running on a private Eucalyptus Cloud continuously retrieves raw data tuples from various information sources including sensors, maps them into RDF triples using rules, and inserts them into the semantic database where it is linked with domain ontologies.

Using semantics allows easy information retrieval by even non-experts, such as portal developers. This is particularly important in a multi-disciplinary project. For example, the USC Micro Grid does not follow a consistent sensor naming scheme. The same type of sensor type, such as kilowatt-hour sensor, has opaque names like "D163Watts", "XLP0100103000022UD" or "BIE_TotalWatts" for different buildings, making it challenging for the portal to maintain static queries for display results as a heatmap. By using semantic concepts rather than a relational database, the portal developer can query for conceptual terms using SPARQL, such as recent "ee:Kilowatt" readings from all "bd:TeachingBuilding":

```
SELECT ?building, ?time, ?kwhReading
WHERE {
   ?event evt:hasSource ?src . ?src ee:hasLocation ?loc .
   ?loc rdf:type bd:TeachingBuilding . ?loc bd:hasCode ?building .
   ?event rdf:type ee:Kilowatt . ?event evt:hasValue ?kwhReading .
   ?event evt:hasTimestamp ?time . FILTER(?time > "10:15:00")}
ORDER BY DESC(?time)
```

This also allows new portal features to be incorporated *rapidly* by just understanding a few concept terms, and *seamlessly* reflects infrastructure upgrades.

3.3 Data Analysis

Data analytics for both off-line demand forecast as well as online DR optimizations utilize the semantic knowledge.

Adaptive Planning Using Machine Learned Forecasting. We use machine learning to train power consumption forecasting models using historical energy use data [3] that are useful for planning equipment upgrades, maintenance schedules, and curtailment policies. These models use different semantic features as indirect influencers of energy use, such as the types of buildings, rooms and customers, academic schedules, weather conditions, and so on. However, not all features may be relevant for prediction at all times. The web portal allows a data analyst to easily explore the semantic knowledge base to identify candidate features of interest. Historical values of these identified features, along with the power consumption of that target buildings, are then extracted from the semantic repository and normalized into a form that can be consumed by a Regression Tree training model. Semantics allow the analyst – who is not an expert on power systems – to still navigate the domain models and pick potential influences of energy use, allowing knowledge to be easily imparted without requiring a domain expert by her side.

Online Optimization Using Semantic CEP. Online data analytics for DR uses complex event processing (CEP) for detecting real-time situations, represented as event patterns, from among streams of events. The limitation of current CEP systems in processing only structural patterns impedes their effective use in an information rich domain like Smart Grid. Existing systems process events

streaming from sensors as plain relational data tuples. As such, complex event patterns can only be defined as a combination of attributes presented in event data. Users have to know the details of event structures and sources before defining low level pattern specifications.

We have introduced semantics into CEP [25] as a solution to meet online DR requirements. Details of *SCEPter*, our semantic CEP system, is outside the scope of this work. However, it does offer several advantages.

Interoperability. The broad space of software and hardware vendors in Smart Grids means that different standards need to co-exist. This extends to data formats and schemas. For example, airflow sensors on campus use different variants of the "airflow" attribute such as "flowrate" and "airvolume" in their event format. With a traditional CEP system, pattern designers are exposed to the structural heterogeneity of events and have to rewrite the same query for different data streams whose formats may vary. A Semantic CEP system helps capture these distinctions, for example using "owl:sameAs" relations, which allows a unified *conceptual* query specification over heterogeneous event formats without in-depth knowledge of standards. This also reduces the complexity operational debugging by having a smaller set of conceptual patterns.

Expressivity. Traditional CEP systems process events solely based on the attributes they posses in the event tuple. By mapping events and their tuple attributes to as part of the semantic ontology, query constraints can then be defined on related domain concepts and entities. This significantly enhances the power of an event pattern specification in detecting very precise situations, while eliminating false positives.

Accessibility. Defining DR event patterns over domain ontologies shield users from lower level details of data streams and their changes. As shown in the examples in § 2, we can easily define patterns that apply to only *meeting rooms* on campus, even if the user has no idea of which buildings have meeting rooms, let alone the sensors that are deployed in those rooms.

In the following sections, we discuss the semantic-enriched event processing approach and optimizations for Smart Grid applications in detail.

4 Semantic Complex Event Processing Model

We provide a semantic stream query language and a *data-driven* processing engine for dynamic DR applications in Smart Grid [25]. Data access systems are typically data-driven or query-driven [5] based on what initiates/completes the operation. CEP systems are data-driven as a pattern is detected when the last event required for a complete match arrives. Query-driven systems such as relational/semantic databases (often) evaluate results as soon as a query is submitted. Conceptually, our system is data-driven as incoming events continuously trigger pattern evaluation. However, since our semantic CEP pattern is specified over both static data in semantic repositories and dynamic stream data, the im-

plementation causes the *event-triggered pattern evaluation* to incorporate both a semantic query part and a subsequent CEP pattern detection.

4.1 Semantic Event Model

The state-of-the-art CEP systems [2,13] process primitive events as relational data tuples, i.e.,

$$primitive\ event ::= < attributes;\ timestamp >$$

Based on the relational model, complex events (event patterns) are defined as compositions of primitive events with attribute constraints. Directly applying CEP systems for dynamic DR requires users define DR event patterns at data level and synchronize patterns with the grid infrastructure upgrades. For example, data schema of space temperature and occupancy measurement streams on USC campus is,

$$event\ tuple = < sensorID,\ reading;\ timestamp >$$

To define Situation 1 as a traditional CEP pattern, users have to explicitly specify the list of thermostat which locate in offices of EE department and keep it up-to-date in the query. In addition, as CEP patterns can only be matched by evaluating syntactic identical attributes, semantic mismatches between user vocabularies and event data have to be addressed manually.

To overcome these limitations, we propose to link dynamic data streams with background ontologies to process semantics of events. Figure 1 shows an example of semantic temperature measurement event. It's essentially a RDF event graph connected to domain ontologies with properties materialized from the original data tuple.

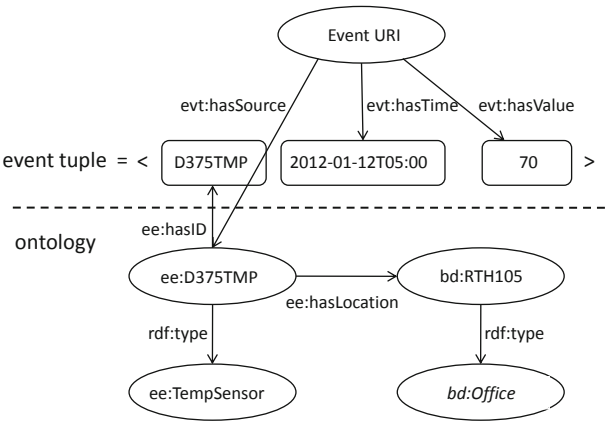


Fig. 1. Semantic Event Linked with Background Knowledge

4.2 Semantic Event Query Model

We propose a two-segment query model over the semantic events described above for dynamic DR situation modeling. The general query structure is,

```
Semantic Event Pattern ::=
      [semantic filtering subpattern]{0, n}
      [syntactic CEP subpattern]+
```

A semantic event pattern in our system consists of two types of subpatterns: semantic filtering subpatterns and syntactic CEP subpatterns. The semantic filtering subpatterns are SPARQL queries to specify semantic constraints and annotations of input events. A pattern can have 0 to n semantic subpatterns where n is the number of streams the pattern correlate. The syntactic CEP subpatterns are traditional CEP queries which specify temporal and logic constraints over filtered and enriched event tuples. One pattern can have 0 or 1 CEP subpattern. For example, Situation 1 can be modeled as a pattern with 1 semantic filtering subpattern and 0 CEP subpattern. The semantic filtering subpattern in SPARQL is,

```
SELECT ?e
WHERE {
   ?e evt:hasSource ?src .
   ?src ee:hasLocation ?loc .
   ?loc rdf:type bd:Office .
   ?loc bd:belongs org:EE_Department .
   ?src rdf:type ee:TempSensor .
   ?e evt:hasValue ?reading .
   bd:GreenBuildingTemp bd:hasValue ?val .
   FILTER(?reading < ?val) }
```

Consider another example, the pattern for Situation 3 correlates 2 streams including the occupancy and temperature measurement streams. It has a semantic filtering subpattern on each stream to query/select event location and constrain the location type as "bd:MeetingRoom". The semantic subpattern for the temperature measurement stream also specifies the temperature reading is less than "bd:GreenBuildingTemp". Denote "?o" as filtered events from the occupancy stream and "?t" as filtered events from the temperature stream and assume the filtered event tuples are annotated with a new location attribute "loc", the CEP subpattern for Situation 3 represented in Siddhi [19] is,

```
AND (?t, ?o)
CONDITION {(?t.loc = ?o.loc) and (?o.reading = false)}
```

We developed the semantic event query processing system around an existing CEP engine kernel, Siddhi. When a new event tuple arrives on input streams,

the corresponding semantic events are materialized and combined with domain ontologies to evaluate semantic subpatterns. Qualified data tuples are extended and passed to the CEP engine for syntactic subpattern matching. For details of the processing engine please refer to [25].

5 Cache Optimization for Continuous Querying

In this section, we discuss optimizations for processing semantic filtering subpatterns over event streams. The baseline approach is to perform semantic queries whenever a new event arrives. However, semantic querying is known as time expensive. It performs inference and self-join operations over the ontology knowledge base. In general, a semantic query with a single path expression requires (n-1) self-joins over the ontology where n is length of the path.

We developed caching algorithms to improve the performance of semantic stream querying. As an initial effort, we make the following assumptions,

- The semantic filtering queries do not correlate multiple events.
- The semantic filtering queries do not contain alternative or disjunctive triple patterns.

5.1 Query Caching

The key observation for query caching is multiple events may share semantic query results so that the system need not evaluate queries for all events. A semantic event is modeled as a directed tree whose root node is the event URI and has edges linked to event properties as shown in Figure 1. We introduce the following definition,

Definition 1. The *event root properties* of a semantic event are the properties directly materialized from its data tuple attributes.

As examples, for the semantic event shown in Figure 1, "ee:D375TMP", "2012-01-12T05:00" and "70" are *event root properties*.

On the other hand, a semantic event query can also be modeled as a directed tree whose root node is an event variable and it has edges connected to property variables, literals, ontology classes or instances. Executing a semantic event query is essentially finding event trees that match the query tree. As an example, Figure 2 shows the query tree for the semantic filtering subpattern of Situation 1.

As shown in the query tree, the inner nodes are all variables and leaf nodes are either literals, ontology classes or instances. The edges can be classified as relation or evaluation edges. We further define,

Definition 2. The *query root properties* of a semantic event for a query are the event root properties which are evaluated in the query.

For example, for the semantic event shown in Figure 1, "ee:D375TMP" and "70" are *query root properties* for the query tree shown in Figure 2. For any query which satisfies the assumptions stated before, its tree graph can be decomposed as conjunctive paths from the event variable node, through the variable nodes

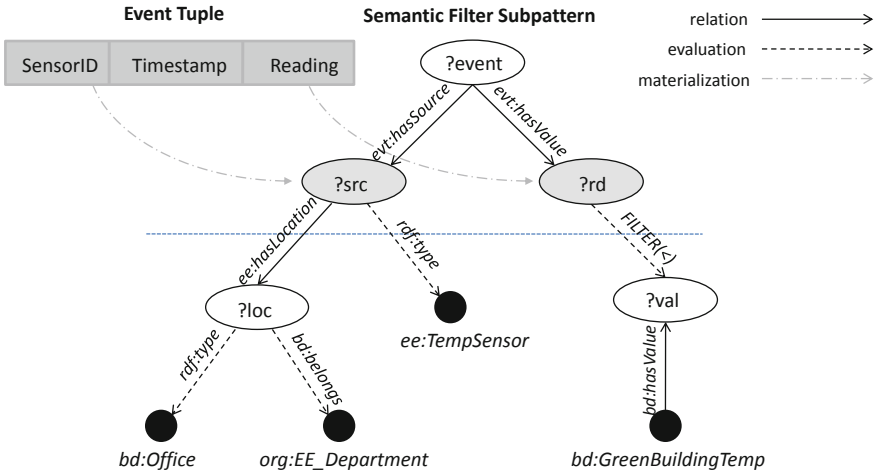


Fig. 2. Semantic Event Query Graph

of *query root properties* to leaf nodes. Whether a semantic query is evaluated to true or false for an event is hence completely determined by the *query root properties* of the event for that query. In other word, if two events share the same *query root properties* for a query, they can share the query result.

Based on the above observation, we design the query caching mechanism as,

Cache Data Structure. The cache is implemented as a collection of hash tables. The SCEP system initializes and maintains one hash table for each query. The *query root properties* are used as the cache key and the boolean query evaluation result is the cache value.

Cache Lookup/Update When a new event e arrives, the system fetches the *query root properties* of e for a query Q and look up the corresponding hash table for matches. If it hit the cache, we use the cached result without actually performing the query. Otherwise the system materializes the semantic event for querying and updates the cache. Currently, we implemented a simple Least-Frequently-Used (LFU) update strategy.

Denote the boolean function that evaluates query Q over event e and domain ontologies O as $Evaluate(Q, e, O)$, the pseudo code for semantic event query with query caching is shown in Algorithm 1.

5.2 Path Caching

In addition to sharing query results between events, it also makes sense to reuse path evaluations between queries especially when a number of queries share a smaller set of path expressions. Consider Situation 1, 3 and 5, Figure 3 shows the path sharing between their semantic filtering subpatterns Q1, Q3 and Q5. It also should be noticed that path expressions with leaf nodes such as "bd:Office" and

Algorithm 1 . Semantic Stream Query with Query Caching

Require: Cache table ht initialized for Q, domain ontologies O
Ensure: Evaluation result v for query Q
1: **while** Receiving semantic event e **do**
2: Compute cache key k for e
3: $v \leftarrow$ ht.get(k)
4: **if** $v! = null$ **then**
5: Return v
6: **else**
7: $v \leftarrow$ Evaluate(Q, e, O)
8: Update $\{k,\ v\}$ to ht based on LFU policy
9: Return v
10: **end if**
11: **end while**

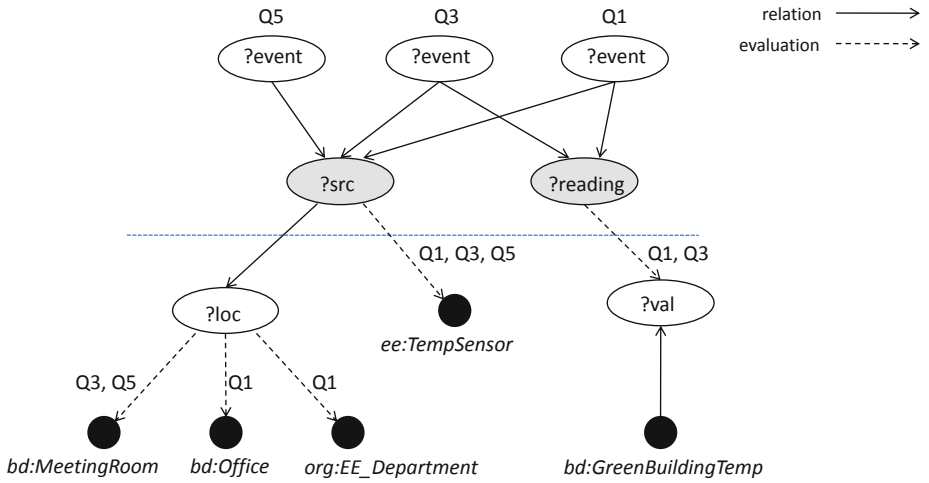


Fig. 3. Sharing Paths between Semantic Event Queries

"bd:MeetingRoom", which are semantically disjoint, only need to be evaluated once for events with the same query root properties. Based on above observations, in the second cache optimization approach we maintain cache tables for individual query path and update caches by referencing the semantic relations between paths. The pseudo code for semantic stream query with path caching is shown in Algorithm 2.

5.3 Evaluations

Experiments are conducted to evaluate the semantic caching algorithms in dynamic DR scenarios expected for the USC Campus Micro Grid. In these experiments, we run the SCEP system on a 12-core AMD Opteron server, with 2.8GHz cores, 32GB physical memory and running Windows Server 2008.

Algorithm 2 . Semantic Stream Query with Path Caching

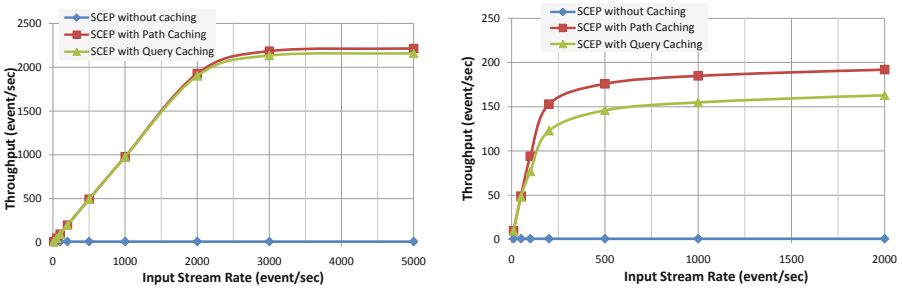
Require: Cache table ht_i initialized for path P_i ($i = 1$ *to* n) of query Q, dht_{ij} ($j = 1$ *to* m) identified as disjoint cache tables of P_i, domain ontologies O
Ensure: Evaluation result v for query Q

```
while Receiving semantic event e do
  Compute cache keys k_i of e for P_i (i = 1 to n)
  v ← true
  for i = 1 to n do
    v_i ← ht_i.get(k_i)
    if v_i = true then
      Continue
    else if v_i = false then
      v ← false
      Return v
    else
      v_i ← Evaluate(P_i, e, O)
      Update {k_i, v_i} to ht_i based on LFU policy
      Update {k_i, !v_i} to dht_ij (j = 1 to m) based on LFU policy
    end if
  end for
  Return v
end while
```

Data collected from HVAC systems and smart meters on the USC campus is used as experimental data streams. We performed two sets of experiments each for three times and the average values are reported here. In the first set of experiments, we submit 9 semantic CEP queries to the SCEP engine and compare the throughput of the system in the case without caching optimization, with query caching and with path caching. In the second set of experiments we submit 120 queries to the engine and evaluate the time performances of the three algorithms again. Figure 4 shows the experiment results. Obviously, the number of queries has significant impact on the system performance. We expect to sample campus sensor data at 1-minute interval for the dynamic DR applications. This requires a minimum throughput of 83 events/second to handle around 5000 data points on campus. Without caching optimization, the system can merely process around 10 events per second with 9 queries and 0.7 event with 120 queries. The throughput of the system ranges from 130 to 2400 events/second in the two experiments with query caching and path caching. We also notice when the number of pattern increases, path caching usually outperforms query caching as it allows evaluation results to be shared between queries.

6 Related Work

Semantic Smart Grid Information Modeling. The power systems industry has been opaque, dominated by a few large companies with proprietary information stacks. Smart Grids are forcing this to change. Standards designed by



(a) SCEP Throughput with 9 Concurrent Queries

(b) SCEP Throughput with 120 Concurrent Queries

Fig. 4. Evaluation of Semantic Caching for Streaming SCEP Queries

organizations like IEC and NIST provide common protocols and data models that can be used by the various participants. There has also been recent work on developing semantic-level Smart Grid information integration framework. [4] proposed a shared ontology model to provide common semantics for Smart Grid applications. The ontology captures domain concepts by transforming existing standards, such as IEC's Common Information Model (CIM), to a uniform conceptual model. Our semantic Smart Grid information model can be considered as an extension to the model proposed in [4], and complements it with broader knowledge that is required for DR decision making. Besides power grid domain elements, we also link these with modular ontologies on physical spaces, organization, and weather that are crucial to DR applications.

Complex Event Processing. Traditional CEP approaches like Cayuga [2] and SASE [13,9] have focused on specifying and detecting temporal and logical relations among syntactical events modeled as an infinite sequences of relational tuples with interval-based timestamps. These use a SQL like query model with operators such as selection, projection, and conditional sequence.

The problem of semantic stream processing has been discussed in C-SPARQL [7] and ETALIS [5,6]. C-SPARQL extends the SPARQL language with window and aggregation clauses to support RDF stream processing. However, while C-SPARQL extensively considers aggregation operations, it does not support several stream processing operators that are essential to Smart Grids, including temporal sequence and negation. ETALIS is a rule-based deductive system that acts as a unified execution engine for temporal pattern matching and semantic reasoning. It implements two languages for specification of event patterns: ETALIS Language for Events (ELE), and EP-SPARQL for stream reasoning. Both event patterns and semantic background knowledge are transformed to Prolog rules and executed by a Prolog inference engine for reasoning and pattern detection. However, these languages independently are not insufficient for our use cases. The ELE pattern language lacks semantic operators while EP-SPARQL supports few temporal operators such as sequence and optional sequence. Rather than adopt a bespoke solution that departs from traditional CEP systems, our proposed semantic CEP

framework is a hybrid that leverages the native features of both CEP (Siddhi) and SPARQL engines to offer a richer query syntax. More practically, it also allowed for rapid construction of such a framework for our Micro Grid using existing tooling, and improves the performance using the proposed optimizations.

Semantic Caching. Semantic caching has been widely studied for database query optimization [14] by storing the results of previously queries locally. [12] discusses caching theory in terms of deciding when answers are in cache, and semantic overlap. [17] describes the use of semantic cache in an ontology-based web mediator system and considers extracting partial results from caches for new queries. A special feature of their approach is organizing cache by concepts and exploiting domain knowledge for defining queries to complement partial cache results. While not breaking new ground in caching strategies, we do apply it to a novel scenario of Semantic CEP where query performance over continuous event data can be punitive otherwise. Our caching algorithms for stream queries leverage existing state-of-the-art, resembling the global caching in [8,12], and partial query caching [15]. Other than caching query results for subsequent relational queries, we cache semantic query results for new data.

7 Conclusion and Future Work

We have discussed incorporating semantics into Smart Grid applications and dynamic stream processing for the USC Micro Grid. Semantic Web domain ontologies form the foundation for diverse participants and DR applications to manage and access data conceptually. Realtime grid observations abstracted as semantic events allow intuitive definition and detection of semantic CEP patterns. Out caching optimizations improve its performance, as validated empirically.

Our work lies in two directions. First, we plan to extend the ontology models from campus Micro Grid to a utility scale, and identify additional semantic event patterns for DR strategies. Second, we will investigate additional optimizations to overcome performance bottlenecks of semantic event processing, which currently limit throughput to less than 3000 events/second even with cache optimization.

Acknowledgments. This work is supported by the Department of Energy under Award Number DE-OE0000192 and the Los Angeles Department of Water and Power. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof, the LA DWP, nor any of their employees.

References

1. FERC assessment of demand response and advanced metering. Staff Report (December 2008)
2. Demers, A., Gehrke, J., et al.: Cayuga: A general purpose event monitoring system. In: The Conference on Innovative Data Systems Research, CIDR (2007)

3. Saima, A., Simmhan, Y., Prasanna, V.: Improving energy use forecast for campus micro-grids using indirect indicators. In: International Workshop on Domain Driven Data Mining (2011)
4. Andrew Crapo, J.L., Wang, X., Larson, R.: The semantically enabled smart grid. Technical report
5. Anicic, D., Fodor, P., Stuhmer, R., Stojanovic, N.: Event-driven approach for logic-based complex event processing. In: International Conference on Computational Science and Engineering (2009)
6. Anicic, D., Rudolph, S., Fodor, P., Stojanovic, N.: Stream reasoning and complex event processing in etalis. Semantic Web Journal (2012)
7. Francesco, B., Daniele, B., et al.: An execution environment for c-sparql queries. In: International Conference on Extending Database Technology (EDBT) (2010)
8. Dar, S., Franklin, M., Johnsson, B., Srivastava, D., Tan, M.: Semantic Data Cache and Replacement. In: Very Large Data Base Conference, VLDB (1996)
9. Diao, Y., Immerman, N., Gyllstrom, D.: SASE+: An agile language for Kleene closure over event streams. Technical report, UMass (2007)
10. Dung, T.Q., Kameyama, W.: A proposal of ontology-based health care information extraction system: Vnhies. In: IEEE International Conference on Research, Innovation and Vision for the Future (2007)
11. Valle, E.D., Celino, I., Dell'Aglio, D.: The experience of realizing a semantic web urban computing application. In: The Terra Cognita Workshop (2009)
12. Godfrey, P., Gryz, J.: Answering Queries by Semantic Caches. In: Bench-Capon, T.J.M., Soda, G., Tjoa, A.M. (eds.) DEXA 1999. LNCS, vol. 1677, pp. 485–498. Springer, Heidelberg (1999)
13. Gyllstrom, D., Wu, E., et al.: SASE: Complex event processing over streams. In: The 3rd Biennial Conference on Innovative Data Systems Research (2007)
14. Halevy, A.Y.: Answering queries using views: A survey. The VLDB Journal (2001)
15. Keller, A.M., Basu, J.: A predicate-based caching scheme for client-server database architectures. The VLDB Journal, 5 (1996)
16. Lord, P., Bechhofer, S., Wilkinson, M.D., Schiltz, G., Gessler, D., Hull, D., Goble, C.A., Stein, L.: Applying Semantic Web Services to Bioinformatics: Experiences Gained, Lessons Learnt. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 350–364. Springer, Heidelberg (2004)
17. Karnstedt, M., Sattler, K., Geist, I., et al.: Semantic caching in ontology-based mediator systems. In: Berliner XML Tage (2003)
18. Zhou, Q., Simmhan, Y., Prasanna, V.: On using semantic complex event processing for dynamic demand response optimization. Technical report, Computer Science Department, University of Southern California (2012)
19. Suhothayan, S., Gajasinghe, K., Loku Narangoda, I., Chaturanga, S., Perera, S., Nanayakkara, V.: Siddhi: A second look at complex event processing architectures. In: ACM GCE Workshop (2011), http://siddhi.sourceforge.net
20. Tao, C., Solbrig, H.R., Sharma, D.K., Wei, W.-Q., Savova, G.K., Chute, C.G.: Time-Oriented Question Answering from Clinical Narratives Using Semantic-Web Techniques. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part II. LNCS, vol. 6497, pp. 241–256. Springer, Heidelberg (2010)

21. Taswell, C.: Doors to the semantic web and grid with a portal for biomedical computing. IEEE Transactions on Information Technology in Biomedicine (2008)
22. Simmhan, Y., Zhou, Q., Prasanna, V.: Chapter: Semantic Information Integration for Smart Grid Applications (2011)
23. Simmhan, Y., Aman, S., et al.: An informatics approach to demand response optimization in Smart Grids. Technical report, USC (2011)
24. Zhou, Q., Natarajan, S., Simmhan, Y., Prasanna, V.: Semantic information modeling for emerging applications in smart grid. In: IEEE Conference on Information Technology: New Generations (2012)
25. Zhou, Q., Simmhan, Y., Prasanna, V.: SCEPter: Semantic complex event processing over end-to-end data flows. Technical Report 12-926, Computer Science Department, University of Southern California (2012)

Evaluating Semantic Search Query Approaches with Expert and Casual Users*

Khadija Elbedweihy, Stuart N. Wrigley, and Fabio Ciravegna

Department of Computer Science, University of Sheffield, UK
{k.elbedweihy,s.wrigley,f.ciravegna}@dcs.shef.ac.uk

Abstract. Usability and user satisfaction are of paramount importance when designing interactive software solutions. Furthermore, the optimal design can be dependent not only on the task but also on the type of user. Evaluations can shed light on these issues; however, very few studies have focused on assessing the usability of semantic search systems. As semantic search becomes mainstream, there is growing need for standardised, comprehensive evaluation frameworks. In this study, we assess the usability and user satisfaction of different semantic search query input approaches (natural language and view-based) from the perspective of different user types (experts and casuals). Contrary to previous studies, we found that casual users preferred the form-based query approach whereas expert users found the graph-based to be the most intuitive. Additionally, the controlled-language model offered the most support for casual users but was perceived as restrictive by experts, thus limiting their ability to express their information needs.

1 Introduction

Semantic Web search engines (e.g. Sindice [1]) offer gateways to locate Semantic Web documents and ontologies; ontology-based natural language interfaces (e.g. NLP-Reduce [2]) and visual query approaches (e.g. Semantic Crystal [2]) allow more user-friendly querying; while others try to provide the same support but on the open Web of Data [3,4]. These search approaches require and employ different query languages. Free-NL provides high expressiveness by allowing users to input queries using their own terms (keywords or full sentences). Controlled-NL provides support during query formulation through suggestions of valid query terms found in the underlying – restrictive – vocabulary.

Finally, view-based (graphs and forms) approaches aim to provide the most support to users by visualising the search space in order to help them understand the available data and the possible queries that can be formulated.

Evaluation of software systems – including user interfaces – has been acknowledged in literature as a critical necessity [5,6]. Indeed, large-scale evaluations foster research and development by identifying gaps in current approaches and suggesting areas for improvements and future work. Following the Cranfield

* This work was partially supported by the European Union 7th FWP ICT based e-Infrastructures Project SEALS (Semantic Evaluation at Large Scale, FP7-238975).

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 274–286, 2012.

model [7] – using a test collection, a set of tasks and relevance judgments – and using standard evaluation measures such as precision and recall has been the dominant approach in IR evaluations, led by TREC [8]. This approach has not been without criticisms [9,10] and there have been long-standing calls for assessing the interactive aspect as well [11,12].

In an attempt to address these issues, more studies have been conducted with a focus on Interactive Information Retrieval (IIR). The ones embodied within TREC (*Interactive Track* [13] and *Complex Interactive Question-Answering* [14]) involved real users to create topics or evaluate documents rather than to assess usability and usefulness of the IR systems. Others investigated users perception of ease-of-use and user control with respect to the effectiveness of the retrieval process [15] or studied the impact and use of cross-language retrieval systems [16]. With respect to the type of users involved in these studies, some [17,18] have opted to further differentiate between *casual users* and *expert users*. In the context of these works and indeed in ours, *casual users* refer to those with very little or no knowledge in a specific field (e.g., Semantic Web, for our study), while *expert users* have more knowledge and experience in that field.

Inheriting IR's evaluation paradigm, Semantic Search evaluation efforts have been largely performance-oriented [6,19] with a limited attention to the user-related aspects [20,21]. Kaufmann and Bernstein [20] conducted a within-subjects (same group of subjects evaluate all the participating tools) evaluation of four tools adopting NL- and graph-based approaches with 48 casual users while the evaluation described in [21] featured NL- and form-based tools.

The evaluation described here is different in the following ways: 1) broader range of query approaches (in contrast to [20,21]), 2) all tools are evaluated within-subjects (in contrast to [21]), and 3) equal-sized subjects groups for casual and expert users (in contrast to [20,21]). These differences are important and allow novel analyses to be conducted since it facilitates direct comparison of the evaluated approaches and a first-time understanding and comparison of how the two types of users perceive the usability of these approaches. Although some IIR studies involved casual and expert users, most of these focused on investigating differences in the search behaviour and strategies [17,18,22].

The remainder of the paper is organized as follows: first, the usability study is described. Next, the results and analyses are discussed together with the main conclusions and finally, the limitations are pointed out with planned future work.

2 Usability Study

The underlying question of the research presented in this paper is how users perceive the usability of different semantic search approaches (specifically support in query formulation and suitability of results returned), and whether this perception is different between expert and casual users. To answer the question, ten casual users and ten expert users were asked to perform five search tasks with five tools adopting NL-based and view-based query approaches. These are user-centric semantic search tools (e.g. query given as natural language or using

a form or a graph) querying a repository of semantic data and returning answers extracted from them. The results returned must be answers rather than documents; however they are not limited to a specific style (e.g., list of entity URIs or visualised results). Experiment results such as query input time, success rates and input of questionnaires are recorded. These results are quantitatively and qualitatively analysed to assess tools' usability and user satisfaction.

2.1 Dataset and Questions

The main requirement for the dataset is to be from a simple and understandable domain for users to be able to formulate the given questions into the tools' query languages. Hence, the geography dataset within the Mooney Natural Language Learning Data[1] was selected. It contained predefined English language questions and has been used by other related studies [20,23]. The five evaluation questions (given below) were chosen to range from simple to complex ones and to test tools' ability in supporting specific features such as comparison or negation.

1. *Give me all the capitals of the USA?*
   This is the simplest question: consisting of only one ontology concept: '*capital*' and one relation between this concept and the given instance: *USA*.
2. *What are the cities in states through which the Mississippi runs?*
   This question contains two concepts: '*city*' and '*state*' and two relations: one between the two concepts and one linking *state* with *Mississippi*.
3. *Which states have a city named Columbia with a city population over 50,000?*
   This question features comparison for a datatype property *city population* and a specific value (50,000).
4. *Which lakes are in the state with the highest point?*
   This question tests the ability for supporting superlatives (*highest point*).
5. *Tell me which rivers do not traverse the state with the capital Nashville?*
   Negation is a traditionally challenging feature for semantic search [24,25].

2.2 Experiment Setup

Twenty subjects were recruited for the evaluation; ten of these subjects were *casual users* and ten were *expert users*. The 20 subjects (12 females, 8 males) were aged between 19–46 with a mean of 30 years. The experiment followed a within-subjects design to allow direct comparison between the evaluated query approaches. Additionally, with this design, usually less participants are required to get statistically significant results [26]. All 20 subjects evaluated the five tools in randomised order to avoid any learning, tiredness or frustration effects that could influence the experiment results. Furthermore, to avoid any possible bias introduced by developers evaluating their own tools, only one test leader – who is also not the developer of any of the tools – was responsible for running the whole experiment.

[1] http://www.cs.utexas.edu/users/ml/nldata.html

For each tool, subjects were given a short demo session explaining how to use it to formulate queries. After that, subjects were asked to formulate each of the five questions in turn using the tool's interface. The order of the questions was randomised for each tool to avoid any learning effects. After testing each tool, subjects were asked to fill in two questionnaires.

Finally, we collected demographics data such as age, profession and knowledge of linguistics (see [27] for details of all three questionnaires). Each experiment with one user took between 60 to 90 minutes.

In assessing usability of user-interfaces, several measurements including time required to perform tasks, success rate and perceived user satisfaction were proposed in the literature of IIR [28,29] and HCI [30,31].

Similar to these studies and indeed to allow for deeper analysis, we collected both objective and subjective data covering the experiment results. The first included: 1) *input time* required by users to formulate their queries, 2) *number of attempts* showing how many times on average users reformulated their query to obtain answers with which they were satisfied (or indicated that they were confident a suitable answer could not be found), and 3) *answer found rate* capturing the distinction between finding the appropriate answer and the user 'giving up' after a number of attempts. This data was collected using custom-written software which allowed each experiment run to be orchestrated.

Additionally, subjective data was collected using think-aloud strategy [32] and two post-search questionnaires. The first is the *System Usability Scale (SUS) questionnaire* [33], a standardised usability test consisting of ten normalised questions covering aspects such as the need for support, training, and complexity and has proven to be very useful when investigating interface usability [34]. The second questionnaire (*Extended Questionnaire*) is one which we designed to capture further aspects such as the user's satisfaction with respect to the tool's query language and the content returned in the results as well as how it was presented. After completing the experiment, subjects were asked to rank the tools according to four different criteria (each one separately): how much they liked the tools (*Tool Rank*); how much they liked their query interfaces: graph-based, form-based, free-NL and controlled-NL (*Query Interface Rank*); how much they found the results to be informative and sufficient (*Results Content Rank*); and finally how much they liked the results presentation (*Results Presentation Rank*). Note that users were allowed to give equal rankings for multiple tools if they had no preference for one over the other. To facilitate comparison, for each criterion, ranking given by all users for one tool was summed and subsequent score was then normalised to have ranges between 0 and 1 (where 1 is the highest).

3 Results and Discussion

Evaluated tools included free-NL- (NLP-Reduce [2]), controlled-NL- (Ginseng [2]), form- (K-Search [35]), and finally graph- based (Semantic-Crystal [2] and Affective Graphs[2]) approaches. Results for both expert and casual users are presented

[2] http://oak.dcs.shef.ac.uk/?q=node/253

in Tables 1 and 2 respectively. In these tables, a number of different factors are reported such as the SUS scores and the tools' rankings. We also include the scores from two of the most relevant questions from the extended questionnaire. *EQ1: liked presentation* shows the average response to the question "I liked the presentation of the answers", while *EQ2: query language easy* shows it for the question "The system's query language was easy to use and understand".

Note that in the rest of this section, we use the term *tool* (e.g. graph-based tools) to refer to the implemented tool as a full semantic search system (with respect to its query interface and approach, functionalities, results presentation, etc.) and the term *query approach* (e.g. graph-based query approach) to specifically refer to the style of query input adopted.

To quantitatively analyse the results, SPSS[3] was used to produce averages, perform correlation analysis and check the statistical significance. The median (as opposed to the mean) was used throughout the analysis since it was found to be less susceptible to outliers or extreme values sometimes found in the data. In the qualitative analysis, the open coding technique [36] was used in which the data was categorised and labelled according to several aspects dominated by usability of the tools' query approaches and returned answers.

3.1 Expert User Results

According to the adjective ratings introduced by [37], Ginseng – with the lowest SUS score – is classified as *Poor*, NLP-Reduce as *Poor* to *OK*, K-Search and Semantic Crystal are both classified as *OK*, while Affective Graphs, which managed to get the highest average SUS score, is classified as *Good*. These results are also confirmed by the tools' ranks (see Table 1): Affective Graphs was selected 60% of the times as the most-liked tool and thus got the highest rank (0.875), followed by Semantic Crystal and K-Search (0.625 and 0.6 respectively) and finally Ginseng and NLP-Reduce got a very low rank (0.225) with each being chosen as the least-liked tools four times and twice, respectively. Since the rankings are an inherently relative measure, they allow for direct tool-to-tool comparisons to be made. Such comparisons using the SUS questionnaire may be less reliable since the questionnaire is completed after each tool's experiment (and thus temporally spaced) with no direct frame of reference to any of the other tools.

Table 1 also shows that Affective Graphs, which is most liked and found to be the most intuitive by users managed to get satisfactory answers for 80% of the queries, followed by K-Search (50%) which is employing the second most-liked query approach. Finally, it was found that all the participating tools did not support negation (except partially by Affective Graphs). This was confirmed by the *answer found rate* for the question "*Tell me which rivers do* ***not*** *traverse the state with the capital nashville?*" being: Affective Graphs: 0.4, Semantic Crystal: 0.1, K-Search: 0.1, Ginseng: 0.1, NLP-Reduce: 0.0.

Expert Users Prefer Graph- and Form- Based Approaches: Results showed that graph- and form- based approaches were the most liked by expert

[3] www.ibm.com/software/uk/analytics/spss/

Table 1. Tools results for expert users. Non-ranked scores are median values; bold values indicate best performing tool in that category.

Criterion	Affective Graphs	Semantic Crystal	K-Search	Ginseng	NLP-Reduce	p-value
SUS (0-100)	**63.75**	50	40	32.5	37.5	0.003
Tool Rank (0-1)	**0.875**	0.625	0.6	0.225	0.225	-
Query Language Rank (0-1)	**0.925**	0.725	0.65	0.425	0.45	-
Results Content Rank (0-1)	0.875	0.875	**0.925**	0.725	0.725	-
Results Presentation Rank (0-1)	0.875	0.875	**0.975**	0.8	0.8	-
EQ1: liked presentation (0-5)	2.5	2.5	**4**	3	3	0.007
EQ2: query language easy (0-5)	**4**	**4**	**4**	2	2.5	0.035
Number of Attempts	**1.5**	2.2	2	1.7	4.1	0.001
Answer Found Rate (0-1)	**0.8**	0.4	0.5	0.4	0.2	0.004
Input Time (s)	88.86	79.55	53.54	102.52	**19.90**	0.001

users. However, in terms of overall satisfaction (see SUS scores and Tool Rank in Table 1), graph-based tools outperformed the form- and NL- based ones. Additionally, feedback showed that *users were able to formulate more complex queries with the view-based approaches (graphs and forms) than with the NL ones (free and controlled)*. Indeed, the ability to visualise the search space provides an understanding of the available data (concepts) as well as connections found between them (relations) which shows how they can be used together in a query [20,38].

It is interesting to note that although Affective Graphs and Semantic Crystal both employ graph-based query approach, users had different perceptions of their usability. More users gave the query interface of Affective Graphs higher scores than Semantic Crystal (quartiles: "3.75 , 5" and "2 , 4.25" respectively) since they found it to be more intuitive. The most repeated (60%) *positive* comment given for Affective Graphs was "*the query interface is intuitive and easy/pleasant to use*". This is a surprising outcome since graph-based approaches are known to be complicated and laborious [20,38]. However, this has not been explicitly assessed from expert users perspective in any similar studies.

An important difference was observed between the two graph-based tools: Semantic Crystal visualizing the entire ontology whereas Affective Graphs opted for showing concepts and relations only selected by the users (see Fig. 1). Although feedback showed that users preferred the first approach, it imposes a limitation on how much can be displayed in the visualisation window. With a small ontology, the graph is clear and can be easily explored; as the ontology gets bigger, the view would easily get cluttered with concepts and links showing relations between them. This would negatively affect the usability of the interface and in turn the user experience.

Expert Users Frustrated by Controlled-NL: Although the guidance provided by the controlled-NL approach was at sometimes appreciated, restricting expert users to the tool's vocabulary was more annoying. This resulted in an unsatisfying experience (lowest SUS score of 32.5 and least liked interface) which is supported by the most repeated *negative* comments given for Ginseng:

- It is frustrating when you cannot construct queries in the way you want.
- You need to know in advance the vocabulary to be able to use the system.

Table 2. Tools results for casual users. Non-ranked scores are median values; bold values indicate best performing tool in that category.

Criterion	Affective Graphs	Semantic Crystal	K-Search	Ginseng	NLP-Reduce	p-value
SUS (0-100)	55	**61.25**	41.25	53.75	43.75	0.485
Tool Rank (0-1)	**0.675**	**0.675**	0.575	0.45	0.275	-
Query Language Rank (0-1)	0.525	0.55	**0.625**	0.525	0.4	-
Results Content Rank (0-1)	0.675	0.75	**0.775**	0.575	0.575	-
Results Presentation Rank (0-1)	0.775	0.7	**0.8**	0.6	0.475	-
EQ1: liked presentation (0-5)	3	3	**3.5**	2.5	2	0.3
EQ2: query language easy (0-5)	**4**	**4**	**4**	3	3	0.131
Number of Attempts	**1.7**	1.8	2.1	**1.7**	4.2	0.001
Answer Found Rate (0-1)	0.4	**0.6**	0.5	0.4	0.2	0.150
Input Time (s)	72.8	75.76	63.59	93.13	**18.6**	0.001

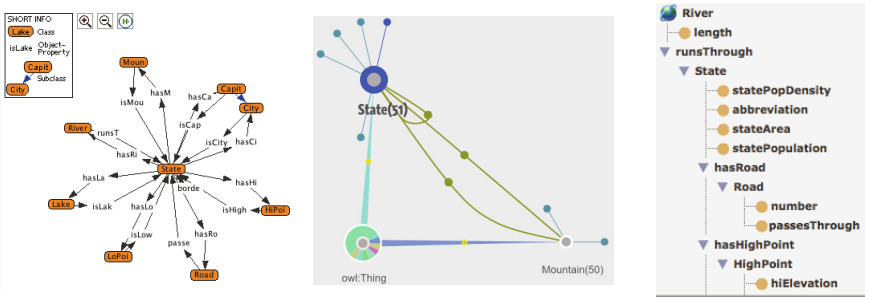
The second comment is in stark contrast to what the controlled-NL approach is designed to provide. It is intended to help users formulate their queries without having to know the underlying vocabulary. However, even with the guidance, users frequently got stuck because they did not know how to associate the suggested concepts, relations or instances together. This is confirmed by users requiring the longest input time when using Ginseng (Table 1 : Input Time).

3.2 Casual User Results

Graph-Based Tools More Complex If Entire Ontology Not Shown: Recall in Section 3.1, expert users preferred the approach of visualising the entire ontology (adopted by Semantic Crystal as shown in Fig. 1a). This was indeed more appreciated by casual users, resulting in Semantic Crystal receiving higher scores. Surprisingly, the lack of this feature caused Affective Graphs to be perceived by casual users as the most complex and difficult to use: 50% of the users found it to be: "*less intuitive and has higher learning curve than NL*".

Tool Interface Aesthetics Important to Casual Users: Most of the casual users (70%) liked the interface of Affective Graphs for having an *animated, modern and visually-appealing* design. This not only created a pleasant search experience but was also helpful during query formulation (e.g., highlighting selected concepts) and in turn balanced the negative effect of not showing the entire ontology, resulting in high user satisfaction (second highest SUS score: 55).

Casual Users Prefer Form-Based Approach: Casual users needed less input time with the form-based approach and found it less complicated than the graph-based approach while allowing more complex queries than the NL-based ones. However, unexpectedly, more attempts were required to formulate their queries using this approach. The presence of inverse relations in the ontology was viewed by casual users as unnecessary redundancy. This impression led to confusion and thus required more trials to formulate the right queries. For instance, to query for the rivers running through a certain state, two alternatives ("State, hasRiver, River" and "River, runsthrough, State") were adopted by



(a) Semantic Crystal (b) Affective Graphs (c) K-Search

Fig. 1. Different visualizations of the Mooney ontology by the tools

users. Tools ought to take the burden off users and provide one unique way to formulate a single query.

Casual Users Liked Controlled-NL Support: Casual users found the guidance offered by suggesting valid query terms very helpful and provided them with more confidence in their queries. Interestingly, they preferred to be 'controlled' by the language model (allowing only valid queries) rather than having more expressiveness (provided by free-NL) while creating more invalid queries.

3.3 Results Independent of User Type

This section discusses results and findings common to both types of users.

Form-Based Faster But More Tedious Than Graph-Based: Results showed that both types of users took less time to formulate their queries with the form-based approach than with the graph-based ones (approximate difference: 36% for experts, 14% for casuals). However, it was found to be more laborious to use than graphs especially when users had to inspect the concepts and properties (presented in a tree-like structure) to select the required ones for the query (see Fig. 1c). This is a challenge acknowledged in the literature [39] for form-based approaches and is supported by the feedback given by users: the most repeated negative comment was "*It was hard to find what I was looking for once a number of items in the tree are expanded*". Additionally, this outcome suggests that input time cannot be used as the sole metric to inform usability of query approaches.

Free-NL Simplest and Most Natural; Suffer from Habitability Problem: The free-NL approach was appreciated by users for being the most simple and natural to them. However, the results showed a frequent mismatch between users' query terms and the ones expected by the tool. This is caused by the abstraction of the search space and is known in literature as the *habitability problem* [2, p.2]. This is supported by the users' most repeated negative comment: "*I have to guess the right words*". They found that they could get answers

with specific query terms rather than others. For instance, using 'run through' with 'river' returns answers which are not given when using 'traverse'. This is also confirmed by the tool (NLP-Reduce) getting the lowest success rate (20%). Furthermore, requiring the highest *number of attempts* (4.1) support users' feedback that they had to rephrase their queries to find the combination of words the tool is expecting. Indeed, this is a general challenge facing natural language interfaces [20,40,41].

Results Content and Presentation Affected Usability and Satisfaction: When evaluating semantic search tools, it is important – besides evaluating performance and usability – to assess the usefulness of the information returned as well as how it is presented. Within this context, our study found that the results presentation style employed by K-Search was the most liked by all users as shown in Tables 1 and 2. It is interesting to note how small details such as organising answers in a table or having a visually-appealing display (adopted by K-Search) have a direct impact on results readability and clarity and, in turn, user satisfaction. This is shown from the most repeated comments given for K-Search: "*I liked the way answers are displayed*" and "*results presentation was easy to interpret*". Additionally, K-Search is the only tool that did not present a URI for an answer but used a reference to the document using a NL label. This was favoured by users who often found URIs to be technical and more targeted towards domain experts. For instance, one user specifically mentioned having "*http://www.mooney.net/geo#tennesse2*" as an answer was not understandable. By examining the ontology, this was found to be the URI of *tennessee river* and it had the '2' at the end to differentiate it from *tennessee state*, which had the URI "*http://www.mooney.net/geo#tennesse*". This suggests that, unless users are very familiar with the data, presenting URIs alone is not very helpful. By analysing users feedback from a similar usability study, Elbedweihy et al. [21] found that when returning answers to users, each result should be augmented with associated information to provide a 'richer' user experience. This was similarly shown by users' feedback in our study with the following comments regarding potential improvements often given for all the tools:

- Maybe a 'mouse over' function with the results that show more information.
- Perhaps related information with the results.
- Providing similar searches would have been helpful.

For example, for a query requiring information about states, tools could go a step further and return extra information about each state – rather than only providing name and URI – such as the *capital, area, population* or *density*, among others. Furthermore, they could augment the results with ones associated with related concepts which might be of interest to users [42,43]. Again, these could be instances of *lakes or mountains* (examples of concepts related to *state*) found in a state. This notion of relatedness or relevancy is clearly domain-dependent and is itself a research challenge. In this context, Elbedweihy et al. [44] suggested a notion of relatedness based on collaborative knowledge found in query logs.

Benefit of Displaying Generated Formal Query Depends on User Type: While casual users often perceived the formal query generated by a tool as

Table 3. Query input time (in seconds) required by expert and casual users

User Type	Affective Graphs	Semantic Crystal	K-Search	Ginseng	NLP-Reduce	p-value
Expert Users	88.86	79.55	53.54	102.52	19.90	0.001
Casual Users	72.8	75.76	63.59	93.13	18.6	0.001

confusing, experts liked the ability to see the formal representation of their constructed query since it increased their confidence in what they were doing. Indeed, being able to perform direct changes to the formal query increased the expressiveness of the query language as perceived by expert users.

Experts Plan Query Formulation More Than Casuals: As shown in Table 3, with most of the tools, expert users took more time to build their queries than casual ones. The feedback showed that the latter often spent more time planning – and verbally describing – their rationale (e.g. "so it understands abbreviations and it seems to work better with sentences than with keywords") during query formulation. Interestingly, studies on user search behaviour found similar results: Tabatabai and Shore found that "*Novices were less patient and relied more on trial-and-error.*" [17, p.238] and Navarro-Prieto et al. showed that "*Experienced searchers ... planned in advance more than the novice participants*" [18, p.8].

4 Conclusions

In this paper, we have discussed a usability study of five semantic search tools employing four different query approaches: free-NL, controlled-NL, graph-based and form-based. The study – which used both expert and casual users – has identified a number of findings, the most important are summarised below.

Graph-based approaches were perceived by expert users as intuitive allowing them to formulate more complex queries, while casual users, despite finding them difficult to use, enjoyed the visually-appealing interfaces which created an overall pleasant search experience. Also, showing the entire ontology helped users to understand the data and the possible ways of constructing queries. However, unsurprisingly, graph-based approach was judged as laborious and time consuming. In this context, the form-based approach required less input time. It was also perceived as a midpoint between NL-based and graph-based, allowing more complex queries than the first, yet less complicated than the latter.

Additionally, casual users found the controlled-NL support to be very helpful whereas expert users found it to be very restrictive and thus preferred the flexibility and expressiveness offered by free-NL. A major challenge for the latter was the mismatch between users' query terms and ones expected by the tool (habitability problem). The results also support the literature showing that negation is a challenge for semantic search tools [24,25]: only one tool provided partial support for negation. Furthermore, the study showed that users often requested the search results to be augmented with more information to have a better understanding of the answers. They also mentioned the need for a more user-friendly

results presentation format. In this context, the most liked presentation was that employed by K-Search, providing results in a tabular format that was perceived as clear and visually-appealing.

To conclude, this usability study highlighted the advantage of visualising the search space offered by view-based query approaches. We suggest combining this with a NL-input feature that would balance difficulty and speed of query formulation. Indeed, providing *optional* guidance for the NL input could be the best way to cater to both expert and casual users within the same interface. These findings are important for developers of future query approaches and similar user interfaces who have to cater for different types of users with different preferences and needs. For future work and, indeed, to have a more complete picture, we plan to assess how the interaction with the search tools affect the information seeking process (usefulness). To achieve this, we will use questions with an overall goal – as opposed to ones which are not part of any overarching information need – and compare users' knowledge before and after the search task. This would also allow us to evaluate advanced features such as formulating complex queries, merging results of subqueries or assessing relevancy and usefulness of additional information presented with the results.

References

1. Tummarello, G., Delbru, R., Oren, E.: Sindice.com: Weaving the Open Linked Data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 552–565. Springer, Heidelberg (2007)
2. Kaufmann, E., Bernstein, A.: Evaluating the usability of natural language query languages and interfaces to semantic web knowledge bases. J. Web Sem. 8 (2010)
3. López, V., Motta, E., Uren, V.: PowerAqua: Fishing the Semantic Web. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 393–410. Springer, Heidelberg (2006)
4. Harth, A.: VisiNav: A system for visual search and navigation on web data. J. Web Sem. 8, 348–354 (2010)
5. Saracevic, T.: Evaluation of evaluation in information retrieval. In: Proc. SIGIR (1995)
6. Halpin, H., Herzig, D.M., Mika, P., Blanco, R., Pound, J., Thompson, H.S., Tran, D.T.: Evaluating Ad-Hoc Object Retrieval. In: Proc. IWEST 2010 Workshop (2010)
7. Cleverdon, C.W.: Report on the first stage of an investigation onto the comparative efficiency of indexing systems. Technical report, The College of Aeronautics, Cranfield, England (1960)
8. Spärck Jones, K.: Further reflections on TREC. Inf. Process. Manage. 36, 37–85 (2000)
9. Voorhees, E.M.: The Philosophy of Information Retrieval Evaluation. In: Peters, C., Braschler, M., Gonzalo, J., Kluck, M. (eds.) CLEF 2001. LNCS, vol. 2406, pp. 355–370. Springer, Heidelberg (2002)
10. Ingwersen, P., Järvelin, K.: The Turn: Integration of Information Seeking and Retrieval in Context. Springer (2005)

11. Salton, G.: Evaluation problems in interactive information retrieval. Information Storage and Retrieval 6, 29–44 (1970)
12. Tague, J., Schultz, R.: Evaluation of the user interface in an information retrieval system: A model. Inf. Process. Manage., 377–389 (1989)
13. Hersh, W., Over, P.: SIGIR workshop on interactive retrieval at TREC and beyond. SIGIR Forum 34 (2000)
14. Kelly, D., Lin, J.: Overview of the TREC 2006 ciQA task. SIGIR Forum 41, 107–116 (2007)
15. Xie, H.: Supporting ease-of-use and user control: desired features and structure of web-based online IR systems. Inf. Process. Manage. 39, 899–922 (2003)
16. Petrelli, D.: On the role of user-centred evaluation in the advancement of interactive information retrieval. Inf. Process. Manage. 44, 22–38 (2008)
17. Tabatabai, D., Shore, B.M.: How experts and novices search the Web. Library & Information Science Research 27, 222–248 (2005)
18. Navarro-Prieto, N., Scaife, M., Rogers, Y.: Cognitive strategies in web searching. In: Proc. the 5th Conference on Human Factors and the Web, vol. 2004, pp. 1–13 (1999)
19. Balog, K., Serdyukov, P., de Vries, A.: Overview of the TREC 2010 Entity Track. In: TREC 2010 Working Notes (2010)
20. Kaufmann, E., Bernstein, A.: How Useful Are Natural Language Interfaces to the Semantic Web for Casual End-Users? In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 281–294. Springer, Heidelberg (2007)
21. Elbedweihy, K., Wrigley, S.N., Ciravegna, F., Reinhard, D., Bernstein, A.: Evaluating semantic search systems to identify future directions of research. In: Proc. 2nd International Workshop on Evaluation of Semantic Technologies (IWEST 2012) (2012)
22. Hölscher, C., Strube, G.: Web search behavior of Internet experts and newbies. Comput. Netw. 33, 337–346 (2000)
23. Popescu, A.M., Etzioni, O., Kautz, H.: Towards a theory of natural language interfaces to databases. In: IUI 2003, pp. 149–157 (2003)
24. Damljanovic, D., Agatonovic, M., Cunningham, H.: FREyA: an Interactive Way of Querying Linked Data using Natural Language. In: Proc. QALD-1 Workshop
25. López, V., Fernández, M., Motta, E., Stieler, N.: PowerAqua: supporting users in querying and exploring the semantic web. Semantic Web 3 (2012)
26. Albert, W., Tullis, T., Tedesco, D.: Beyond the Usability Lab: Conducting Large-Scale User Experience Studies. Elsevier Science (2010)
27. Bernstein, A., Reinhard, D., Wrigley, S.N., Ciravegna, F.: Evaluation design and collection of test data for semantic search tools. Technical Report D13.1, SEALS Consortium (2009)
28. Miller, R.: Human Ease of Use Criteria and Their Tradeoffs. IBM, Systems Development Division (1971)
29. Kelly, D.: Methods for Evaluating Interactive Information Retrieval Systems with Users. Found. Trends Inf. Retr. 3, 1–224 (2009)
30. Hix, D., Hartson, H.R.: Developing User Interfaces: Ensuring Usability Through Product and Process. J. Wiley (1993)
31. Shneiderman, B.: Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison Wesley Longman (1998)
32. Ericsson, K.A., Simon, H.A.: Protocol analysis: Verbal reports as data. MIT Press (1993)

33. Brooke, J.: SUS: a quick and dirty usability scale. In: Usability Evaluation in Industry, pp. 189–194. CRC Press (1996)
34. Bangor, A., Kortum, P.T., Miller, J.T.: An Empirical Evaluation of the System Usability Scale. Int't J. Human-Computer Interaction 24, 574–594 (2008)
35. Bhagdev, R., Chapman, S., Ciravegna, F., Lanfranchi, V., Petrelli, D.: Hybrid Search: Effectively Combining Keywords and Semantic Searches. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 554–568. Springer, Heidelberg (2008)
36. Strauss, A., Corbin, J.: Basics of qualitative research: grounded theory procedures and techniques. Sage Publications (1990)
37. Bangor, A., Kortum, P.T., Miller, J.T.: Determining what individual SUS scores mean: Adding an adjective rating scale. J. Usability Studies 4, 114–123 (2009)
38. Uren, V., Lei, Y., López, V., Liu, H., Motta, E., Giordanino, M.: The usability of semantic search tools: a review. Knowledge Engineering Review 22, 361–377 (2007)
39. López, V., Motta, E., Uren, V., Sabou, M.: Literature review and state of the art on Semantic Question Answering (2007)
40. López, V., Uren, V., Sabou, M., Motta, E.: Is question answering fit for the Semantic Web? A survey. Semantic Web 2, 125–155 (2011)
41. Uren, V., Lei, Y., López, V., Liu, H., Motta, E., Giordanino, M.: The usability of semantic search tools: a review. The Knowledge Eng. Rev. 22, 361–377 (2007)
42. Meij, E., Mika, P., Zaragoza, H.: Investigating the Demand Side of Semantic Search through Query Log Analysis. In: Proc. SemSearch 2009 (2009)
43. Meij, E., Bron, M., Hollink, L., Huurnink, B., de Rijke, M.: Mapping queries to the Linking Open Data cloud: A case study using DBpedia. J. Web Sem. 9, 418–433 (2011)
44. Elbedweihy, K., Wrigley, S.N., Ciravegna, F.: Improving Semantic Search Using Query Log Analysis. In: Proc. Interacting with Linked Data (ILD) 2012 Workshop (2012)

Extracting Justifications from BioPortal Ontologies

Matthew Horridge[1], Bijan Parsia[2], and Ulrike Sattler[2]

[1] Stanford University
California, USA, 94305
matthew.horridge@stanford.edu
[2] The University of Manchester
Oxford Road, Manchester, M13 9PL
bparsia@cs.man.ac.uk

Abstract. This paper presents an evaluation of state of the art black box justification finding algorithms on the NCBO BioPortal ontology corpus. This corpus represents a set of naturally occurring ontologies that vary greatly in size and expressivity. The results paint a picture of the performance that can be expected when finding all justifications for entailments using black box justification finding techniques. The results also show that many naturally occurring ontologies exhibit a rich justificatory structure, with some ontologies having extremely high numbers of justifications per entailment.

1 Introduction

A justification $\mathcal{J}$ for an entailment η in an ontology $\mathcal{O}$ is a minimal subset of $\mathcal{O}$ that is sufficient to entail η. More precisely, $\mathcal{J}$ is a justification for $\mathcal{O} \models \eta$ (read as $\mathcal{O}$ entails η) if $\mathcal{J} \subseteq \mathcal{O}$, $\mathcal{J} \models \eta$ and for all $\mathcal{J}' \subsetneq \mathcal{J}$ $\mathcal{J}' \not\models \eta$. There can be multiple, possibly overlapping, justifications for a given ontology and entailment. Depending upon context, justifications are also known as MUPS (Minimal Unsatisfiability Preserving Sub-TBoxes) [25] or MINAS (Minimal Axiom Sets) [2].

A *justification finding service* computes justifications for an ontology and an entailment. An implementation of a justification finding service is a key component in many of the *explanation* and *debugging* tools that exist for ontology development environments such as Swoop [17], the RaDON plugin for the NeOn Toolkit [14], the explanation workbench for Protégé-4 [12], the explanation facility in OWL Sight [8], and the explanation view in TopBraid Composer [19]. Justification finding services are also increasingly being used as auxiliary services in other applications for example in incremental reasoning [3], reasoning over very large ABoxes [4], belief base revision [9], meta-modelling support [5], default reasoning [24], eliminating redundant axioms in ontologies [7], and laconic justification finding [13].

Given the prominence and importance of justifications, it is no surprise that there is a large literature on techniques and optimisations for computing them.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 287–299, 2012.

Much of this literature [31,30,16,20,18,26,27,26,15] is focused on empirical investigations which, generally speaking, are undertaken to validate specific performance optimisations and implementation techniques. This begs the question as to why further empirical investigation is required. In essence, most of the existing empirical work is performed with *prototype implementations* on small collections of ontologies that are *chosen* to show off the effect of specific optimisations and demonstrate proof of concept. This is obviously a completely valid thing to do. However, many of the experiments do not provide a true picture of how highly optimised and robustly implemented justification finding techniques, that take advantage of all published optimisations, will perform on state of the art ontologies that are now widely available. In addition to this, none of the existing experiments were designed to investigate the justification landscape of a broad corpus of ontologies—that is, there is very little data about the numbers and sizes of justifications that one could encounter in naturally occurring ontologies.

The overall aim of this paper is to therefore present a thorough investigation into the practicalities of computing all justifications for entailments in published, naturally occurring ontologies. In particular, ontologies which are representative of typical modelling and are not tutorial or reasoner test-bed ontologies. The end goal is to provide a view of how modern, robustly implemented and highly optimised justification finding algorithms, coupled with modern highly optimised reasoners, perform on realistic inputs, and to paint a picture of the richness of the justification landscape.

All of the data and software, including ontologies, extracted entailments, entailment test timings, hitting set tree statistics, justifications and other raw results, is available online[1] for third parties to access. It should be of interest to those working in justification based research, ontology comprehension, reasoner development, and module extraction amongst other areas.

2 Justification Finding Techniques

In general, algorithms for computing justifications are described using two axes of classification. The first, the *single-all-axis* is whether an algorithm computes a *single* justification for an entailment or whether it computes *all* justifications for an entailment. The second, the *reasoner-coupling-axis* is whether the algorithm is a *black-box* algorithm or whether it is a *glass-box* algorithm. The categorisation is based entirely on the part played by reasoning during the computation of justifications. In essence, justifications are computed as a *direct consequence of reasoning* in glass-box algorithms, whereas they are *not* computed as a direct consequence of reasoning in black-box algorithms. In this sense, glass-box algorithms are tightly interwoven with reasoning algorithms, whereas black-box algorithms simply use reasoning to compute whether or not an entailment follows from a set of axioms. Because of space constraints, and the fact that black-box justification finding services work with any OWL reasoner, this paper focuses entirely on results for black-box justification finding. A detailed analysis of the

[1] http://www.stanford.edu/~horridge/publications/2012/iswc/justextract

differences between glass-box and black-box justification finding algorithm performance may be found in [10]. An advantage of black-box algorithms is that they can be easily and robustly implemented [16]. A perceived disadvantage of black-box algorithms is that they can be inefficient and impractical due to a potentially large search space [29].

Black-Box Algorithms for Computing Single Justifications. The basic idea behind a black-box justification finding algorithm is to systematically test different subsets of an ontology in order to find one that corresponds to a justification. Subsets of an ontology are typically explored using an "expand-contract" strategy. In order to compute a justification for $\mathcal{O} \models \eta$, an initial, small, subset $\mathcal{S}$ of $\mathcal{O}$ is selected. The axioms in $\mathcal{S}$ are typically the axioms whose signature has a non-empty intersection with the signature of η, or axioms that "define"[2] terms in the signature of η. A reasoner is then used to check if $\mathcal{S} \models \eta$, and if not, $\mathcal{S}$ is expanded by adding a few more axioms from $\mathcal{O}$. This *incremental expansion* phase continues until $\mathcal{S}$ is large enough so that it entails η. When this happens, either $\mathcal{S}$, or some subset of $\mathcal{S}$, is *guaranteed* to be a justification for η. At this point $\mathcal{S}$ is gradually *contracted* until it is a *minimal* set of axioms that entails η i.e. a justification for η in $\mathcal{O}$.

Black-Box Algorithms for Computing All Justifications. When formulating a repair plan for an entailment, or attempting to understand an entailment, it is usually necessary to compute *all* justifications for that entailment. This can be achieved using black-box techniques for finding single justifications in combination with techniques that are borrowed from the field of *model based diagnosis* [1]. Specifically, all justifications can be computed by performing systematic "repairs" of the ontology in question, which eliminate already found justifications, and searching for new justifications after each repair. In order to compute these repairs a classical *a hitting set tree* based algorithm is used. A full discussion of this algorithm, which is based on seminal work by Reiter [23], is beyond the scope of this paper, but suffice it to say, the algorithm is widely used in various fields, has been well used and documented in the field of computing justifications [16], and is reasonably well understood in this area. Due to space constraints a more detailed presentation is not offered here, but a comprehensive overview may be found in Chapter 3 of [10].

3 Materials (The BioPortal Corpus)

The number of published real world ontologies has grown significantly since computing justifications for entailments in OWL ontologies was first investigated from around 2003 onwards. In particular, in the last three years the number of ontologies in the biomedical arena has grown considerably. Many of these ontologies have been made available via the *NCBO BioPortal* ontology repository [22]. At the time of writing, BioPortal provides access to the imports closures of over 250 bio-medical ontologies in various formats, including OWL and OBO[3] [28].

[2] For example, the axiom $A \sqsubseteq B$ defines the class name A.

[3] OBO may be seen as an additional serialisation syntax for OWL.

Not only is BioPortal useful for end users who want to share and use biomedical ontologies, it is also useful for ontology tools developers as it provides a corpus of ontologies that is attractive for the purposes of implementation testing. In particular, it provides ontologies that: vary greatly in size; vary greatly in expressivity; are real world ontologies; are developed by a wide range of groups and developers and contain a wide variety of modelling styles; and finally, are not "cherry picked" to show good performance of tools.

Curation Procedure. The BioPortal ontology repository was accessed on the 12th March 2011 using the BioPortal RESTful Service API. In total, 261 ontology documents (and their imports closures) were listed as being available. Out of these, there were 125 OWL ontology documents, and 101 OBO ontology documents, giving a total of 226 "OWL compatible" ontology documents that could theoretically be parsed into OWL ontologies.

Parsing and Checking. Each listed OWL compatible ontology document was downloaded and parsed by the OWL API. OBO ontology documents were parsed according to the lossless OWL-OBO translation given in [6] and [21]. Any imports statements were recursively dealt with by downloading the document at the imports statement URL and parsing it into the imports closure of the original BioPortal "root" ontology. Each axiom that was parsed into the imports closure was labelled with the name of the ontology document from where it originated. If an imported ontology document could not be accessed (for whatever reason) the import was silently ignored.

Out of the 226 OWL compatible ontology documents that were listed by the BioPortal API, 7 could not be downloaded due to HTTP 500[4] errors, and 1 ontology could not be parsed due to syntax errors. This left a total of 218 OWL and OBO ontology documents that could be downloaded parsed into OWL ontologies. After parsing, four of the ontologies were found to violate the OWL 2 DL global restrictions. In all cases, the violation was caused by the use of transitive (non-simple) properties in cardinality restrictions. These ontologies were discarded and were not processed any further, which left 214 ontologies.

Entailment Extraction. Three reasoners were used for entailment extraction: FaCT++, HermiT and Pellet. Each ontology was checked for consistency. Five of the 214 ontologies were found to be inconsistent. Next, each *consistent* ontology was classified and realised in order to extract entailments to be used in the justification finding experiments. Entailed *direct* subsumptions between named classes (i.e. axioms of the form $A \sqsubseteq B$) were extracted, along with *direct* class assertions between named individuals and named classes (i.e. axioms of the form $A(a)$). It was decided that these kinds of entailments should be used for testing purposes because they are the kinds of entailments that are exposed through the user interfaces of tools such as Protégé-4 and other ontology browsers—they are therefore the kinds of entailments that users of these tools typically seek justifications (explanations) for. The set of entailments for each ontology

[4] An HTTP 500 error is an error code that indicated the web server encountered an internal error that prevented it from fulfilling the client request.

was then filtered so that it only contained *non-trivial* entailments in accordance with Definition 1.

Definition 1 (Non-Trivial Entailment). *Given an ontology $\mathcal{O}$, such that $\mathcal{O} \models \alpha$, the entailment α in $\mathcal{O}$ is non-trivial if $\mathcal{O} \setminus \{\alpha\} \models \alpha$*

Intuitively, for an ontology $\mathcal{O}$ and an entailment α such that $\mathcal{O} \models \alpha$, α is a non-trivial entailment in $\mathcal{O}$ either if α is not asserted in $\mathcal{O}$ (i.e. $\alpha \notin \mathcal{O}$) or, α is asserted in $\mathcal{O}$ (i.e. $\alpha \in \mathcal{O}$) but $\mathcal{O} \setminus \{\alpha\} \models \alpha$, i.e. $\mathcal{O}$ with α removed still entails α. In total there were 72 ontologies with non-trivial entailments which accounts for just over one third of the consistent OWL and OBO ontologies contained in BioPortal.

Reasoner Performance. Due to practical considerations, a timeout of 30 minutes of CPU time was set for each task of consistency checking, classification and realisation. There were just three ontologies, for which consistency checking (and hence classification and realisation) could not be completed within this time out. These were: GALEN, the Foundational Model of Anatomy (FMA) and NCBI Organismal Classification. These ontologies were discarded and not processed any further.

Ontologies With Non-Trivial Entailments. There were 72 BioPortal ontologies that contained at least one non-trivial entailment. The list of these ontologies may be found in [10] and is available online as a summary[5]. For these ontologies, the average number of logical axioms (i.e. non-annotation axioms) per ontology was 10,645 (SD=31,333, Min=13, Max=176,113). The average number of non-trivial entailments per ontology was 1,548 (SD=6,187, Min=1, Max=49,537). The expressivity of the BioPortal ontologies with non-trivial entailments ranged from $\mathcal{EL}$ and $\mathcal{EL}$++ (corresponding to the OWL2EL profile) through to $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$ (the full expressivity of OWL 2 DL).

In summary, the ontology corpus provided by the BioPortal exhibits varying numbers of non-trivial entailments with a wide range of expressivities. It reflects current modelling practices and the kinds of ontologies that people use in tools.

4 Method and Results

All of the experiments detailed below were carried out using Pellet version 2.2.0[6]. For ontology loading, manipulation and reasoner interaction, the OWL API [11] version 3.2.2 was used. The OWL API has support for manipulating ontologies at the level of axioms, and so it is entirely suited for the implementation of the justification finding algorithms.

[5] http://www.stanford.edu/ horridge/publications/2012/iswc/justextract/data/bioportal-corpus-non-trivial-entailment-summary.pdf

[6] The primary reason for using Pellet was that Pellet provides robust implementations of the OWL API reasoner interfaces and has reliable support for setting timeouts—a feature that is crucial for long running experiments.

Having introduced the overall test setup, the experiment is now described in detail.

Algorithm Implementation. The black box algorithm for finding all justifications for an entailment, and its sub-routine algorithms, presented in [10] (Algorithms 4.1, 4.2, 4.5 and 4.6) were implemented in Java against the OWL API version 3.2.0. In essence this algorithm (Algorithm 4.1) is the de-facto standard *black box* algorithm for computing all justifications for an entailment. It does this by constructing a hitting set tree, using standard hitting set tree optimisations such as node reuse, early path termination etc. Its sub-routine algorithm for finding single justifications (Algorithm 4.2) uses the expand contract technique, where expansion is incremental, done by modularisation and selection function (Algorithm 4.5), and contraction is done using a divide and conquer strategy (Algorithm 4.6).

Test Data. The test data consisted of the 72 BioPortal ontologies that contained non-trivial entailments. For each ontology, the set of *all* non-trivial direct sub-concept ($A \sqsubseteq B$) and direct concept assertion ($A(a)$) entailments were extracted and paired up with the ontology.

Method. The experiments were performed on MacBook Pro with a 3.06 GHz Intel Core 2 Duo Processor. The Java Virtual Machine was allocated a maximum of 4 GB of RAM. Pellet 2.2.2 was used as a backing reasoner for performing entailment checks, with each entailment check consisting of a load, followed by a query to ask whether or not the entailment held. The algorithm implementation described above was used to compute all justifications for each non-trivial entailment for each ontology. For each entailment, the CPU time for computing all justifications was measured, along with the number and sizes of justifications. For the sake of practicalities, because some ontologies have tens of thousands of non-trivial entailments (e.g. 49,000+ entailments for the coriell-cell-line ontology), a soft time limit of 10 minutes was imposed on computing all justifications for any one entailment. Additionally, an entailment test time limit of 5 minutes was placed on entailment checking.

Results.[7] Figure 1 shows a percentile plot for time to compute all justifications. Note that mean values for each ontology are shown as transparent bars with white outlines. The x-axis, which shows Ontology Id, is ordered by the value of the 99th percentile. This percentile was chosen because it provides a good picture of how the algorithm will perform in practice for the vast majority of entailments. It also draws out the remaining 1 percent of outliers rather clearly.

There were *seven* ontologies that contained one or more entailments for which it was *not* possible to compute *all* justifications. These ontologies, along with the total number of entailments and the number of failed entailments are shown in Table 1. Table 1 also shows the mean/max number of justifications and

[7] Large scalable plots of figures, and a spreadsheet containing the data used to generate them can be found at
http://www.stanford.edu/~horridge/publications/2012/iswc/justextract/

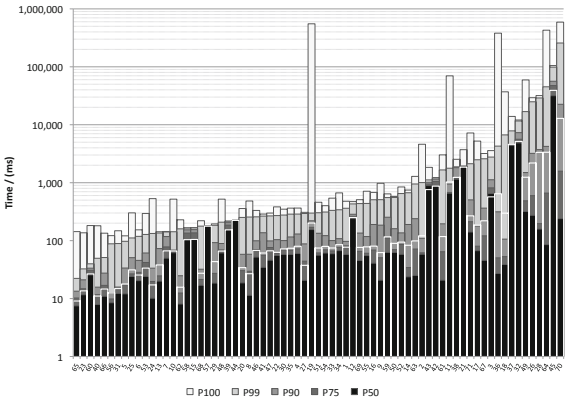


Fig. 1. Percentile and Mean Times to Compute All Justifications. Mean times are shown in white outlines. The x-axis (Ontology) is sorted by the 99th percentile (P99).

mean/max justification size (number of axioms per justification) per failed entailment and the mean entailment checking times per failed entailment. In these seven ontologies there were three ontologies for which the failures occurred over less than one percent of entailments tested, a further three ontologies where the failures occurred for less than 7 percent of entailments tested, and one final ontology, where failures occurred for almost 75 percent of entailments tested. In this last ontology *all* of the failures were due to *entailment checking timeouts*. The failures relating to all of the other ontologies were due to timeouts during *construction of the hitting set tree*, which became too large to search within a period of 10 minutes.

Figure 2 and Figure 3 provide a picture of the justification landscape for the BioPortal ontologies. Figure 2 shows the the mean number of justifications per entailment per percentile. It should be noted that the percentiles are calculated from a *reverse* ordering of entailments based on justification size. That is, the nth percentile contains n percent of entailments that have the *largest* number of justifications. The x-axis in Figure 2 is ordered by the mean value of the 100th percentile (i.e. mean number of justifications per entailment). Figure 3 shows the mean number of axioms per justification per percentile along with the maximum

Table 1. Black-Box Find All Timeouts

Ont.	Ents.	Failed	% Failed	Computed Justifications				Entailment Check Time / (ms)		
				Number		Size				
				Mean	Max	Mean	Max	Mean	SD	Max
19	49537	16	0.03	40.6	65	15.7	23	1.1	0.4	3
36	2230	1	0.04	414.0	414	13.5	17	0.5	0.3	2
64	566	4	0.71	1284.5	1411	25.1	28	2.5	1.6	13
70	3997	188	4.70	448.5	1060	12.7	24	7.0	77.8	141,721
45	148	9	6.08	1.6	2	21.9	24	1,979.9	6,762.4	41,840
3	44	3	6.82	1271.3	1494	26.9	35	2.5	1.3	10
32	35	26	74.29	2.2	7	2.5	8	2,601.2	23,781.0	270,004

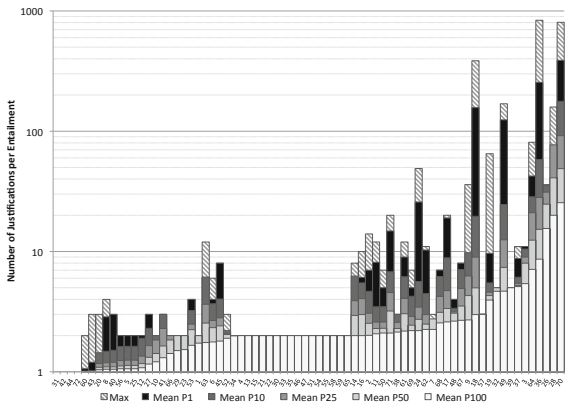


Fig. 2. The Mean Number of Justifications per Entailment for Various Percentiles. Percentiles of the Entailments Sorted by the Number of Justifications in Descending Order—e.g. P10 represents the top 10 percent of entailments with the highest number of justifications per entailment.

number of axioms per entailment. The percentiles are calculated from a reverse ordering on justification size. For example, the nth percentile contains n percent of justifications that have a mean size *greater* than the mean of that percentile.

5 Analysis and Discusssion

The Practicalities of Computing All Justifications for An Entailment. Out of the 72 ontologies it was possible to compute all justifications for all direct atomic subconcept and concept assertion entailments in 65 ontologies. There were seven ontologies that contained some entailments for which not all justifications could be computed. These failures are discussed below, however, the results from this experiment provide strong empirical evidence that it is largely practical to compute all justifications for these kinds of entailments in the BioPortal ontologies. Although the results cannot be statistically generalised to ontologies outside of the BioPortal corpus it is reasonable to assume that the results are suggestive for other real world ontologies.

Reasons for Failures. Seven of the 72 ontologies contained entailments for which not all justifications could be computed. Broadly speaking there were two reasons for this: (1) The justifications for each failed entailment were numerous and large in size. This resulted in the size of the hitting set tree growing to a limit where it was not possible to close all branches within 10 minutes. In particular, for Ontology 36 the hitting set tree grew to over 3 million nodes, and for Ontology 70 the hitting set tree grew to over 1.6 million nodes. This compares to hitting set tree sizes in the tens of thousands for successful entailments. (2) Entailment checking performance was such that the number of entailment

checks, in combination with the time for each check, made it impossible to construct the hitting set tree within 10 minutes. This was the case with Ontology 45 and Ontology 32, both of which had average entailment checking times that were three orders of magnitude higher than for other ontologies. This problem was particularly endemic for Ontology 32, which suffered the largest number of failures, and had the worst entailment checking performance of all ontologies (M=2,601.2 ms, SD=23,781.0 ms, MAX=270,004 ms). Leaving aside entailment checking performance problems, which can be regarded as being out of the scope of control of this work, the number of justifications that were computable for failed entailments was very high. For example, for Ontology 3, which percentage-wise suffered the highest number of failures, the implementation was still able to compute on average 1271 justifications per failed entailment, with a maximum of 1494 justifications. With the exception of Ontology 32, which had a very high percentage of failures due to poor entailment checking performance, it is fair to say that over the whole corpus, and within individual ontologies, the failure rate is low to very low, thus indicating the robustness of the algorithms on real world ontologies. When the algorithm does fail to find *all* justifications, it is still possible to find *some* justifications, and the number of found justifications tends to be very large.

The Acceptability of Times for Computing All Justifications. As can be seen from Figure 1, the majority of ontologies contained entailments for which all justifications could be computed within 1 second. For all but six ontologies, it was possible to compute all justifications for 99 percent of entailments within 10 seconds. Only two ontologies required longer than one minute for computing all justifications for 99 percent of entailments in these ontologies, with 90 percent of entailments in these ontologies falling below the one minute mark. It is clear to see that there are some outlying entailments in the corpus. In particular, Ontology 19 (the Coriell Cell Line Ontology) contains the most significant outlier, with one percent of entailments in this ontology requiring almost 150 seconds for computing all justifications. However, it appears that the times are perfectly acceptable for the purposes of generating justifications for debugging or repair in ontology development environments.

The Number of Justifications per Entailment. As can be seen from Figure 2, the number of justifications per entailment varied over much of the BioPortal corpus. There were just four ontologies which had on average one justification per entailment. Even ontologies with low average numbers of justification per entailment did exhibit some entailments with large numbers of justifications as evidenced by the band of ontologies from 60 to 52 on the left hand side of Figure 2. On the right hand side of Figure 2, the band of ontologies from 14 through to 70 represent ontologies with very large numbers of justifications per entailment. For example, Ontology 70 had on average 25 justifications per entailment, with 10 percent of entailments having over 177 justifications, and 50 percent of entailments having over 48 justifications. This was closely followed by Ontology 28, which had on average 20 justifications per entailment, with 50 percent of entailments having over 40 justifications. The maximum number of

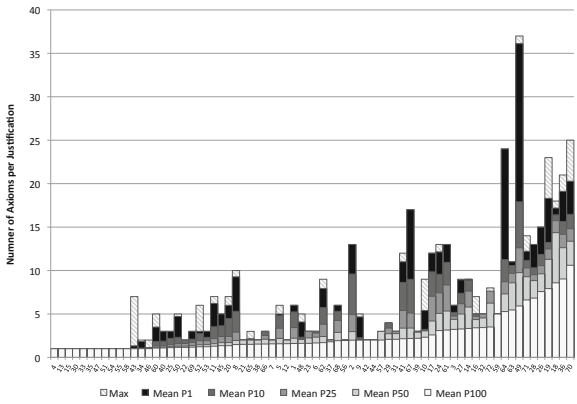


Fig. 3. Mean Number of Axioms per Justification. Percentiles of the Entailments Sorted by the Number of Justifications in Descending Order—e.g. P10 represents the top 10 percent of entailments with the highest number of justifications per entailment.

justifications for any one entailment occurred in Ontology 36, which had 837 justifications for one entailment. At this point it is worth noting that *none* of the empirical work detailed in the literature has uncovered ontologies with these large and very large numbers of justifications per entailment.

The Size of Justifications. Figure 3 shows the mean numbers of axioms per justification per ontology. The ontologies are ordered by mean number of axioms per justification. There is a clear band of ontologies to the left hand side of Figure 3 that only have, on average, one axiom per justification. Recall that each entailment is a non-trivial entailment, which means that these justifications are not simply "self" justifications. In general the mean values (100th percentile) for each ontology are fairly low, with only 11 ontologies (shown on the right hand side of Figure 3) having over 5 axioms per justification on average . However, as witnessed by the 1st, 10th, 25th and 50th percentile columns in Figure 3, there are in fact many ontologies with many entailments that have larger numbers of axioms per justification. For example there are 10 ontologies where 50 percent of justifications contained over 7 axioms, and 10 percent of justifications contained 10 to 16 axioms. At the top end of the scale, several ontologies contained justifications with very large numbers of axioms. For example Ontologies 48, 50, 63, 18 and 41 contained justifications with 21, 23, 24, 25 and 37 axioms respectively. Finally it should be noted that these larger justifications do not simply consist of long chains of atomic subclass axioms. All in all, the number of justifications per entailment, and the size of justifications points to considerable logical richness being present in many ontologies in the BioPortal corpus.

6 Conclusions and Suggestions for Future Work

The detailed empirical investigation that has been carried out and presented in this paper provides strong evidence to conclude that computing all justifications for direct class subsumption and direct class assertion entailments in the BioPortal corpus of consistent ontologies is practical. That is, for the vast majority of entailments in the majority of ontologies all justifications can be computed in under 10 minutes of CPU time. In essence, the justification finding algorithms used in the empirical evaluation here show good and robust runtime performance on realistic inputs.

While model based diagnosis techniques for computing all justifications fare extremely well, there are examples of entailments in realistic consistent ontologies for which it is not possible to compute all justifications. In these cases an incomplete solution, which could still consist of hundreds of justifications, must be accepted.

The number of justifications for entailments in naturally occurring domain ontologies can be very high. In the work presented here the number peaked at around 1000 justifications per entailment. The sizes of justifications in these ontologies can be very large, peaking at around 40 axioms per justification. The majority of ontologies with non-trivial entailments have multiple justifications per entailment with multiple axioms per justification.

In terms of future work, there are several strands that should be pursued. The first is that the experiments described here should be replicated and verified by third parties. It is reasonable to assume that the results presented here will be repeatable with other OWL reasoners, but this should ideally be investigated and verified. Finally, the experiments should be carried out on different ontology corpora. At the time of writing, third party non-biomedical-ontology installations of the BioPortal software are coming online. It would be interesting to compare the repositories of ontologies from different communities, in terms of non-trivial entailments, number justifications per entailment, size of justifications etc. and see how the justificatory structure and modelling style varies from one community to another.

References

1. Readings in Model Based Diagnosis. Morgan Kaufmann Publishers Inc. (1992)
2. Baader, F., Peñaloza, R.: Axiom pinpointing in general tableaux. Journal of Logic Computation 20(1), 5–34 (2010)
3. Cuenca Grau, B., Halaschek-Wiener, C., Kazakov, Y.: History Matters: Incremental Ontology Reasoning Using Modules. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 183–196. Springer, Heidelberg (2007)
4. Dolby, J., Fokoue, A., Kalyanpur, A., Kershenbaum, A., Schonberg, E., Srinivas, K., Ma, L.: Scalable semantic retrieval through summarization and refinement. In: AAAI 2007 (2007)

5. Glimm, B., Rudolph, S., Völker, J.: Integrated Metamodeling and Diagnosis in OWL 2. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 257–272. Springer, Heidelberg (2010)
6. Golbreich, C., Horrocks, I.: The OBO to OWL mapping, GO to OWL 1.1! In: OWLED 2007 (2007)
7. Grimm, S., Wissmann, J.: Elimination of Redundancy in Ontologies. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part I. LNCS, vol. 6643, pp. 260–274. Springer, Heidelberg (2011)
8. Grove, M.: OWLSight (October 2009), http://pellet.owldl.com/ontology-browser
9. Halaschek-Wiener, C., Katz, Y., Parsia, B.: Belief base revision for expressive description logics. In: OWLED 2006 (2006)
10. Horridge, M.: Justification Based Explanation in Ontologies. Ph.D. thesis, School of Computer Science, The University of Manchester (2011)
11. Horridge, M., Bechhofer, S.: The OWL API: A Java API for OWL ontologies. Semantic Web 2(1), 11–21 (2011)
12. Horridge, M., Parsia, B., Sattler, U.: Explanation of OWL entailments in Protégé-4. Poster and Demo Track, ISWC 2008 (2008)
13. Horridge, M., Parsia, B., Sattler, U.: Laconic and Precise Justifications in OWL. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 323–338. Springer, Heidelberg (2008)
14. Ji, Q., Haase, P., Qi, G., Hitzler, P., Stadtmüller, S.: RaDON — Repair and Diagnosis in Ontology Networks. In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 863–867. Springer, Heidelberg (2009)
15. Ji, Q., Qi, G., Haase, P.: A Relevance-Directed Algorithm for Finding Justifications of DL Entailments. In: Gómez-Pérez, A., Yu, Y., Ding, Y. (eds.) ASWC 2009. LNCS, vol. 5926, pp. 306–320. Springer, Heidelberg (2009)
16. Kalyanpur, A.: Debugging and Repair of OWL Ontologies. Ph.D. thesis, The Graduate School of the University of Maryland (2006)
17. Kalyanpur, A., Parsia, B., Hendler, J.: A tool for working with web ontologies. International Journal on Semantic Web and Information Systems 1 (2005)
18. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding All Justifications of OWL DL Entailments. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 267–280. Springer, Heidelberg (2007)
19. Knublauch, H.: Composing the semantic web: Explaining inferences, http://composing-the-semantic-web.blogspot.com/2007/08/explanining-inferences.html
20. Meyer, T., Lee, K., Booth, R., Pan, J.Z.: Finding maximally satisfiable terminologies for the description logic $\mathcal{ALC}$. In: AAAI 2006 (2006)
21. Mungall, C.: OBO Flat File Format 1.4 syntax and semantics (February 2011), ftp://ftp.geneontology.org/pub/go/www/obo-syntax.html
22. Noy, N.F.: BioPortal: Ontologies and integrated data resources at the click of a mouse. Nucleic Acids Research 37 (May 2009)
23. Reiter, R.: A theory of diagnosis from first principles. Artificial Intelligence 32, 57–95 (1987)

24. Scharrenbach, T., d'Amato, C., Fanizzi, N., Grütter, R., Waldvogel, B., Bernstein, A.: Default Logics for Plausible Reasoning with Controversial Axioms. In: Bobillo, F. (ed.) URSW 2010 (2010)
25. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: IJCAI 2003 (2003)
26. Schlobach, S., Huang, Z., Cornet, R., van Harmelen, F.: Debugging incoherent terminologies. Journal of Automated Reasoning 39, 317–349 (2007)
27. Shchekotykhin, K., Friedrich, G., Jannach, D.: On computing minimal conflicts for ontology debugging. In: ECAI 2008 (2008)
28. Smith, B.: The OBO Foundary: Coordinated evolution of ontology to support biomedical data integration. Nature Biotechnology
29. Stuckenschmidt, H.: Debugging OWL ontologies - a reality check. In: EON-SWSC 2008 (2008)
30. Suntisrivaraporn, B.: Polynomial-Time Reasoning Support for Design and Maintenance of Large-Scale Biomedical Ontologies. Ph.D. thesis, Technical University of Dresden (2009)
31. Suntisrivaraporn, B., Qi, G., Ji, Q., Haase, P.: A Modularization-Based Approach to Finding All Justifications for OWL DL Entailments. In: Domingue, J., Anutariya, C. (eds.) ASWC 2008. LNCS, vol. 5367, pp. 1–15. Springer, Heidelberg (2008)

Linked Stream Data Processing Engines: Facts and Figures*

Danh Le-Phuoc[1], Minh Dao-Tran[2], Minh-Duc Pham[3],
Peter Boncz[3], Thomas Eiter[2], and Michael Fink[2]

[1] Digital Enterprise Research Institute, National University of Ireland, Galway
danh.lephuoc@deri.org
[2] Institut für Informationssysteme, Technische Universität Wien
{dao,eiter,fink}@kr.tuwien.ac.at
[3] Centrum Wiskunde & Informatica, Amsterdam
{p.minh.duc,p.boncz}@cwi.nl

Abstract. Linked Stream Data, i.e., the RDF data model extended for representing stream data generated from sensors social network applications, is gaining popularity. This has motivated considerable work on developing corresponding data models associated with processing engines. However, current implemented engines have not been thoroughly evaluated to assess their capabilities. For reasonable systematic evaluations, in this work we propose a novel, customizable evaluation framework and a corresponding methodology for realistic data generation, system testing, and result analysis. Based on this evaluation environment, extensive experiments have been conducted in order to compare the state-of-the-art LSD engines wrt. qualitative and quantitative properties, taking into account the underlying principles of stream processing. Consequently, we provide a detailed analysis of the experimental outcomes that reveal useful findings for improving current and future engines.

1 Introduction

Linked Stream Data [18] (LSD), that is, the RDF data model extended for representing stream data generated from sensors and social network applications, is gaining popularity with systems such as Semantic System S [8], Semantic Sensor Web [19] and BOTTARI [10]. Several platforms have been proposed as processing engines for LSD, including Streaming SPARQL [7], C-SPARQL [5], EP-SPARQL [2] on top of ETALIS, SPARQL$_{stream}$ [9], and CQELS [14]. By extending SPARQL to allow continuous queries over LSD and Linked Data, these engines bring a whole new range of interesting applications that integrate stream data with the Linked Data Cloud.

All above platforms except Streaming SPARQL provide publicly accessible implementations. As processing LSD has gained considerable interest, it is desirable to have a comparative view of those implementations by evaluating them on common criteria

* This research has been supported by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-II), by Marie Curie action IRSES under Grant No. 24761 (Net2), by the Austrian Science Fund (FWF) project P20841, and by the European Commission under contract number FP720117287661 (GAMBAS) and FP72007257943 (LOD2).

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 300–312, 2012.

in an *open benchmarking framework*. Such a framework is also valuable in positioning new engines against existing ones, and in serving as an evaluation environment for application developers to choose appropriate engines by judging them on criteria of interest.

Unfortunately, no benchmarking systems for LSD processing exist. Close to one is Linear Road Benchmark [4], which however is designed for relational stream processing systems and thus not suitable to evaluate graph-based queries on LSD processing engines. Furthermore, [14] provides a simplistic comparison of CQELS, CSPARQL, and ETALIS, which is based on a fixed dataset with a simple data schema, simple query patterns, and just considers average query execution time as the single aspect to measure the performance. With further experience and studies on theoretical/technical foundations of LSD processing engines [15], we observed that the following evaluation-related characteristics of these engines are critically important.

– The difference in semantics has to be respected, as the engines introduce their own languages based on SPARQL and similar features from Continuous Query Language [3];
– The execution mechanisms are also different. CSPARQL uses *periodical execution*, i.e., the system is scheduled to execute periodically (time-driven) independent of the arrival of data and its incoming rate. On the other hand, CQELS and ETALIS follow the *eager execution* strategy, i.e., the execution is triggered as soon as data is fed to the system (data-driven). Based on opposite philosophies, the two strategies have a large impact on the difference of output results.
– For a single engine, any change in the running environment and experiment parameters can lead to different outputs for a single test.

All these characteristics make a meaningful comparison of stream engines a nontrivial task. To address this problem, we propose methods and a framework to facilitate such meaningful comparisons of LSD processing engines wrt. various aspects. Our major contribution is a framework coming with several customizable tools for simulating realistic data, running engines, and analyzing the output. Exploiting this framework, we carry out an extensive set of experiments on existing engines and report the findings.

The outline of the paper is as follows. Section 2 describes our evaluation framework, including the data generator for customizable data from realistic scenarios. In Section 3 we present experimentation methodology and results, including design and strategies (Section 3.1), as well as several testing aspects and outcomes ranging from functionality, correctness, to performance (Sections 3.2–3.4). Section 4 reports general findings and provides a discussion, while Section 5 considers related benchmarking systems. Finally, Section 6 concludes the paper with an outlook to future work.

2 Evaluation Framework

Social networks (SNs) provide rich resources of interesting stream data, such as sequences of social discussions, photo uploading, etc. Viewed as highly-connected graphs of users, SNs is an ideal evaluation scenario to create interesting test cases on graph-based data streams. Therefore, our evaluation environment provides a data generator to realistically simulate such data of SNs. Next, the graph-based stream data schema and the data generator are described.

2.1 Graph-Based Stream Data Schema

The data schema is illustrated by a data snapshot in Figure 1. This snapshot has two layers for *stream data* and *static data* corresponding to what users continuously generate from their social network activities and user metadata including user profiles, social network relationships, etc. The details of these two layers are described below.

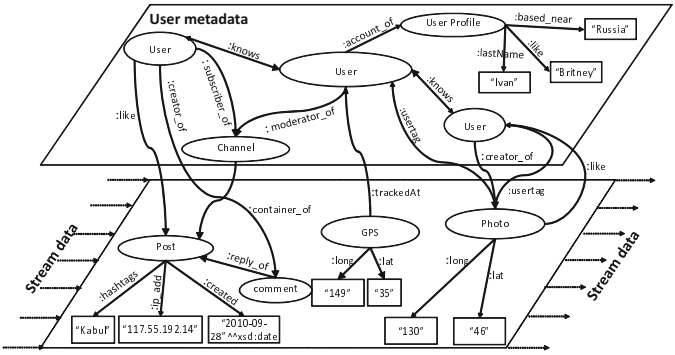


Fig. 1. Logical schema of the stream data in a social network

Stream Data. that is updated or arrives frequently, is shown in the bottom layer. It contains various sources of streaming data:

– *GPS stream* (S_{gps}): inspired by the use case in Live Social Semantics [1], we assume that each user has a GPS tracking device to send updated information about her current location to the SN frequently. This information contains latitude/longitude of the user's position, e.g., the location of an event where user is attending, and the sending time.
– *Posts and comments stream* (S_{pc}): there is a huge stream of posts and comments in the SN as users start or join discussions. Similar to the availability of the "wall" for each Facebook user or the "Tweet timeline" for each Twitter, every user in our generated SN has her own forum for writing posts. People who subscribe to this forum (or "follow" the forum moderator as in Twitter) can read and reply to the posts and comments created in the forum. Each forum is used as a channel for the posting stream of a user. In this stream, we are particularly interested in the stream of "likes" (i.e., people who show their interest in a post), denoted by S_{pclike}, the stream of tags (i.e., set of words representing the content of the discussion), denoted by S_{tags}.
– *Photos stream* (S_{fo}): Uploaded photos and their associated attributes provide interesting information for discovering user habits, friend relationships, etc. In this stream, we focus on exploiting useful information from the stream of user tagged in a photo, S_{fotags}, and the stream of likes per photo, denoted by S_{folike}.

Static Data. U_{data}, that is not frequently changed or updated, is shown in the upper layer. It contains user profile information (name, date of birth, relationship status, etc.), the relationships between users and the channel where they write posts, comments.

2.2 Data Generator

To the best of our knowledge, there exists no stream data generator that can realistically simulate the stream data in SNs. We propose a novel data generator for LSD, called *Stream Social network data Generator* (S2Gen). It generates data according to the schema in Section 2.1 in consideration of the continuous query semantics [3,7,5,9,2,14] and various realistic data distributions such as the skewed distributions of posts/comments. As window operators are primitive operators in a continuous query, the correlations of simulated data have to affect on data windows over streams. To meet this requirement, S2Gen uses the "window sliding" approach from the structure-correlated social graph generator S3G2 [16]. As such, to generate the stream data, it slides a window of users along all users in the social graph and creates social activities for each user (writing a post/comment, uploading photos, sending GPS tracking information). For creating a particular stream data, e.g., S_{pc}, S2Gen extracts all the posts/comments created for all the users, then sorts them according to their timestamps, and finally serializes these data to a file. A stream player is created in order to push the stream data from this file into a streaming engine. Similarly, S_{fo}, S_{pclike}, S_{folike}, and S_{gps} are created.

For static data, S2Gen generates the user profiles and the friendship information of all the users in order to form the static data, i.e., U_{data}. The details of this step and how to simulate the data correlations in the static data are the same as in S3G2. Note that all the generated stream data is correlated with non-stream data, e.g., user tags in the photo stream are correlated with friendship information. Various realistic situations are also simulated while generating stream data, e.g., for GPS stream data, around a specific time, the latitude and longtitude sent by those people attending the same event are close to each other and that of the event's location.

For flexibility, S2Gen offers a range of parameters. Some main parameters used in following experiments are:

– *Generating period*: the period in which the social activities are generated, e.g., 10 days, one month, etc. By varying this parameter, one can create streams with different sizes for testing scalability.
– *Maximum number of posts/comments/photos for each user per week*: each of these parameters can be adjusted in order to change the amount of data that arrives in a window of time. It thus can increase/decrease the input rate (e.g., number of triples/seconds) as the stream player pushes the data according to a window of time. Besides, it also varies the total amount of generated streaming data for a fixed generating period.
– *Correlation probabilities*: there are various parameters for the data correlations between graph data and the graph structure, e.g., the probability that users will be connected if they are living in the same area. They can be customized to specify how data is skewed according to each data attribute. The tested systems need to recognize these correlation properties in order to optimize their query plan.

3 Experimentation

3.1 Evaluation Design and Strategies

The setup to evaluate an engine E with a stream query Q is as follows. Suppose that Q requires as input a non-empty set of finite streams $\mathcal{S}_Q = \{S_1, \ldots, S_m\}$, $(m \geq 1)$, and possibly static data. Let $R = \{r_1, \ldots, r_m\}$ be a set of rates (elements/sec) such that each S_i is fed into E at rate r_i. We expect as the output a sequence of elements $O(E, Q, R) = o_1, \ldots, o_n$, abbreviated by O_E when Q and R are clear from the context.

Our evaluation design is general enough to capture different stream engines. For examples, E can be CQELS, CSPARQL, or EP-SPARQL[1] for testing LSD engines, where the static data is a set of RDF graphs, each S_i is a Link Data stream, i.e., a sequence of RDF triples, and the output is a sequence of triples or SPARQL results.

Note that EP-SPARQL is just a wrapper of the Prolog-based stream engine ETALIS. When testing EP-SPARQL, we observed that it suffered considerably heavy loads from parsing big RDF graphs. Moreover, it does not support multi-threading to easily control the rates of input streams. Recently, JTALIS has been developed as a Java wrapper for ETALIS. It does not exhibit the above parsing problems, as it works with facts and Prolog rules. Furthermore, using Java makes it very convenient to control input rates via multi-threading. Hence we decided to evaluate JTALIS. Here the static data is a set of ground facts, each S_i as well as the output is a sequence of ground facts,[2] and queries can be formalized as sets of Prolog rules. We thus compare CQELS, CSPARQL, and JTALIS with the following plan:[3]

– *Functionality tests*: we introduce a set of stream queries with increasing complexities and check which can be successfully processed by each engine;
– *Correctness tests*: for the executable queries, we run the engines under the same conditions to see whether they agree on the outputs. In case of big disagreement, we introduce notions of *mismatch* to measure output comparability. When the engines are incomparable, we thoroughly explain the reasons to our best understanding of them.
– *Performance tests*: for queries and settings where comparability applies, we execute the engines under their maximum input rates and see how well they can handle the limit. Performance is also tested wrt. different aspects scalability, namely the volume of static data size and the number of simultaneous queries.

3.2 Functionality Tests

We use the SN scenario generated by the tool in Section 2.2. Here, the static data is U_{data}, and the incoming streams are S_{gps}, S_{pc}, S_{pclike}, S_{fo}, S_{folike} (cf. Section 2.1). The data generator considers many parameters to produce plausible input data, but for our experimental purpose, we are interested in the size (number of triples/facts) of the static data and input streams, i.e., $|U_{data}|$, $|S_{pc}|$, etc.

[1] SPARQL$_{stream}$ implementation has not supported native RDF data (confirmed by the main developer).
[2] We normalize the outputs to compare sets of facts and SPARQL result sets.
[3] All experiments are reproducible and recorded, details are available at http://code.google.com/p/lsbench/

Table 1. Queries classification

	Patterns covered							S	N_P	N_S	Engines		
	F	J	A	E	N	U	T				CQ	CS	JT
Q_1	√								1	1	√	√	√
Q_2		√						√	2	1	√	√	√
Q_3		√						√	3	1	√	√	√
Q_4	√	√							4	1	√	√	√
Q_5		√						√	3	2	√	√	∅
Q_6		√						√	4	2	√	√	∅
Q_7	√	√						√	7	2	√	Ⓢ	∅
Q_8		√			√				3	2	×	Ⓢ	∅
Q_9	√	√				√		√	8	4	√	E	∅
Q_{10}			√						1	1	√	√	√
Q_{11}		√	√	√					2	2	×	√	×
Q_{12}			√				√		1	1	×	√	×

F: filter **J**: join **E**: nested query **N**: negation **T**: top k **U**: union **A**: aggregation **S**: uses static data
N_P: number of patterns, N_S: number of streams, Ⓢ: syntax error, E: error, ∅: return no answer, ×: not supported
CQ: CQELS, CS: C-SPARQL, JT: JTALIS

The engines are run on a Debian squeeze amd64 2x E5606 Intel Quad-Core Xeon 2.13GHz with 16GB RAM, running OpenJDK Runtime Environment (IcedTea6 1.8.10) OpenJDK 64-Bit Server VM, and SWI-Prolog 5.10.1.

To evaluate the engines by query expressiveness, we propose 12 queries which cover different desired features and aspects of stream queries. Table 1 reports the query patterns in detail and our observation of which queries can be successfully executed by which engines. It shows that a number of desired features are not yet satisfactorily covered. CQELS supports neither negation nor nested aggregation. CSPARQL reports syntax errors on Q_7(complicated numeric filter), Q_8(negation), and encounters runtime error on Q_9(most complicated query). Regarding JTALIS, patterns in Q_5-Q_9 are theoretically supported, but the run produces no output. Also, there is no support for explicit representations of aggregations that works with timing windows; with kind help from the JTALIS team, we encoded the simple counting aggregation in Q_{10} by recursive rules, but left out the more complicated aggregations in Q_{11}, Q_{12}.

3.3 Correctness Tests

Based on the supported functionalities from all engines, we first evaluate them on Q_1–Q_4 and Q_{10} with a static data of 1000 user profiles, and a stream of posts during one month: $|U_{data}| = 219825$ and $|S_{pc}| = 102955$, at two rates of 100 and 1000 input elements per second,[4] which are considered slow and fast, respectively. Then, we check the agreement on the results between the engines by simply comparing whether they return the same number of output elements. Here we adopt the *soundness* assumption, i.e., every output produced by the engines are correct. The results are reported in the "Output size" columns of Table 2. It turned out that:

(i) CSPARQL disagrees with the rest wrt. the total number of output triples. It returns duplicates for simple queries while for complicated ones, it misses certain outputs;
(ii) CQELS and JTALIS agree on the total number of output triples on most of the case, except for Q_4.

[4] Each element is a triple (resp., fact) for CQELS, CSPARQL (resp. JTALIS).

Explaining these mismatches, regarding (i), CSPARQL follows the periodical execution strategy, i.e., the query processor is triggered periodically no matter how fast or slow the inputs are streamed into the system. When the execution is faster than the update rate, the engine will re-operate on the same input before the next update, hence outputs replicated results between two updates. In contrast, when the execution is slower than the update rate, certain inputs are ignored between two consecutive executions. Thus, for complicated queries, one expects that CSPARQL misses certain outputs.

CQELS and JTALIS, on the other hand, follow the eager execution strategy, and trigger the computation incrementally as soon as new input arrives; or, in case data arrives during computation, they queue the input and consume it once the engine finishes with the previous input. Therefore, eager ones do not produce overlapping outputs.

For the differences between CQELS and JTALIS that lead to observation (ii), the execution speed and windows play an important role. For simple queries Q_1–Q_3,[5] the two engines perform on more or less the same speed, hence the results are identical. For the more complex query Q_4, the inputs contained in the time-based windows determine the outputs of each execution. The slower the execution rate, the less input in the windows, as more of the input is already expired when the new input is processed. Consequently, output for Q_4 produced by JTALIS (which is slower) is smaller than that produced by CQELS. To circumvent this problem, one can use triple-based windows. Unfortunately, this type of windows is not yet supported by JTALIS.

The total number of outputs is not ideal to cross-compare the engines. We next propose a more fine-grained criterion by tolerating duplication and checking for mismatch.

Comparing by Mismatch. We now propose a function to compute the mismatch $mm(E_1, E_2, Q, R)$ between two output sequences $O_{E_1} = a_1, \ldots, a_{n_1}$ and $O_{E_2} = b_1, \ldots, b_{n_2}$ produced by engines E_1, E_2 running query Q at rates R, i.e., we would like to see how much of the output from E_1 is not produced by E_2.

An *output-partition* of O_{E_1} is defined as a sequence of blocks $A_1, \ldots, A_m$ where each A_i is a subsequence $a_{i1}, \ldots, a_{i\ell_i}$ of O_{E_1} such that the sequence $a_{11}, \ldots, a_{1\ell_1}, a_{21}, \ldots, a_{2\ell_2}, \ldots, a_{m1}, \ldots, a_{m\ell_m}$ is exactly O_{E_1}. There are multiple output-partitions of O_{E_1}, but we are only interested in two special ones:

- the *periodical output-partition*: each A_i is corresponding to the result of an execution of E_1, when E_1 is a periodical engine;
- the *eager output-partition*: $A_i = a_i$, i.e., a sequence of a single output element.

A *slice* of O_{E_2} from j, $1 \leq j \leq n$ is the sequence $O_{E_2}\lfloor j \rfloor = b_j, \ldots, b_{n_2}$. A block A_i is *covered* by a slice $O_{E_2}\lfloor j \rfloor$ iff for every $a_{ik} \in A_i$, there exists $b_t \in O_{E_2}\lfloor j \rfloor$ such that $a_{ik} = b_t$. In case of non-coverage, the *maximal remainder* of A_i wrt. $O_{E_2}\lfloor j \rfloor$ is defined by $P_i^j = A_i \Cap O_{E_2}\lfloor j \rfloor = a_{ir_1}, \ldots, a_{ir_{k_i}}$ such that for $1 \leq s \leq k_i$, $a_{ir_s} \in A_i$ and there exists $b_t \in O_{E_2}\lfloor j \rfloor$ such that $a_{ir_s} = b_t$. Intuitively, P_i^j is constructed by keeping elements in A_i that also appear in $O_{E_2}\lfloor j \rfloor$; in other words, P_i^j is the maximal sub-block of A_i which is covered by $O_{E_2}\lfloor j \rfloor$.

[5] Reason for identical total number of output tuples for Q_{10} will be made clear in explaining the mismatch.

Table 2. Output Mismatch, $|U_{data}| = 219825$, $|S_{pc}| = 102955$

	Rate: 100 (input elements/sec)									Rate: 1000 (input elements/sec)								
	Output size			Mismatch (%)						Output size			Mismatch (%)					
Q	CQ	CS	JT	CQ—CS		CQ—JT		CS—JT		CQ	CS	JT	CQ—CS		CQ—JT		CS—JT	
1	68	604	68	1.47	0.00	0.00	0.00	0.00	1.47	68	662	68	1.47	0.00	0.00	0.00	0.00	1.47
2	68	124	68	1.47	0.00	0.00	0.00	0.00	1.47	68	123	68	1.47	0.00	0.00	0.00	0.00	1.47
3	533	1065	533	0.00	0.00	0.00	0.00	0.00	0.00	533	1065	533	0.00	0.00	0.00	0.00	0.00	0.00
4	11948	125910	1442	1.69	1.10	87.93	0.00	78.91	0.07	11945	127026	4462	1.54	1.12	62.65	0.00	52.79	0.02
10	28021	205986	28021	14.96	0.04	87.66	0.00	44.67	0.00	28021	209916	28021	14.70	0.04	86.30	0.00	43.25	0.00

Table 3. (Comparable) Maximum Execution Throughput

	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_{10}
CQELS	24122	8462	9828	1304	7459	3491	2326
C-SPARQL	10	1.68	1.63	10	1.72	1.71	10
JTALIS	3790	3857	1062	99	—	—	87

If P_i^j is an empty sequence, we define $match(P_i^j) = j$. Otherwise, for each element $a_{ir_s} \in P_i^j$, let $match(a_{ir_s})$ be the smallest index t, where $j \leq t \leq n_2$, such that $a_{ir_s} = b_t$, and let $match(P_i^j) = \min\{match(a_{ir_s}) \mid a_{ir_s} \in P_i^j\}$.

We now can define the *maximal remainder sequence* of O_{E_1} that is covered by O_{E_2} as $T_1, \ldots, T_m$, where $T_1 = P_1^1 = A_1 \Cap O_{E_2}\lfloor 1 \rfloor$ and $T_i = P_i^{match(T_{i-1})} = A_i \Cap O_{E_2}\lfloor match(T_{i-1}) \rfloor$ for $1 < i \leq m$. Intuitively, we progressively compute the maximal remainder of each block, starting with the slice $O_{E_2}\lfloor 1 \rfloor$ from the beginning of O_{E_2}. When finishing with one block, we move on to the next one and shift the slice to the minimal match of the last block.

The mismatch is $mm(E_1, E_2, Q, R) = \dfrac{\Sigma_{i=1}^{m}(|A_i| - |T_i|)}{\Sigma_{i=1}^{m}|A_i|} \times 100\%$

Table 2 reports the mismatches between the engines on Q_1-Q_4 and Q_{10}. For a column labeled with E_1—E_2 where $E_1 \neq E_2 \in \{CQ, CS, JT\}$, the left sub-column presents $mm(E_1, E_2, Q, R)$ and the right one shows $mm(E_2, E_1, Q, R)$, respectively. For simple queries Q_1-Q_3, the mismatches are very small, meaning that CSPARQL computes many duplicates but almost of all its output is covered by CQELS and JTALIS.

When the query complexity increases in Q_4 and Q_{10}, CSPARQL misses more answers of CQELS as $mm(CQ, CS, Q_4, 100) = 1.69\%$ and $mm(CQ, CS, Q_{10}, 100) = 14.96\%$. On the other hand, JTALIS produces far less output than the other two for Q_4, due to the reasons in explanation (ii) above. The big mismatches here (from 52.79% to 87.93%) result from the different execution speeds of JTALIS and the other engines.

Interestingly, CQELS and JTALIS output the same number of tuples for Q_{10}, but the contents are very different: $mm(CQ, JT, Q_{10}, 100) = 87.66\%$ and $mm(CQ, JT, Q_{10}, 1000) = 86.30\%$. This is because Q_{10} is a simple aggregation, which gives one answer for every input; hence we have the same number of output tuples between CQELS and JTALIS, which follow the same execution strategy (eager). However, again the difference in execution speed causes the mismatch in the output contents. For all queries

that JTALIS can run, $mm(JT, CQ, Q, R) = 0$ where $Q \in \{Q_1, \ldots, Q_4, Q_{10}\}$ and $R \in \{100, 1000\}$, meaning that the output of JTALIS is covered by the one of CQELS.

In concluding this section, we formalize the notion of *comparability by mismatch* as follows. Given a set of engines $\mathcal{E}$, a query Q, and rates R, the engines in $\mathcal{E}$ are said to be comparable with a prespecified mismatch tolerance ϵ, denoted by $comp(\mathcal{E}, Q, R, \epsilon)$, iff for every $E_1, E_2 \in \mathcal{E}$, it holds that $mm(E_1, E_2, Q, R) \leq \epsilon$.

3.4 Performance Tests

This section defines execution throughput, and then reports on this measure when the engines run on a basic setting as well as when different input aspects scale.

Execution Throughput. Besides comparability, one also would like to see how fast the engines are in general. We therefore conduct experiments to compare the engines wrt. performance. The most intuitive measure to show the performance of a stream processing engine is *"throughput,"* which is normally defined as the average number of input elements that a system can process in a unit of time. However, as mentioned above, systems like C-SPARQL using the periodical execution mechanism can skip data coming in at high stream rate. Therefore, a maximum streaming rate is not appropriate to measure "throughput." Moreover, as shown in previous sections, there are several reasons that make the output from such engines incomparable. We thus propose *"comparable maximum execution throughput"* as a measure for the performance test.

According to this measure, we first need to make sure that the engines produce *comparable* outputs at some (slow) rate, e.g., $comp(\{CQELS, CSPARQL, JTALIS\}, Q_2, 100, 0.147)$ holds. When this is settled, we modify all periodical engines such that the input and execution rates are synchronized. Interestingly, confirmed by our tests on all executable queries on C-SPARQL, there are only marginal changes in execution rates when varying input rates. In this particular evaluation, thanks to the APIs from CSPARQL that notify when an execution finishes, we can achieve this modification by immediately streaming new inputs after receiving such notifications. Note that CSPARQL schedules the next execution right after the current one unless explicitly specified.

Then, given the size N_E of the input streamed into an engine $E \in \mathcal{E}$ and the total running time T_E that E needs to be processed by a query Q, assume that E *immediately* reads the new input once finishing with the previous one, the number of executions is N_E as E does one computation per input. When $comp(\mathcal{E}, Q, R, \epsilon)$ holds, the *comparable maximum execution throughput of* E is defined as $cmet(E, \mathcal{E}, Q, R, \epsilon) = N_E/T_E$.

Table 3 reports the value of $cmet$ for Q_1–Q_6 and Q_{10}. For Q_i, the comparable test is fulfilled by $comp(\{CQELS, CSPARQL, JTALIS\}, Q_i, 100, \epsilon_i)$ where $\epsilon_i = \max\{mm(E_1, E_2, Q_i, 100) \mid E_1 \neq E_2 \in \{CQ, CS, JT\}\}$ with the mismatch values taken from Table 2. It shows that CQELS and JTALIS have higher $cmet$ than CSPARQL by some orders of magnitude.

Scalability Tests. Next, we investigate how the systems behave wrt. to two aspects of scalability, namely (i) static data size, and (ii) the number of queries. Regarding the former, we gradually increment the static data by generating test cases with increasing

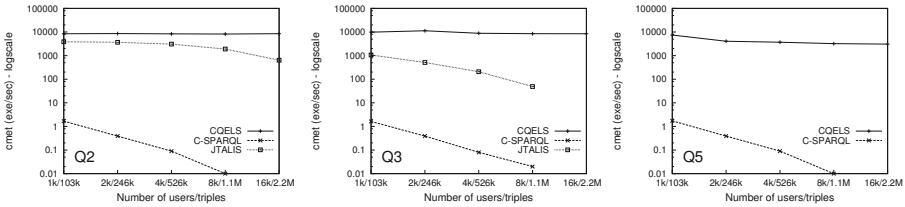


Fig. 2. Comparable max. execution throughput for varying size of static data

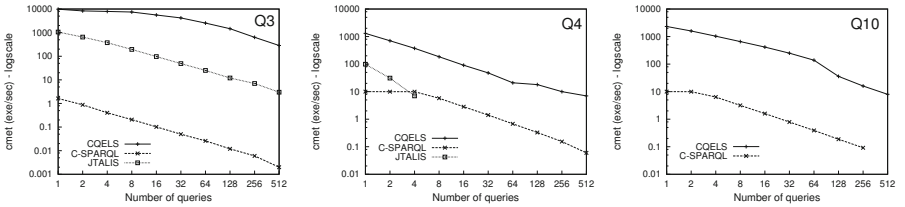


Fig. 3. Comparable max. execution throughput running multiple query instances

number of users. For the latter, we can use the same query template but augmenting the number of registered query instances. From queries generated from the same pattern, we expect better scalability effects on the engines that support multiple query optimization [12]. Figures 2 and 3 report *cmet* when running the engines on those settings. As seen in Figure 2, C-SPARQL's *cmet* dramatically decreases when the size of static data increases; JTALIS performs better than C-SPARQL. On the other hand, CQELS's performance only slightly degrades for complicated queries like Q5. In Figure 3, CQELS still outperforms to C-SPARQL and JTALIS but the speed of throughput deterioration is still linear like those of the counterparts. Here, the performance gains mainly come from the performance of single queries.

4 Findings and Discussion

As most of the considered systems are scientific prototypes and work in progress, unsurprisingly they do not support all features and query patterns. Moreover, the comparability tests in the Section 3.2 clearly exhibit that even for queries with almost identical meaning resp. semantics, the outputs sometimes are significantly different due to the differences in implementation. The differences in outputs are also contributed by intrinsic technical issues of handling streaming data , e.g., time management, potentially fluctuate execution environment [12,15]. Therefore, any reasonable cross-system comparison must take comparability criteria akin to those we considered into account. In addition, comparability tests with tolerant criteria are useful for testing the correctness of a query engine if there is an equivalent baseline system, i.e, given that the baseline system has standard features that a new system should conform to.

The performance and scalability tests show that throughout C-SPARQL yields considerably lower throughput compared to JTALIS and CQELS. This provides further

evidence for the argument that the recurrent execution may waste significant computing resources. Recurrent results can be useful for some applications, such as answering "return the last 10 tweets of someone." However, re-execution is unnecessary unless there are updates between consecutive executions, and thus "incremental computing" is mainly recommended in the literature [12,15]. There, outputs are incrementally computed as a stream, and recurrences can be extracted by a sliding window. In this fashion, the output of eager incremental computing as by CQELS and JTALIS can be used to answer recurrent queries. As the incremental execution by CQELS and JTALIS outperforms the periodic computation of C-SPARQL by an order of magnitude, the conjecture is that they would also answer recurrent queries as described above more efficiently.

Comparing CQELS and JTALIS, the former performs better mainly because it uses a native and adaptive approach (cf. [14]). Note that the performance of C-SPARQL and JTALIS heavily depends on underlying systems, viz. a relational stream processing engine and Prolog engine, respectively. Their performance can thus benefit from optimizations of the (or use of the) underlying engines. Similarly, CQELS would profit from more sophisticated, optimized algorithms compared to the current one [14].

The scalability tests show that all engines have not yet been optimized for scalable processing. Apparently, C-SPARQL already exhibits performance problems on simple queries involving static data beyond 1 million triples. JTALIS, while capable of handling these settings, struggles on more complicated queries with static datasets larger than 1 million triples. CQELS is the only system that precomputes and indexes intermediate results from sub-queries over the static data [14], and therefore scales well with increasing static data size. Clearly, this technique is not restricted to CQELS and applicable to C-SPARQL, JTALIS, and any other LSD engine to improve on scalability wrt. static data. However, CQELS does also not scale well when increasing the number of queries (sharing similar patterns and data windows). The same holds for C-SPARQL and JTALIS, which clearly testifies that none of the systems employ multiple query optimization techniques [12,15] , e.g., to avoid redundant computations among queries sharing partial computing blocks and memory.

5 Related Work

We already mentioned characteristics of LSD engines that are critically relevant to this paper in Section 1. In-depth theoretical/technical foundations of LSD processing can be found in [15]. We next briefly review existing related benchmarking systems.

Linear Road Benchmark [4] is the only published benchmarking system for relational data stream processing engines so far. However, it focuses on ad-hoc scenarios to evaluate outputs. On the other hand, our work treats the processing engines and queries as black boxes. Furthermore, while Linear Road Benchmark only focuses on a single quantitative metric "scale factor," our evaluation studies several aspects of the systems as shown above. Concerning stream data generator, NEXMark[6] can be used to test relational data stream systems. However, not only is its data schema quite simple but the simulated data is unrealistically random.

[6] http://datalab.cs.pdx.edu/niagara/NEXMark/

Triple Storage Benchmarks. With increasing availability of RDF triple stores,[7] a number of RDF benchmarks have been developed to evaluate the performance of these systems.[8] However, most of the popular benchmarks such as BSBM [6], LUBM [13] and SP2Bench [17] are either limited in representing real datasets or mostly relational-like [11]. On top of that, none of them focus on time-varying data and continuous query. By extending recently developed Social Network Interlligence Benchmark (SIB)[9], our evaluation framework is natively designed not only to support continuous queries over LSD but also to simulate realistic graph-based stream data.

6 Conclusion

In this work, we propose the first—to the best of our knowledge—customizable framework with a toolset for cross-evaluating Linked Stream Data (LSD) engines. Along with the framework we developed a methodology and measures to deal with conceptual and technical differences of LSD engine implementations. Powered by this environment, another main contribution in this paper is a systematic and extensive experimental analysis, revealing interesting functional facts and quantitative results for state-of-the-art LSD engines (see Section 3). Our findings from this analysis identify performance shortcomings of these engines that need to be addressed in further developments of these, but also by future LSD processing engines.

It is often the case that linked stream data engines are rated negatively when compared with relational stream engines. Therefore, for further work, we will provide a baseline test set [4] with corresponding relational data schema [6] to compare LSD engines with relational ones. On top that, we plan to extend the evaluation framework in order to support LSD engines that enable continuous approximation queries and feature load shedding [12]. Also, distributed LSD engines are expected to be in place soon, and evaluating them is another challenging and interesting topic of research to pursue.

References

1. Alani, H., Szomszor, M., Cattuto, C., Van den Broeck, W., Correndo, G., Barrat, A.: Live Social Semantics. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 698–714. Springer, Heidelberg (2009)
2. Anicic, D., Fodor, P., Rudolph, S., Stojanovic, N.: EP-SPARQL: a unified language for event processing and stream reasoning. In: WWW, pp. 635–644 (2011)
3. Arasu, A., Babu, S., Widom, J.: The CQL continuous query language: semantic foundations and query execution. The VLDB Journal 15(2), 121–142 (2006)
4. Arasu, A., Cherniack, M., Galvez, E., Maier, D., Maskey, A.S., Ryvkina, E., Stonebraker, M., Tibbetts, R.: Linear road: a stream data management benchmark. In: VLDB, pp. 480–491 (2004)

[7] http://www.w3.org/wiki/SemanticWebTools#RDF_Triple_Store_Systems

[8] http://www.w3.org/wiki/RdfStoreBenchmarking

[9] http://www.w3.org/wiki/Social_Network_Intelligence_BenchMark

5. Barbieri, D.F., Braga, D., Ceri, S., Grossniklaus, M.: An execution environment for C-SPARQL queries. In: EDBT, pp. 441–452. ACM (2010)
6. Bizer, C., Schultz, A.: The Berlin SPARQL benchmark. Int. J. Semantic Web Inf. Syst. 5(2), 1–24 (2009)
7. Bolles, A., Grawunder, M., Jacobi, J.: Streaming SPARQL - Extending SPARQL to Process Data Streams. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 448–462. Springer, Heidelberg (2008)
8. Bouillet, E., Feblowitz, M., Liu, Z., Ranganathan, A., Riabov, A., Ye, F.: A Semantics-Based Middleware for Utilizing Heterogeneous Sensor Networks. In: Aspnes, J., Scheideler, C., Arora, A., Madden, S. (eds.) DCOSS 2007. LNCS, vol. 4549, pp. 174–188. Springer, Heidelberg (2007)
9. Calbimonte, J.-P., Corcho, O., Gray, A.J.G.: Enabling Ontology-Based Access to Streaming Data Sources. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 96–111. Springer, Heidelberg (2010)
10. Celino, I., Dell'Aglio, D., Valle, E.D., Balduini, M., Huang, Y., Lee, T., Kim, S.-H., Tresp, V.: Bottari: Location based social media analysis with semantic web. In: ISWC (2011)
11. Duan, S., Kementsietsidis, A., Srinivas, K., Udrea, O.: Apples and oranges: a comparison of rdf benchmarks and real rdf datasets. In: SIGMOD, pp. 145–156 (2011)
12. Golab, L., Özsu, M.T.: Data stream management. Synthesis Lectures on Data Management, pp. 1–73 (2010)
13. Guo, Y., Pan, Z., Heflin, J.: LUBM: A benchmark for owl knowledge base systems. Web Semantics: Science, Services and Agents on the World Wide Web 3(2-3), 158–182 (2005)
14. Le-Phuoc, D., Dao-Tran, M., Parreira, J.X., Hauswirth, M.: A Native and Adaptive Approach for Unified Processing of Linked Streams and Linked Data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 370–388. Springer, Heidelberg (2011)
15. Le-Phuoc, D., Xavier Parreira, J., Hauswirth, M.: Linked Stream Data Processing. In: Eiter, T., Krennwallner, T. (eds.) Reasoning Web 2012. LNCS, vol. 7487, pp. 245–289. Springer, Heidelberg (2012)
16. Minh Duc, P., Boncz, P.A., Erling, O.: S3G2: A Scalable Structure-Correlated Social Graph Generator. In: TPCTC, Turkey, (2012)
17. Schmidt, M., Hornung, T., Lausen, G., Pinkel, C.: Sp^2bench: A SPARQL performance benchmark. In: ICDE, pp. 222–233 (2009)
18. Sequeda, J.F., Corcho, O.: Linked stream data: A position paper. In: SSN (2009)
19. Sheth, A., Henson, C., Sahoo, S.S.: Semantic sensor web. IEEE Internet Computing (2008)

Benchmarking Federated SPARQL Query Engines: Are Existing Testbeds Enough?

Gabriela Montoya[1], Maria-Esther Vidal[1], Oscar Corcho[2], Edna Ruckhaus[1], and Carlos Buil-Aranda[3]

[1] Universidad Simón Bolívar, Venezuela
{gmontoya,mvidal,ruckhaus}@ldc.usb.ve
[2] Ontology Engineering Group, Universidad Politécnica de Madrid, Spain
ocorcho@fi.upm.es
[3] Department of Computer Science, Pontificia Universidad Católica, Chile
cbuil@ing.puc.cl

Abstract. Testbeds proposed so far to evaluate, compare, and eventually improve SPARQL query federation systems have still some limitations. Some variables and configurations that may have an impact on the behavior of these systems (e.g., network latency, data partitioning and query properties) are not sufficiently defined; this affects the results and repeatability of independent evaluation studies, and hence the insights that can be obtained from them. In this paper we evaluate FedBench, the most comprehensive testbed up to now, and empirically probe the need of considering additional dimensions and variables. The evaluation has been conducted on three SPARQL query federation systems, and the analysis of these results has allowed to uncover properties of these systems that would normally be hidden with the original testbeds.

1 Introduction

The number of RDF datasets made publicly available through SPARQL endpoints has exploded in recent years. This fact, together with the potential added value that can be obtained from the combination of such distributed data sources, has motivated the development of systems that allow executing queries over federated SPARQL endpoints (e.g., SPARQL-DQP [2], Jena's ARQ[1], RDF::Query[2], ANAPSID [1], FedX [10], ADERIS [3]). Some systems use SPARQL 1.0 or ad-hoc extensions, while others rely on the query federation extensions that are being proposed as part of the upcoming SPARQL 1.1 specification [6].

In parallel to the development of federated SPARQL query evaluation systems, several testbeds have been created (e.g., as described in [2,7,8]), which complement those already used for single-endpoint query evaluation. The role of these testbeds is to allow evaluating and comparing the main characteristics of these systems, so as to provide enough information to improve them. Among

[1] http://jena.apache.org/
[2] http://search.cpan.org/~gwilliams/RDF-Query/

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 313–324, 2012.

the features evaluated by these testbeds we can cite: *i*) functional requirements supported, *ii*) efficiency of the implementations with different configurations of datasets and with different types of queries, or *iii*) resilience to changes in the configurations of these systems and the underlying datasets. The most recent and complete testbed is FedBench [8], which proposes a variety of queries in different domains and with different characteristics, including star-shaped, chain-like and hybrid queries, and complex query forms using an adaptation of SP2Bench [9]. These testbeds are steps forward towards establishing a continuous benchmarking process of federated SPARQL query engines. However, they are still far from effectively supporting such benchmarking objectives. In fact, they do not specify completely or even consider some of the dependent and independent variables and configuration setups that characterize the types of problems to be tackled in federated SPARQL query processing, and that clearly affect the performance and quality of different solutions. This may lead to incorrect characterizations when these testbeds are used to select the most appropriate systems in a given scenario, or to decide the next steps in their development.

For example, testbeds like the one in [2] have limitations. First, queries are executed directly on live SPARQL endpoints; this means that experiments are not reproducible, as the load of endpoints and network latency varies over time. Second, queries were constructed for the data available in the selected endpoints at the time of generating the testbed, but the structure of these underlying RDF data sources changes, and may result in queries that are returning different answers or that do not return any answer at all. In cases like FedBench [8], the level of reproducibility is improved by using datasets that can be handled locally. However, as shown in Section 2, there are variables that are not yet considered in this benchmark (e.g., network latency, dataset configurations) and that are important in order to obtain more accurate and informative results.

The objective of this paper is to describe first the characteristics exhibited by these testbeds (mainly focusing on FedBench) and reflect on their current limitations. Additional variables and configuration setups (e.g., new queries, new configurations of network latency details, new dataset distribution parameters) are proposed in order to provide more accurate and well-informed overviews of the current status of each of the evaluated systems, so that the experiments to be executed can offer more accurate information about the behavior of the evaluated systems, and hence they can be used in continuous improvement processes. Finally, we describe briefly the results of our evaluation of this extended testbed using three different federated query engines: ARQ, ANAPSID, and FedX.

2 Some Limitations of Existing Testbeds

There is no unique "one-size-fits-all" testbed to measure every characteristic needed by an application that requires some form of federated query processing [8]. However, regardless, existing testbeds can still be improved so that they can fulfill their role in continuous benchmarking processes.

We will first illustrate why we need to improve existing testbeds, particularly FedBench, by describing a scenario where the use of the testbed in its current

form may lead to wrong decisions. We have executed the FedBench testbed with three systems (ANAPSID, ARQ, and FedX) on the three sets of queries proposed (Life Science, Cross Domain, and Linked Data) [4]. We have used different simulated configurations for network latencies and different data distributions of the datasets used in the experiments. As a result, we observe interesting results that suggest the need for improvements. For instance, for the Cross-Domain query `CD1`, all systems behave well in a perfect network (as shown in Table 1). However, their behavior changes dramatically when network latencies are considered. For instance, ARQ is not able to handle this query for medium-fast and fast networks, given the timeout considered; the time needed to execute the query in the case of FedX grows from 0.72 secs. (perfect network) to 2.23 secs. (fast network) and 16.93 secs. (medium-fast network); and for ANAPSID the results are similar for perfect and fast networks, and grows slower in medium-fast networks.

Table 1. Evaluation of FedBench query CD1-Number of results and execution time (secs.) under different network latency conditions. Timeout was set up to 30 minutes. Perfect Network (No Delays); Fast Network (Delays follow Gamma distribution ($\alpha = 1$, $\beta = 0.3$); Medium-Fast Network (Delays follow Gamma distribution ($\alpha = 3$, $\beta = 1.0$)).

	Number of results			Execution time (secs.) (first tuple)			Execution time (secs.) (all tuples)		
Query Engine	**Medium**	**Fast**	**Perfect**	**Medium**	**Fast**	**Perfect**	**Medium**	**Fast**	**Perfect**
ANAPSID	61	61	61	0.98	0.17	0.16	0.98	0.17	0.16
FedX	61	61	61	16.93	2.23	0.72	16.93	2.23	0.72
ARQ	–	–	63	–	–	0.98	–	–	0.98

This is also the case for other FedBench queries (e.g., LD10, LD11, LS7, CD2), where different behaviors can be observed depending not only on network latency, but also on additional parameters, e.g., data distribution. What these examples show is that those parameters are also important when considering federated query processing approaches, and should be configured in a testbed, so as to provide sufficient information for decision makers to select the right tool for the type of problem being handled, or for tool developers to understand better the weaknesses of their systems and improve them accordingly, if possible.

Finally, there is also another aspect that is important when considering the quality of existing testbeds, and it is the fact that sometimes there are not sufficient explanations about the purpose of each of the parameters that can be configured. For example, in the case of FedBench there are several parameters that are considered when describing queries, as presented in [8], such as whether the query uses operators like conjunctions, unions, filters or optionals, modifiers like DISTINCT, LIMIT, OFFSET or ORDERBY, and structures like star-shaped queries, chains or hybrid cases. While this is quite a comprehensive set of features to characterize a SPARQL query, there are no clear reasons about why each of the 36 queries from the testbed are included. Only some examples are provided in [8], explaining that LS4 "includes a star-shaped group of triple patterns of drugs which is connected via owl:sameAs link to DBpedia drug entities", or that CD5 is a "chain-like query for finding film entities linked

via owl:sameAs and restricted on genre and director". However, there are no explanations in the paper or in the corresponding benchmark website about the reasons for including each of them. Furthermore, there are parameters that are not adequately represented (e.g., common query operators like optionals and filters do not appear in cross domain or linked data queries), and characteristics that are not sufficiently discussed (e.g., the number of triple patterns in each basic graph pattern appearing in the query, the selectivity of each part of the query, etc.), which makes the testbed not complete enough.

In summary, while we acknowledge the importance of these testbeds in the state of the art of federated query processing evaluation, we can identify some of their shortcomings which we illustrate and describe in different scenarios.

3 Benchmark Design

In this section we describe some of the variables that have an impact on federated SPARQL query engines. There are two groups of variables: independent and dependent. Independent variables are those characteristics that need to be minimally specified in the benchmark in order to ensure that evaluation scenarios are replicable. Independent variables have been grouped into four dimensions: Query, Data, Platform, and Endpoint.

Dependent (or observed) variables are those characteristics that are normally influenced by independent variables, as described in Table 2, and that will be measured during the evaluation:

- *Endpoint Selection Time*. Elapsed time between query submission and the generation of the SPARQL 1.1 federated query annotated with the endpoints where sub-queries will be executed[3].
- *Execution Time*. This variable is in turn comprised of: *i*) *Time for the first tuple* or elapsed time between query submission and first answer, *ii*) *Time distribution* of the reception of query answers, and *iii*) *Total execution time*.
- *Answer Completeness*. Number of answers received in relation to the data available in the selected endpoints.

In the following sections we describe independent variables in more detail.

3.1 Query Dimension

This dimension groups variables that characterize the queries in terms of their structure, evaluation, and query language expressivity. Regarding the structure of the query, we focus on three main aspects: *i*) the query plan shape, *ii*) the number of basic triple patterns in the query, and *iii*) the instantiations of subject, object and/or predicates in the query.

[3] This variable is applicable only in cases where the system handles SPARQL 1.0 queries and no endpoints are specified in the query; hence, these queries have to be translated into SPARQL 1.1 or into an equivalent internal representation.

Table 2. Variables that impact the behavior of SPARQL federated engines

	Independent Variables	Observed Variables		
		Endpoint Selection Time	Execution Time	Answer Completeness
Query	query plan shape	✓	✓	✓
	# basic triple patterns	✓	✓	✓
	# instantiations and their position	✓	✓	
	join selectivity		✓	
	# intermediate results		✓	
	answer size		✓	
	usage of query language expressivity	✓	✓	
	# general predicates	✓	✓	✓
Data	dataset size		✓	
	data frequency distribution		✓	
	type of partitioning	✓	✓	✓
	data endpoint distribution	✓	✓	✓
Platform	cache on/off	✓	✓	
	RAM available	✓	✓	
	# processors	✓	✓	
Endpoint	#endpoints	✓	✓	✓
	endpoint type	✓	✓	
	relation graph/endpoint/instance		✓	✓
	network latency	✓	✓	✓
	initial delay	✓	✓	
	message size		✓	
	transfer distribution	✓	✓	✓
	answer size limit		✓	✓
	timeout		✓	✓

Shape. Query plans may be star-shaped, chain-shaped or a combination of them, as described in [8]. In general, the shape of the input queries and of the query plans generated by the systems has an important impact on the three dependent variables identified in our evaluation (endpoint selection time, if applicable, execution time and answer completeness). The shape of the query plans will be in turn affected by the **number of basic triple patterns** in the query since this number will influence the final query shape. Query evaluation systems can apply different techniques when generating query plans for a specific type of input query, and this will normally yield different selection and execution times, and completeness results. For example, a query plan generator may or may not group together all graph patterns related to one endpoint.

Instantiations and their position in triple patterns. This is related to whether any of the elements of the triple patterns in the query (subject, object or predicate) are already instantiated, i.e., bounded to some URI. Together with **join selectivity**, instantiation has an important impact on the potential number of intermediate results that may be generated throughout query execution. For instance, the absence of instantiations (e.g., presence of variables) in the predicate position of a triple pattern may have an important impact in query execution time, because several endpoints may be able to provide answers for the pattern.

Answer size and **number of intermediate results**. If the number of answers or intermediate results involved in a query execution is large, it may take a long time to transfer them across the network, and hence this may affect the query execution time.

Usage of query language expressivity. The use of specific SPARQL operators may affect the execution time and the completeness of the final result set. For example, the OPTIONAL operator is one of the most complex operators in SPARQL [5] and may add a good number of intermediate results, while the FILTER operator may restrict the intermediate results and answer size.

General predicates (e.g., `rdf:type`, `owl:sameAs`) are commonly used in SPARQL queries. However, as they normally appear in most datasets it is not always clear to which endpoint the corresponding subquery should be submitted, and this may have an impact in both endpoint selection and query execution time.

3.2 Data Dimension

We now describe the independent variables related to the characteristics of the RDF datasets that are being accessed. An RDF dataset can be defined in terms of its **size** and its **structural characteristics** like the number of subjects, predicates and objects, and the *in* and *out* degree of properties. These characteristics impact the number of triples that are transferred, and hence the total execution time. Additionally, they may affect the performance of the individual endpoints.

Partitioning and **data distribution** are two of the most important variables that need to be specified in the context of queries against federations of endpoints. Partitioning refers to the way that the RDF dataset is fragmented. Data distribution is the way partitions are allocated to the different endpoints. Data may be fully centralized, fully distributed, or somewhere in between. A dataset may be fragmented into disjunct partitions; the partitioning may be done horizontally, vertically or a combination of both. Horizontal partitioning fragments triples so that they may contain different properties. Vertical partitioning produces fragments which contain all the triples of at least one of the properties in the dataset. Horizontal partitioning impacts on the completeness of the answer whereas vertical partitioning affects the execution time. Partitions may be replicated in several endpoints, even in all of the endpoints, i.e., fully replicated, so that the availability of the system increases in case of endpoint failure or endpoint delay. Table 3 compares the behavior of ANAPSID and FedX with different configurations. The two engines behave similarly when there is one dataset per endpoint and in horizontal partitioning without replication. For vertical partitioning without replication, one engine is superior to the other. When partitioning with replication, one engine outperforms the other in vertical partitioning, and the inverse behavior occurs with horizontal partitioning.

Table 4 shows another example of the effect of data distribution on the query execution time, again for ANAPSID and FedX. We can observe that when there are multiple endpoints, results are similar, while with a network with no delay (perfect network) and all datasets in a single endpoint, one of the engines clearly outperforms the other in one order of magnitude.

Results in Tables 3 and 4 support the claim that data partitioning, data distribution and network delays need to be explicitly configurable in testbeds.

Table 3. Impact of Data Partitioning and Distribution on FedBench query LD10 (Perfect Network). Vertical Partitioning: triples of predicates `skos:subject`, `owl:sameAs`, and `nytimes:latest_use` were stored in fragments. **Vertical Partitioning Without Replication**: three endpoints, each fragment in a different endpoint. **Vertical Partitioning With Replication**: corresponds to use four endpoints and store one of the three fragments in the four endpoints, another fragment in two endpoints, and the last fragment in one endpoint. Horizontal Partitioning: triples of the three predicates were partitioned in two fragments; each fragment has data to produce at least one answer. **Horizontal Partitioning Without Replication** two endpoints; one fragment in a different endpoint. **Horizontal Partitioning With Replicas**: four endpoints; one fragment is replicated in each endpoint, the other fragment in only one endpoint.

Query Engine	Execution time First Tuple (secs.)	Execution time All Tuples (secs.)	Number of Results
One Dataset per Endpoint			
FedX	1.06	1.06	3
ANAPSID	1.08	1.28	3
Vertical Partitioning Without Replication			
FedX	0.69	0.69	3
ANAPSID	3.88	14.25	3
Horizontal Partitioning Without Replication			
FedX	0.72	0.72	3
ANAPSID	0.03	0.03	1
Vertical Partitioning With Replication			
FedX	0.85	0.85	14
ANAPSID	4.06	14.48	3
Horizontal Partitioning With Replication			
FedX	0.91	0.91	25
ANAPSID	0.06	0.06	1

3.3 Platform Dimension

The *Platform* dimension groups variables that are related to the computing infrastructure used in the evaluation. Here we include a minimum set of parameters, related to the system's cache, available RAM memory and number of processors, since this dimension may contain many more parameters that are relevant in this context, and that should anyway be explicitly specified in any evaluation setup when using this testbed.

Turning the **cache management** function in the system together with the **available RAM** may affect greatly the query execution time. The meaning of dropping and warming up cache needs to be clearly specified as well as the

Table 4. Impact of Data Distribution on FedBench query CD1 (Perfect Network). All Datasets in one endpoint versus datasets distributed in different endpoints.

Query Engine	Execution time First Tuple (secs.)	Execution time All Time (secs.)	Number of Results
Single Endpoint-All Databasets			
FedX	0.51	0.51	61
ANAPSID	0.045	0.046	61
Multiple Endpoints			
FedX	0.72	0.72	61
ANAPSID	0.17	0.17	61

number of iterations where an experiment is run in warm cache, and when cache contents are drooped off. In the context of federations of endpoints, information on endpoint capabilities may be stored in cache. The **number of processors** is also a relevant variable in the context of federated queries. If the infrastructure offers several processors, operators may parallelize their execution, and the execution time may be affected positively.

3.4 Endpoint Dimension

This dimension comprises variables that are related to the number and capabilities of the endpoints used in the testbed.

The first variable to be considered is the **number of SPARQL endpoints** where the query will be submitted and the **type of endpoints** that are used for the evaluation. The first variable affects all three observed variables, specially the result completeness because different endpoints may produce different answers. The **relationship between the number of instances, graphs and endpoints** of the systems used during the evaluation is also an important aspect that needs to be specified. Different configurations of these relationships may impact the three dependent variables.

There are several variables that have an important impact on the execution time, such as the **transfer distribution**, which is the time distribution of the transmission of packets by the endpoints, the **network latency**, which defines the delay in sending packets through the network, and the **initial endpoint delay**. An example of the impact of different network delays is illustrated in Table 5. Two queries from the Linked Data collection of FedBench were executed (LD10 and LD11). Note that ANAPSID and FedX behave similarly in LD10 when there is no delay; however, when delays are considered, FedX outperforms ANAPSID. On the other hand, in LD11 ANAPSID outperforms FedX when delays are present. In fact, ANAPSID is able to produce the first tuple after the same amount of time, independently of the delay.

Finally, SPARQL endpoints normally allow configuring a **limit on the answer size** of the queries and a **timeout**, so as to prevent users to query the entire dataset. This may generate empty result sets or incomplete results, particularly when endpoint sub-queries are complex.

4 Some Experimental Results

In this section we illustrate how the testbed extension can be used to better understand the behavior of some of the existing federated query engines. The extended testbed has been executed on three systems (ANAPSID, ARQ and FedX) with several configurations for the independent variables identified in Section 3. The complete result set generated by these executions can be browsed at the DEFENDER portal[4].

[4] http://159.90.11.58/

Table 5. Impact of Network latency on FedBench queries LD10 and LD11. Timeout was set up to 30 minutes and Message Size is 16KB. Perfect Network (No Delays); Fast Network (Delays follow Gamma distribution ($\alpha = 1$, $\beta = 0.3$); Medium-Fast (Delays follow Gamma distribution ($\alpha = 3$, $\beta = 1.0$); Medium-Slow (Delays follow Gamma distribution ($\alpha = 3$, $\beta = 1.5$); Slow (Delays follow Gamma distribution ($\alpha = 5$, $\beta = 2.0$)).

Query Engine	Query	Execution time First Tuple (secs.)	Execution time All Tuples (secs.)	Number of Results
Perfect Network				
ANAPSID	LD10	1.08	1.29	3
	LD11	0.06	0.09	376
FedX	LD10	1.06	1.06	3
	LD11	5.44	5.44	376
Fast Network				
ANAPSID	LD10	18.13	22.89	3
	LD11	0.06	2.80	376
FedX	LD10	3.45	3.45	3
	LD11	14.21	14.22	376
Medium Fast Network				
ANAPSID	LD10	191.78	241.58	3
	LD11	0.07	27.86	376
FedX	LD10	27.27	27.27	3
	LD11	108.93	108.93	376
Medium Slow Network				
ANAPSID	LD10	287.88	362.59	3
	LD11	0.05	41.74	376
FedX	LD10	41.42	41.42	3
	LD11	162.45	162.45	376
Slow Network				
ANAPSID	LD10	653.44	819.72	3
	LD11	0.09	92.52	376
FedX	LD10	87.19	87.19	3
	LD11	347.93	347.93	376

Now we will focus on one of the analyses that a system developer may be interested in, in the context of the continuous benchmarking process that we have referred to in this paper. That is, we are not analyzing the whole set of results obtained from the execution, but only a subset of it. Specifically, let's assume that we are interested in understanding the performance of the three evaluated systems under different data distributions in an ideal scenario, with no or negligible connection latency. Our hypothesis is that existing query engines are sensible to the way data is distributed along different endpoints, even when the network is perfect. Therefore, these results may be useful to validate that hypothesis and to understand whether a set of federated datasets for which we have the corresponding RDF dumps should be better stored in a single endpoint or in different endpoints to offer answers more efficiently. Based on the set of variables identified in our study, the following experimental setup is used:

Datasets and Query Benchmarks. We ran 36 queries against the FedBench dataset collections [8]: DBPedia, NY Times, Geonames, KEGG, ChEBI, Drugbank, Jamendo, LinkedMDB, and SW Dog Food. These queries include 25 FedBench queries and eleven complex queries[5]. The latter are added to

[5] http://www.ldc.usb.ve/~mvidal/FedBench/queries/ComplexQueries

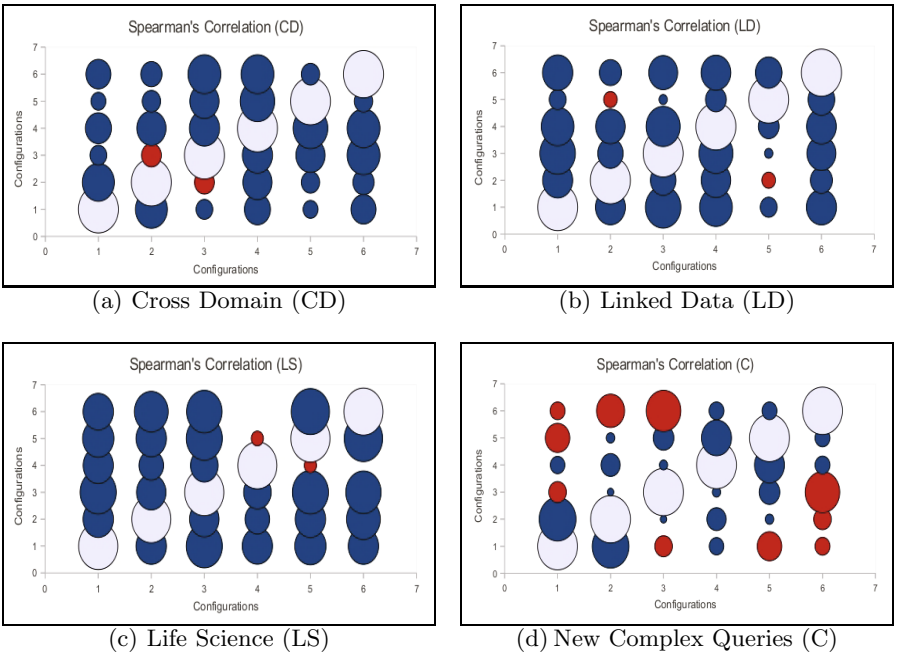
cover some of the missing elements in the former group of queries. They are comprised of between 6 and 48 triple patterns, and can be decomposed into up to 8 sub-queries; and they cover different SPARQL operators. Virtuoso[6] was used to implement endpoints, and the timeout was set up to 240 secs. or 71,000 tuples. Experiments were executed on a Linux Mint machine with an Intel Pentium Core 2 Duo E7500 2.93GHz 8GB RAM 1333MHz DDR3.

Network Latency. We configured a perfect network with no delays. The size of the message corresponded to 16KB.

Data Distribution. We considered two different distributions of the data: *i*) `Complete`: the FedBench collections were stored into a single graph and made accessible through one single SPARQL endpoint, and *ii*) `Federated`: the FedBench collections were stored in nine Virtuoso endpoints.

Therefore, we consider the queries in four groups and six configurations: **Configuration 1:** ANAPSID `Complete` Distribution, **Configuration 2:** ANAPSID `Federated` Distribution, **Configuration 3:** ARQ `Complete` Distribution, **Configuration 4:** ARQ `Federated` Distribution, **Configuration 5:** FedX `Complete` Distribution, **Configuration 6:** FedX `Federated` Distribution. In each configuration, the corresponding queries were ordered according to the total execution time consumed by the corresponding engines. For example, ANAPSID in a `Complete` Distribution, i.e., **Configuration 1**, the Cross-Domain queries were ordered as follows: CD2, CD3, CD4, CD5, CD1, CD7, and CD6. Queries of each configuration were compared using the Spearman's Rho correlation. A high positive value of correlation value between two configurations indicates that the corresponding engines had a similar behavior, i.e., the trends of execution time of the two engines are similar. Thus, when **Configuration 1** is compared to itself, the Spearman's Rho correlation reaches the highest value (1.0). On the other hand, a negative value indicates an inverse correlation; for example, this happened with Complex Queries to ARQ in a `Complete` Distribution (**Configuration 3**) when compared to FedX `Federated` Distribution (**Configuration 6**); its value is -0.757. Finally, a value of 0.0 represents that there is no correlation between the two configurations, e.g., for Life Science queries **Configuration 4** and **Configuration 6**. Figure 1 illustrates the results of this specific study (again, the data used for this study is available through the DEFENDER portal). White circles represent the highest value of correlation; red ones correspond to inverse correlations, while blue ones indicate a positive correlation. The size of the circles is proportional to the value of the correlation. Given a group of queries, a low value of correlation of one engine in two different distributions suggests that the distribution affects the engine behavior, e.g., FedX and ARQ in Complex Queries with different data distributions have correlation values of 0.143 and 0.045, respectively. Furthermore, the number of small blue circles between configurations of different data distributions of the same engine, indicate that this parameter affects the behavior of the studied engine. Because there are several of these points in the Complex Queries plot, we can conclude that

[6] http://virtuoso.openlinksw.com/



(a) Cross Domain (CD) (b) Linked Data (LD)

(c) Life Science (LS) (d) New Complex Queries (C)

Fig. 1. Spearman's Rho Correlation of Queries in three FedBench sets of queries (a) Cross-Domain (CD), (b) Life Science (LS), (c) Linked Data (LD) and (d) New Complex Queries. Six configurations: (1) ANAPSID `Complete` Distribution; (2)ANAPSID `Federated` Distribution; (3) ARQ `Complete` Distribution; (4) ARQ `Federated` Distribution; (5) FedX `Complete` Distribution; (6) FedX `Federated` Distribution. White circles correspond to correlation value of 1.0; blue circles indicate a positive correlation (Fig.1(d) points (3,4) and (5,6) correlation values 0.045 and 0.143, respectively); red circles indicate a negative correlation (Fig.1(d) points (2,6) and (6,3) correlation values -0.5 and -0.757, respectively). Circles' diameters indicate absolute correlation values.

these two parameters (query complexity and data distribution) allow uncovering engines' behavior that could not be observed before. This illustrates the need for the extensions proposed in this paper.

5 Conclusion and Future Work

In this paper we have shown that there is a need to extend current federated SPARQL query testbeds with additional variables and configuration setups (e.g., data partitioning and distribution, network latency, and query complexity), so as to provide more accurate details of the behavior of existing engines, which can then be used to provide better comparisons and as input for improvement proposals. Taking those additional variables into account, we have extensively evaluated three of the existing engines (ANAPSID, ARQ and FedX), and have made available those results for public consumption in the DEFENDER portal,

which we plan to maintain up-to-date on a regular basis. We have also shown how the generated result dataset can be used to validate hypotheses about the systems' behavior.

Our future work plans will be focused on continuing with the evaluation of additional federated SPARQL query engines, and with the inclusion of additional parameters in the benchmark that may still be needed to provide more accurate and well-informed results.

Acknowledgements. This work has been funded by the project myBigData (TIN2010-17060), and DID-USB. We thank Maribel Acosta, Cosmin Basca, and Raúl García-Castro for fruitful discussions.

References

1. Acosta, M., Vidal, M.-E., Lampo, T., Castillo, J., Ruckhaus, E.: ANAPSID: An Adaptive Query Processing Engine for SPARQL Endpoints. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 18–34. Springer, Heidelberg (2011)
2. Buil-Aranda, C., Arenas, M., Corcho, O.: Semantics and Optimization of the SPARQL 1.1 Federation Extension. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part II. LNCS, vol. 6644, pp. 1–15. Springer, Heidelberg (2011)
3. Lynden, S., Kojima, I., Matono, A., Tanimura, Y.: ADERIS: An Adaptive Query Processor for Joining Federated SPARQL Endpoints. In: Meersman, R., Dillon, T., Herrero, P., Kumar, A., Reichert, M., Qing, L., Ooi, B.-C., Damiani, E., Schmidt, D.C., White, J., Hauswirth, M., Hitzler, P., Mohania, M. (eds.) OTM 2011, Part II. LNCS, vol. 7045, pp. 808–817. Springer, Heidelberg (2011)
4. Montoya, G., Vidal, M.-E., Acosta, M.: DEFENDER: a DEcomposer for quEries against feDERations of endpoints. In: Extended Semantic Web Conference, ESWC Workshop and Demo 2012. LNCS (2012)
5. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. TODS 34(3) (2009)
6. Prud'hommeaux, E., Buil-Aranda, C.: SPARQL 1.1 federated query (November 2011)
7. Quilitz, B., Leser, U.: Querying Distributed RDF Data Sources with SPARQL. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 524–538. Springer, Heidelberg (2008)
8. Schmidt, M., Görlitz, O., Haase, P., Ladwig, G., Schwarte, A., Tran, T.: FedBench: A Benchmark Suite for Federated Semantic Data Query Processing. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 585–600. Springer, Heidelberg (2011)
9. Schmidt, M., Hornung, T., Lausen, G., Pinkel, C.: SP2bench: A SPARQL performance benchmark. In: ICDT, pp. 4–33 (2010)
10. Schwarte, A., Haase, P., Hose, K., Schenkel, R., Schmidt, M.: FedX: Optimization Techniques for Federated Query Processing on Linked Data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 601–616. Springer, Heidelberg (2011)

Tag Recommendation for Large-Scale Ontology-Based Information Systems

Roman Prokofyev[1], Alexey Boyarsky[2,3,4], Oleg Ruchayskiy[5], Karl Aberer[2], Gianluca Demartini[1], and Philippe Cudré-Mauroux[1]

[1] eXascale Infolab, University of Fribourg, Switzerland
{firstname.lastname}@unifr.ch

[2] Ecole Polytechnique Fédérale de Lausanne, Switzerland
{firstname.lastname}@epfl.ch

[3] Instituut-Lorentz for Theoretical Physics, U. Leiden, The Netherlands

[4] Bogolyubov Institute for Theoretical Physics, Kiev, Ukraine

[5] CERN TH-Division, PH-TH, Geneva, Switzerland
oleg.ruchayskiy@cern.ch

Abstract. We tackle the problem of improving the relevance of automatically selected tags in large-scale ontology-based information systems. Contrary to traditional settings where tags can be chosen arbitrarily, we focus on the problem of recommending tags (e.g., concepts) directly from a collaborative, user-driven ontology. We compare the effectiveness of a series of approaches to select the best tags ranging from traditional IR techniques such as TF/IDF weighting to novel techniques based on ontological distances and latent Dirichlet allocation. All our experiments are run against a real corpus of tags and documents extracted from the ScienceWise portal, which is connected to ArXiv.org and is currently used by growing number of researchers. The datasets for the experiments are made available online for reproducibility purposes.

1 Introduction

The nature of scientific research is drastically changing. Fewer and fewer scientific advances are carried out by small groups working in their laboratories in isolation. In today's data-driven sciences (be it biology, physics, complex systems or economics), the progress is increasingly achieved by scientists having heterogeneous expertise, working in parallel, and having a very contextualized, local view on their problems and results. We expect that this will result in a fundamental phase transition in how scientific results are obtained, represented, used, communicated and attributed. Different to the classical view of how science is performed, important discoveries will in the future not only be the result of exceptional individual efforts and talents, but alternatively an emergent property of a complex community-based socio-technical system. This has fundamental implications on how we perceive the role of technical systems and in particular information processing infrastructures for scientific work: they are no longer a subordinate instrument that facilitates daily work of highly gifted individuals, but become an essential tool and enabler for performing scientific progress, and eventually might be the instrument within which scientific discoveries are made, represented and brought to use.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 325–336, 2012.

Any such tool should in our opinion possess two central components. One is a *field-specific ontology*, i.e., a structured organization of the knowledge created by the researchers in a given field, along with a formal description of the information and processes they utilize. While in some important cases (e.g., in bioinformatics or chemistry) it is possible to create large ontologies of sufficiently homogeneous concepts and automatically manipulate them using formal rules (see e.g. [13]), the ontology of scientific knowledge *per se* is very complex and vaguely defined at any given point in time. Scientific ontologies can therefore only be created by a combination of existing automatic methods and novel approaches that will enable human-machine collaboration between scientists and the knowledge management infrastructures allowing to combine presentation of new results, in-depth discussions, "user-friendly" introductions for young scientists, and meta-data to relate semantically similar concepts or pieces of content. Today, there are no standard tools to insert, store and query such meta-data online, which mostly remains "in the heads of the experts" [1].

The organization of scientific information does not end with the generation of the scientific ontology. The second crucial element is a set of meaningful connections between such an ontology and the body of research material (papers, books, datasets, etc.). The challenge here is to connect semi-structured data to the natural language content of scientific papers through semantically meaningful relations. This creates a number of challenges to the current state-of-the-art in information retrieval, entity recognition and extraction (since scientific concepts can have many different names and context-dependent meanings).

In this paper, we tackle the problem of *ontology-based tagging*, i.e., of improving the relevance of automatically selected tags in large-scale ontology-based information systems. Contrary to traditional settings where tags can be chosen arbitrarily, we focus on the problem of recommending tags (e.g., concepts) directly from a collaborative, user-driven ontology.

The contributions of this paper are as follows:

- We formally define the task of ontology-based tagging and suggest standard metrics borrowed from Information Retrieval to evaluate it.
- We contribute a real document collection, a domain-specific ontology, and lists of expert-provided tags picked from the ontology and assigned to the documents as a standard evaluation collection for ontology-based tagging.
- We compare the effectiveness of standard Information Retrieval techniques (based on Term Frequency and Inverse Document Frequency) on our evaluation collection.
- We also compare the effectiveness of ontology-based techniques (e.g., based on ontological neighborhood or subsumption) and semantic clustering techniques (such as Latent Semantic Indexing and Dirichlet Allocation).
- Finally, based on the results of our experiments, we draw conclusions w.r.t. the practically and usefulness of using a given technique for ontology-based tagging and discuss future optimizations that could be used to improve our results.

The rest of this paper is structured as follows: We start by discussing related work in Section 2. We briefly present ScienceWise, the infrastructure we leverage on for our experiments, and formally define the task we tackle in Section 3. We discuss our metrics and data sets in Section 4. We report on our experimental results and compare

the effectiveness of various approaches for ontology-based tagging in Section 5, before concluding in Section 6.

2 Related Work

Research on tag recommendation can be classified into two main categories. A first class of approaches look at the contents of the resources while a second type look at the structure connecting users, resources, and tags. Examples of the former class include content-based filtering [11] and collaborative-filtering tag suggestion techniques [17]. Along similar lines, we previously experimented with tag propagation in document graphs in [6]. The latter class includes approaches that focus on the user rather than just providing tag recommendations given a resource. In [10] a set of candidate tags is created and then filtered based on choices made by the user in the past. An approach based on a user-resource-tag graph is FolkRank [8]: It computes popularity scores for resources, users, and tags based on the well-known PageRank algorithm. The assumption is that importance of resources and users propagates to tags.

Word sense disambiguation (WSD) is the task of identifying the correct meaning of an ambiguous word (e.g., 'bank' can indicate either a financial institution or a river bank). A common technique for WSD is to exploit the context of ambiguous words, that is, other words in its vicinity (e.g., in the same sentence). An approach following this idea has been used by Semeraro et al. in [4] where among all the possible senses for a word in WordNet [7], the correct one is chosen by measuring the distance (based on text similarity functions) between the word context and its synsets (i.e., the set of all synonyms for one sense).

Though tag recommendation and disambiguation have been studied extensively (both for free-text tagging and folksonomy systems), surprisingly little research has been carried-out for tag recommendation and disambiguation in a Semantic Web context. Contag [3] is an early system recommending tags by extracting topics using online Web 2.0 services and matching them to an ontology using string similarity. To the best of our knowledge, the present effort is the first systematic and repeatable experimental study of tag recommendation for large-scale and collaborative ontology-based information systems.

3 The ScienceWISE System

The ScienceWISE system allows a community of scientists, working in a specific domain, to generate dynamically as part of their daily work an *interactive semantic environment*, i.e., a field-specific ontology with direct connections to research artifacts (e.g., research papers) and scientific data management services. The central use-cases of ScienceWISE are *annotations* (e.g., adding "supplementary material" or meta-data to scientific artifacts) and *semantic bookmarking* (e.g., creating virtual collections of research papers from `ArXiv.org` [2]).

The system has been public for about one year and is accessible by scientists via our website[1], as well as via `ArXiv.org` and the CERN Document Server[2]. The system cur-

[1] `http://sciencewise.info/`
[2] `http://cds.cern.ch`

rently counts above 200 *active users* (using our services on a regular basis), thousands of annotated papers, and is now receiving several new registrations *daily*.

The domain-specific ontology is central to our system and allows us to integrate all heterogeneous pieces of data and content shared by the users. Since the underlying domain of the ontology is often rapidly changing and only loosely-defined, the best way to keep it up to date is to crowdsource its construction through the community of expert scientists. To create the initial version of the ontology, we have performed a semi-automated import from many science-oriented ontologies and online encyclopedias. After this initial step, ScienceWISE users (who are domain experts) are allowed to edit elements of the ontology (e.g., adding new definitions or new relations) in order to improve both its quality and coverage. Presently, the ScienceWISE ontology counts more than 60'000 unique entries, each with its own definitions, alternative forms, and semantic relations to other entries.

In the context of this paper, we focus on two important and related problems that we have to tackle in order to improve the user experience: tag recommendation and tag disambiguation. We note that those two tasks are key not only in our setting, but for all large-scale, collaborative and ontology-based information systems that are currently gaining momentum on the Internet.

3.1 Tag Recommendation

When users bookmark an `ArXiv.org` paper, our system attempts to automatically select the most relevant tags for characterizing the paper. The tags in question are in our case scientific concepts that are defined in the ontology. A user-friendly interface allows then to correct the system recommendation, e.g., by adding relevant tags or removing irrelevant tags from the top-k list that the system recommended.

More formally, the tag recommendation task can be defined as follows: a set of expert users bookmark scientific papers $\{P_1, \ldots, P_n\} \in \mathcal{P}$. A ranked list of tags $(t_1^j, \ldots, t_{m_j}^j)$ is initially built for each paper P_j by selecting tags from the ontology concepts $(t_i^j \in \mathcal{T} \ \forall \ i, j)$. This list is curated *a posteriori* by the expert users. We write T_{rel}^j to denote the set of relevant tags chosen by the experts for paper P_j. The other tags are defined as irrelevant: $T_{\overline{rel}}^j \equiv \mathcal{T} \setminus T_{rel}^j$.

3.2 Tag Disambiguation

The second problem we tackle is tag disambiguation. Since the same literal can appear in the label of several concepts, it is often difficult to disambiguate isolated terms appearing in a paper. For instance, if *anomaly* appears in the text of a scientific paper, should it be related to the *quantum anomaly* concept, to *experimental anomaly* or to *reactor neutrino anomaly*? All are valid scientific concepts but are however very different semantically. Similarly, depending on the context the abbreviation *DM* can mean *Dark matter* (cosmology), *Distance measure* (astronomy), or *Density matrix* (statistical mechanics).

The goal of this second task is to detect such cases and to develop methods to effectively predict which concept an isolated literal should be related to. Obviously, this second task directly relates to our first task, since disambiguating tags produces more relevant results and hence improves the quality of tag recommendation in the end. Formally, given a term (literal) τ appearing in the text of a paper and a set of automatically

selected tags $\{t_1, \ldots, t_m\}$ corresponding to concepts whose label all contain the literal τ, our goal is to automatically select the right tag(s) $t \in T^{\tau}_{rel}$ corresponding to the correct semantics of the literal as chosen by our expert users.

4 Experimental Setting

4.1 Hypotheses

We consider the following hypotheses for the tag recommendation task: i) concepts appearing in the title and the abstract of a paper are highly relevant to that paper, ii) excluding concepts that are too generic yields better recommendations, and iii) using the structure of the ontology can help us recommend better tags. To evaluate those hypotheses, we compare eight different techniques in Section 5.1.

For the tag disambiguation task, we study whether applying clustering techniques on the papers using their concepts as features allows us to disambiguate concepts with a high accuracy. To evaluate this hypothesis, we test two clustering techniques (LDA and K-means) in Section 5.2.

4.2 Metrics

We evaluate the effectiveness of our approach using four standard metrics borrowed from Information Retrieval:

Precision@k defined as the ratio between the number of relevant tags taken from the top-k recommended tags for paper P_j and the number k of tags considered: $P@k = \frac{\sum_{i=1}^{k} \mathbb{1}(t_i^j \in T_{rel}^j)}{k}$ (where $\mathbb{1}(cond)$ is an indicator function equal to 1 when $cond$ is true and 0 otherwise).

Recall@k defined as the ratio between the number of relevant tags in the top-k for paper P_j and the total number of relevant tags: $R@k = \frac{\sum_{i=1}^{k} \mathbb{1}(t_i^j \in T_{rel}^j)}{|T_{rel}^j|}$

R-Precision defined as Precision@R, where R is the total number of relevant tags for paper P_j: $RP = P@|T_{rel}^j|$.

Average Precision defined as the average of Precision@k values calculated at each rank where a relevant tag is retrieved over the total number of relevant tags: $AP = \frac{\sum_{i=1}^{|T_{rel}^j|} P@i \; \mathbb{1}(t_i^j \in T_{rel}^j)}{|T_{rel}^j|}$.

Those definitions are valid for one paper only. In the following, we also report values averaged over the entire document collection, e.g., Mean Average Precision (MAP) defined as: $MAP = \frac{1}{n} \sum_{j=1}^{n} AP_j$. The metrics for tag disambiguation are derived similarly (see below Section 5.2).

4.3 Data Sets

We use real data as available on our platform for all our experiments. Our document collection contains all the articles bookmarked by our top-5 most prolific users (user

ids 14, 16, 17, 21 and 40). This represents 16'725 scientific papers and 15'083 tags representing 2'157 distinct scientific concepts (out of the 16'725 total number of concepts currently available in our field-specific ontology). If the same paper is bookmarked by more than one user, we take the tags *union* as the relevant set of tags. For the tag disambiguation experiments, we based our experiments on 2'400 articles originating from 6 different top-categories or `ArXiv.org` (400 articles per category).

The experimental data as well as the main scripts we used for our experiments are available on `http://sciencewise.info/media/iswc/`. The data can also be queried using our SPARQL endpoint[3] or browsed online (e.g., `http://data.sciencewise.info/page/bookmarks/2100` gives the bookmark data for paper id 2100).

5 Experimental Results

We report below on our techniques and experimental results for tag recommendation and tag disambiguation.

5.1 Recommending Tags

We compare eight different techniques for tag recommendation below. Most of our approaches are based on term-weighting [15], which is a key technique used in most large-scale information retrieval systems. Basic term-weighting works as follow in our ontology-based context. First, we create an index from the labels of all scientific concepts appearing in the ScienceWISE ontology by considering their stem using Porter's suffix stripping [12]. Then, for each new bookmarked paper, we analyze all the terms appearing in the paper. Given the importance of acronyms in scientific papers, we first determine whether the term is an acronym or not by inspecting its length, capitalization, and by trying to match it to known terms[4]. Two cases can occur at this point: i) if the term is an acronym we consider it *as is* and try to match it to our concept index ii) otherwise, the term is stemmed and then matched using an efficient exact string matching method [9] to the concept index.

We give a brief description of the various methods we experimented with below. We note that each of the following methods was carefully examined and optimized to yield the best possible results we could get after batteries of tests (e.g., we use fined-grained document frequencies and optimal thresholds for all the methods below).

`tf`: Our first approach simply ranks potential tags by counting the number of matches between the terms appearing in the paper and the concept index. While basic, this approach performs relatively well in our context since we consider a restricted number of terms only (our matching process is *mediated* through the ontology). In a standard setting without a field-specific ontology, this approach would perform poorly[5].

[3] `http://d2r.sciencewise.info/openrdf-sesame/repositories/SW`

[4] We consider that the term is an acronym if it is ≤ 5 letters, all capitalized, and if we cannot find it in the Ubuntu corpus of American words [`http://packages.ubuntu.com/lucid/wamerican`]

[5] It would lead to a MAP smaller than 1% in our case.

tfidf: This second method extends the approach above by applying standard TF*IDF [14]. We use a fine-grained document frequency in this case, based on the top categories of papers in `ArXiv.org` rather than the entire document collection (i.e., IDF is computed based on the paper that share the same `ArXiv.org` topic as the paper being bookmarked), as this performs better in practice.

tf_simpleIDF: In the ScienceWISE ontology, some scientific concepts are marked as "basic". While legitimate, those science concepts are deemed rather general by our users and non-specific to any domain (*mass*, or *velocity* are two examples of such concepts). Under the `simpleIDF` scheme, IDF is not computed; rather, the system simply penalizes basic concepts and systematically puts them at the bottom of the ranked list (i.e., the ranked list of basic tags appears after the ranked list of other tags).

tfidf_title: The scientific terms that appear in titles and abstracts of the scientific papers often carry some special significance. Hence, we modify the TF-IDF ranking to promote the concepts appearing in the title into the top positions of the ranking list. Along similar lines, any concept appearing in the abstract has its TF score doubled (which also promotes it higher up in the list of "suggested tags").

tf_title: The same as above, but discarding IDF and only taking into account TF when ranking.

combined: In this approach we combine `tfidf_title` but use `simpleIDF` to compute the document frequency. As we will see below, only marginally impacts on the effectiveness of the approach while drastically reducing computational complexity for large collections of papers. This is the ranking method that we have decided to deploy on our current production version of ScienceWISE.

ont-depth: Scientific concepts are often organized hierarchically in our ontology, with more specific, sub-concepts deriving from higher-level more general concepts. In this approach, we try to penalize more general concepts (that have a smaller depth in the ontology) and favor more specific concepts. More specifically, we penalize more generic concept by $c\_depth/distance\_from\_root\_concept$ where c_depth is a constant (we use $c\_depth = 1$ below, which yields the best results in our setting).

ont-neighbor: Many scientific concepts are linked to further, related concepts in our ontology. Hence, we take advantage of the semantic graph relating the concepts by improving the scores of those concepts that are direct neighbors of top-k ranked concepts. More specifically, we bump the ranking of direct neighbors of top-ranked concepts by $+c\_neighbor$ (we use $c\_neighbor = 3$ below, which yields the best results in our setting).

Figure 1 compares our different approaches on a Precision VS Recall graph along with the overall results in terms of MAP and R-precision. Results for Precision@k are depicted on Figure 2.

We observe the following:

1. Simple TF ranking yields the worst precision. However, a relatively minor improvement (boosting rank of concepts that occur in the title and abstract, technique called `tf_title` in this paper) greatly improves performance for low k.

2. Performance of the `tfidf_title` is only marginally better than `combined`, with the latter one also being considerably faster (since the global IDF measure does not

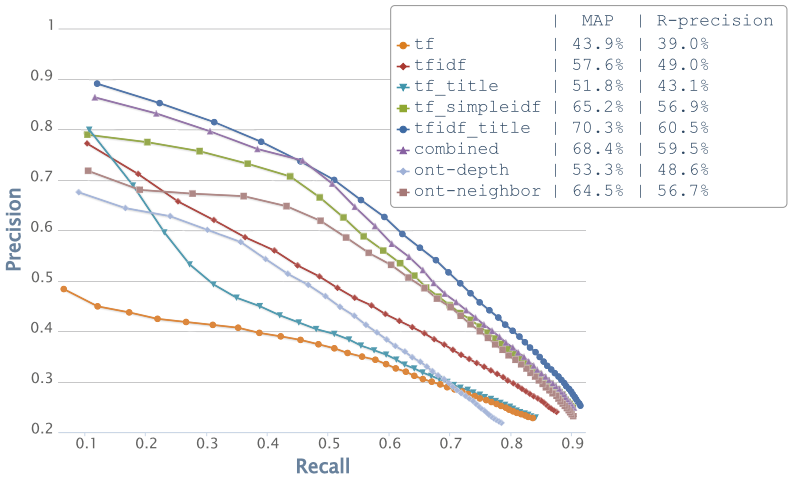


Fig. 1. Precision VS Recall for our various tag recommendation approaches

have to be computed). Both significantly outperform the standard `tfidf` ranking, which demonstrates that one can leverage the structure of scientific texts (where terms in the title and abstract are often very carefully chosen) in order to extract meaningful information.

3. The method leveraging the subsumption relations (*ont-depth*) performs surprisingly poorly. Further variants leveraging the subsumption hierarchies we experimented with behaved even worse. Choosing the right level in the hierarchy seems to be key, and hence favoring too specific (or, conversely, too generic) concepts yields suboptimal results (that are either too specific, and thus unrelated to the paper being analyzed, or too generic and thus are deemed less relevant also).

4. The method based on concept neighborhood (*ont-neighbor*) performs relatively well but is not better than simpler methods. The problem in that case seems to lie in the semantics of the relations between the concepts, which are often arbitrary in our ScienceWISE ontology and hence interconnect semantically heterogeneous concepts. One way of correcting this would be to (automatically or manually) create additional *same-as* or *see-also* relationships in our ontology, and to leverage such relationships to return additional relevant results (we successfully applied such techniques recently on the LOD graph, see [16]).

In summary, the careful use of some specific properties of the ontology (e.g., *basic* concepts) together with information about position of the terms in the document (e.g., in the title or abstract) allow to significantly increase precision in comparison with the baseline methods (increasing MAP up to 70%).

5.2 Disambiguating Tags

In order to tackle our second problem, we have implemented a special interface, that permits a user to confirm or provide a disambiguation for abbreviations or ambiguous concepts when bookmarking a paper. To help the user in this task, we cluster the

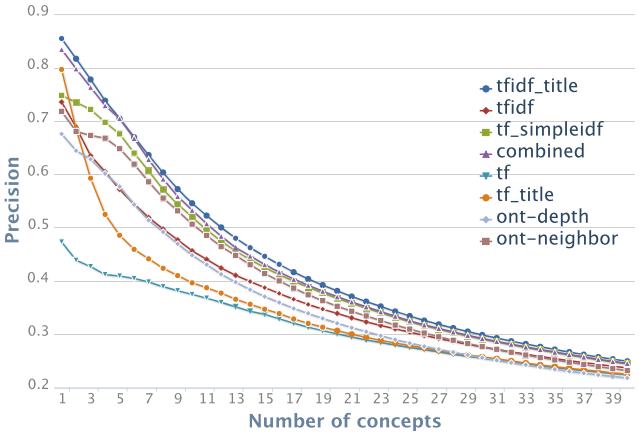


Fig. 2. Precision@k of our various ranking techniques for tag recommendation

collection of bookmarked papers into *topics* in an attempt to guess the correct disambiguation. We start by experimenting with the following techniques:

`lda`: Dirichlet Allocation (LDA) [5] is a standard tool in probabilistic topic modeling. Applied to IR, LDA basically considers that each document is a mixture of a small number of topics and that each word is attributable to one of the topics. It is conceptually similar to probabilistic latent semantic analysis, except that in LDA the topic distributions are assumed to have Dirichlet priors, which often lead to better results in practice. We have use the LDA implementation as available from the Mallet package[6] in our experiments.

`k-means`: works similarly but takes advantage of the well-known k-means clustering technique to cluster the documents.

Since the results produced by both clustering methods only define attribution of each paper to the cluster and does not tell exactly

We consider our data set comprising papers from several disjoint `ArXiv.org` subject classes[7] and split these collections into clusters using LDA and K-Means algorithms. The number of clusters is chosen to be equal to those of primary `ArXiv.org` subject classes.

Next, we use the resulting classification to generate a set of suggestions for the concepts/abbreviation disambiguation. Using our test collection, we determine for each paper its primary subject class (equivalently, topic) and generated a list of suggestions based on this. The results are shown in Figure 3.

The actual accuracy of LDA-based disambiguation is impressive (**75%**). One can in addition add ontological information to improve the disambiguation process and further

[6] `http://mallet.cs.umass.edu/`

[7] Each paper on `ArXiv.org` belongs to one or several *Subject classes*, chosen by the authors of the paper.

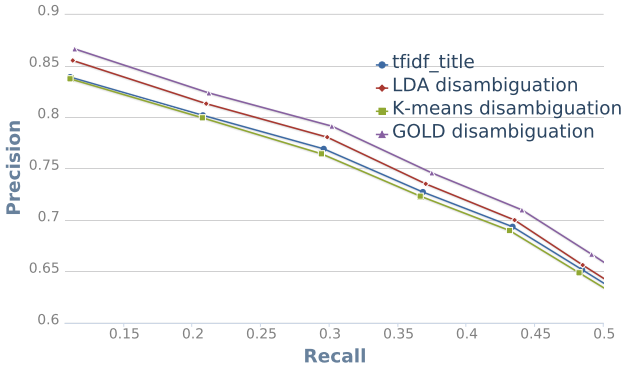


Fig. 3. Precision VS Recall using tag disambiguation

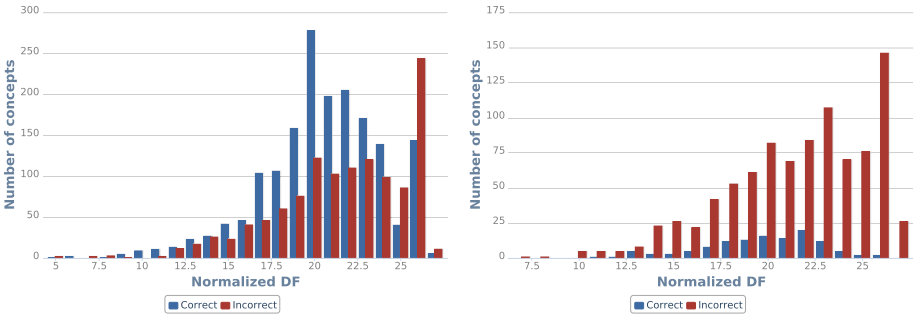


Fig. 4. Comparison of document frequency distribution for one-word concepts from the first 5 positions in the ranking (left panel) and from the positions (6–12). NormalizedDF is defined via Eq. (1) in the text.

boost the accuracy. For example, if among the concepts to disambiguate there is both a concept and subconcept (e.g. *power spectrum* and *matter power spectrum*) and if we provide the most specific concept, the accuracy raises to **88%**. We compare this to the standard k-means clustering algorithm, which only yields an accuracy of **47%**.

Composite Concepts. Another approach to the disambiguation problem we experimented with is based on mereology and *composite concepts*. Concepts in a scientific ontology can often be expressed as *composites* of some other ontological concepts. For example, a concept *mass of particle* is a composite of two basic scientific concepts: *mass* and *particle*. Very often the composite concepts are presented in many different literal forms. Moreover, it is custom to "shorten" the term (e.g. use *mass* instead of *mass of a star*, or simply *cluster* instead of *galaxy cluster*). Although this situation is formally similar to the previous one, it is impossible to guess what concepts should be disambiguated.

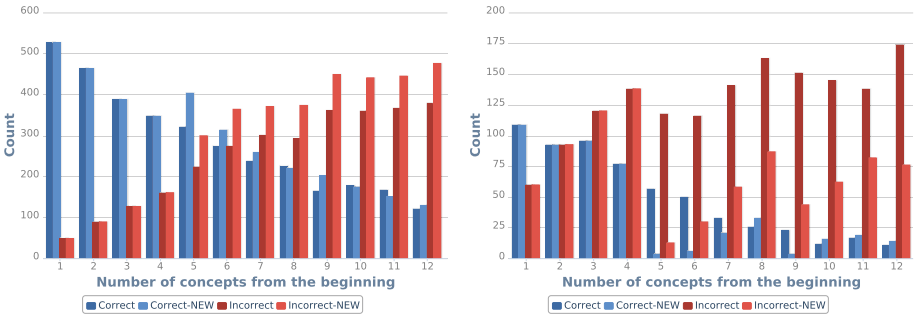


Fig. 5. Comparison of acceptance/rejection rate as a function of position in the ranking list before and after penalization of one-word concepts. Left panel shows change of the rejection rate for all concepts, right panel demonstrates rejection rate for one-word concepts.

We have tested a hypothesis that *one-word concepts more often have a "generic meaning" than their many-words counterparts.*. If this is really the case, a proper tuning of the IDF function would be able to improve the ranking significantly. To determine whether this is indeed the case, we considered the *document frequency* (DF) distribution for the one-word tags. The normalized DF on the x-axis is defined as

$$\text{normalized DF} = \log_{1.5}\left(\frac{\text{number of docs. containing a concept}}{\text{total number of docs. in collection}} \times 10^5\right) \qquad (1)$$

The corresponding histograms are shown in Fig. 4 where one can see (quite surprisingly) that the DF distribution for "correct" and "incorrect" concepts are roughly the same (although the correct ones are shifted somewhat to the lower DF region). Therefore, the one-word concepts bear no clear correlation with the document frequency. Based on these results, we decided to implement a simple strategy for one-word concepts that appear in position 6 and below in our `tf_baseline` ranking list are further penalized. The results of this experiment are shown in Fig. 5. Applied on our tag recommendation strategy, such a disambiguation approach yields and improvement in MAP of about 0.5% on average.

6 Conclusions

In this paper, we addressed the problem of ontology-based tagging of scientific papers. We compared the effectiveness of various methods to recommend and disambiguate tags within a large-scale information system. Compared to classic tag recommendation, the proposed techniques select tags directly from a collaborative, user-driven ontology. Extensive experiments have shown that the use of a community-authored ontology together with information about the position of the concepts in the documents allows to significantly increase precision over standard methods. Also, several more specific techniques such as ontology-based neighborhood selection, LDA classification and one-word-concept penalization for tag disambiguation yield surprisingly good results and collectively represent a good basis for further experimentation and optimizations.

References

1. Aberer, K., Boyarsky, A., Cudré-Mauroux, P., Demartini, G., Ruchayskiy, O.: An integrated socio-technical crowdsourcing platform for accelerating returns in escience. In: ISWC (Outrageous Ideas Track) (2011)
2. Aberer, K., Boyarsky, A., Cudré-Mauroux, P., Demartini, G., Ruchayskiy, O.: ScienceWISE: a Web-based Interactive Semantic Platform for scientic collaboration. In: ISWC (Demonstration Track) (2011)
3. Adrian, B., Sauermann, L., Roth-berghofer, T.: Contag: A semantic tag recommendation system. In: Proceedings of ISemantics 2007, pp. 297–304. JUCS (2007)
4. Basile, P., Degemmis, M., Gentile, A.L., Lops, P., Semeraro, G.: The JIGSAW Algorithm for Word Sense Disambiguation and Semantic Indexing of Documents. In: Basili, R., Pazienza, M.T. (eds.) AI*IA 2007. LNCS (LNAI), vol. 4733, pp. 314–325. Springer, Heidelberg (2007)
5. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022 (2003)
6. Budura, A., Michel, S., Cudré-Mauroux, P., Aberer, K.: Neighborhood-Based Tag Prediction. In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 608–622. Springer, Heidelberg (2009)
7. Fellbaum, C.: Wordnet. In: Theory and Applications of Ontology: Computer Applications, pp. 231–243 (2010)
8. Jäschke, R., Marinho, L.B., Hotho, A., Schmidt-Thieme, L., Stumme, G.: Tag recommendations in social bookmarking systems. AI Commun. 21(4), 231–247 (2008)
9. Knuth, D.E., Morris Jr., J.H., Pratt, V.R.: Fast pattern matching in strings. SIAM Journal on Computing 6(2), 323–350 (1977)
10. Lipczak, M.: Tag recommendation for folksonomies oriented towards individual users. In: ECML PKDD Discovery Challenge (2008)
11. Mishne, G.: Autotag: a collaborative approach to automated tag assignment for weblog posts. In: Carr, L., De Roure, D., Iyengar, A., Goble, C.A., Dahlin, M. (eds.) WWW, pp. 953–954. ACM (2006)
12. Porter, M.F.: An algorithm for suffix stripping. In: Readings in Information Retrieval, pp. 313–316. Morgan Kaufmann Publishers Inc., San Francisco (1997)
13. Sahoo, S.S., Sheth, A., Henson, C.: Semantic provenance for escience: Managing the deluge of scientific data. IEEE Internet Computing 12(4), 46–54 (2008)
14. Salton, G., McGill, M.J.: Introduction to modern information retrieval (1986)
15. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. 24(5), 513–523 (1988)
16. Tonon, A., Demartini, G., Cudre-Mauroux, P.: Combining inverted indices and structured search for ad-hoc object retrieval. In: SIGIR (2012)
17. Xu, Z., Fu, Y., Mao, J., Su, D.: Towards the semantic web: Collaborative tag suggestions. In: Collaborative Web Tagging Workshop at WWW 2006, Edinburgh, Scotland (2006)

Evaluation of Techniques for Inconsistency Handling in OWL 2 QL Ontologies

Riccardo Rosati, Marco Ruzzi, Mirko Graziosi, and Giulia Masotti

DIAG, Sapienza Università di Roma
Via Ariosto 25, I-00185 Roma, Italy

Abstract. In this paper we present the Quonto Inconsistent Data handler (QuID). QuID is a reasoner for OWL 2 QL that is based on the system Quonto and is able to deal with inconsistent ontologies. The central aspect of QuID is that it implements two different, orthogonal strategies for dealing with inconsistency: ABox repairing techniques, based on data manipulation, and consistent query answering techniques, based on query rewriting. Moreover, by exploiting the ability of Quonto to delegate the management of the ABox to a relational database system (DBMS), such techniques are potentially able to handle very large inconsistent ABoxes. For the above reasons, QuID allows for experimentally comparing the above two different strategies for inconsistency handling in the context of OWL 2 QL. We thus report on the experimental evaluation that we have conducted using QuID. Our results clearly point out that inconsistency-tolerance in OWL 2 QL ontologies is feasible in practical cases. Moreover, our evaluation singles out the different sources of complexity for the data manipulation technique and the query rewriting technique, and allows for identifying the conditions under which one method is more efficient than the other.

1 Introduction

One of the most important current issues in OWL ontology management is dealing with inconsistency, that is, the presence of contradictory information in the ontology [8]. It is well-known that the classical semantics of OWL and Description Logics (DL) is not *inconsistency-tolerant*, i.e., it does not allow for using in a meaningful way any piece of information in an inconsistent ontology. On the other hand, the size of ontologies used by real applications is scaling up, and ontologies are increasingly merged and integrated into larger ontologies: the probability of creating inconsistent ontologies is consequently getting higher and higher (see e.g. [4]).

In this paper we focus on *ABox inconsistency*, i.e., the case of inconsistent ontologies where the TBox (intensional part of the ontology) is consistent, while the ABox (extensional part of the ontology) is inconsistent with the TBox, i.e., a subset of the assertions in the ABox contradicts one or more TBox assertions.

We follow an approach that is formally based on inconsistency-tolerant semantics; such semantics overcome the limitations of the classical DL semantics in inconsistency management. In particular, we consider inconsistency-tolerant semantics for general DLs recently proposed in [5], called IAR *semantics*, for which reasoning has been studied in the context of the Description Logics of the *DL-Lite* family, and in particular

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 337–349, 2012.

the DL *DL-Lite*$_A$, that underlies the OWL profile OWL 2 QL. The IAR semantics is centered around the notion of *ABox repair*, which is a very simple and natural one: the ABox repair of a DL ontology is the intersection of all the maximal subsets of the ABox that are consistent with the TBox.

Recently, two different methods for reasoning under the IAR inconsistency-tolerant semantics have been studied: techniques based on the computation of the ABox repair (*ABox cleaning*) and techniques based on a tranformation of the queries posed to the (possibly inconsistent) ontology (*consistent query rewriting*). In particular, in [5] it was proved that computing the ABox repair of a *DL-Lite*$_A$ ontology under the IAR semantics is a tractable problem. Then, in [6] a technique for query answering under IAR-semantics in *DL-Lite*$_A$ is presented: instead of modifying the ABox, this method is based on computing a rewriting Q' of the initial query Q and then evaluating the query Q' with respect to the original ABox.

We argue that the results of [5,6] are potentially very important from the practical viewpoint, for the following reasons: (i) they are based on formally grounded notions of inconsistency-tolerant semantics; (ii) they identify (to the best of our knowledge) the first inconsistency-tolerant semantics in DLs for which query answering is tractable. So, based on such results, in principle it might be possible to define practical algorithms for handling inconsistency in OWL 2 QL.

This paper starts from the above results, and tries to provide an experimental evaluation and comparison of both the ABox cleaning approach and the consistent query rewriting approach mentioned above. In particular, our main goal was to address the following fundamental questions: (i) is ABox cleaning a feasible technique? (ii) is consistent query rewriting a feasible technique? (iii) under which conditions consistent query rewriting is to prefer to ABox cleaning (and vice versa)?

In this paper, we provide the following contributions:

(1) We present effective techniques for both ABox cleaning and consistent query rewriting in *DL-Lite*$_A$/OWL 2 QL under IAR semantics. To this aim, we present the QUonto Inconsistent Data handler (QuID), that implements, within the Quonto system,[1] techniques for both the computation of the ABox repair of a *DL-Lite*$_A$ ontology under the above semantics, as well as techniques for computing the consistent query rewriting of queries. QuID constitutes (to the best of our knowledge) the first implementation of tractable algorithms for handling inconsistent instances in OWL ontologies. Moreover, Quonto delegates the management of the ABox to a relational database system (DBMS). Therefore, for ABox cleaning, all modifications of the ABox are delegated to the DBMS through SQL queries and updates; and for consistent query rewriting, the rewritten query can be directly executed by the DBMS on the original database. This potentially allows for handling inconsistency in very large ABoxes under both techniques.

(2) We present the results of a set of experiments that we have conducted using QuID. These results clearly show that ABox cleaning in *DL-Lite*$_A$ is actually scalable: QuID is able to efficiently compute the IAR repair of both complex and large ontologies, whose ABoxes contain up to millions of assertions and have hundreds of thousands of assertions inconsistent with the TBox. On the other hand, the results for the query answering

[1] http://www.dis.uniroma.it/~quonto

technique based on consistent query rewriting are in general less encouraging, since the structural complexity of the reformulated queries makes the whole query answering process slower than the approach based on ABox cleaning, although consistent query rewriting does not require pre-processing of the ABox.

(3) Our experimental results allow us to understand the actual impact of the different aspects involved in the computation of the ABox repair and in consistent query rewriting, and the limits and possibilities of the two approaches implemented in QuID.

The rest of the paper is organized as follows. In Section 2, we present a detailed algorithm for computing IAR repairs in *DL-Lite*$_A$. In Section 3, we briefly recall the algorithm presented in [6] for consistent query rewriting under IAR semantics in *DL-Lite*$_A$. In Section 4 we present the QuID system and report on the experimental evaluation we have conducted with QuID. Finally, in Section 5 we conclude the paper.

2 ABox Cleaning Technique for OWL 2 QL

We start by briefly recalling the DL *DL-Lite*$_A$ and the IAR semantics.

In this paper we consider DL ontologies specified in *DL-Lite*$_A$, a member of the *DL-Lite* family of tractable Description Logics [2,1], which is at the basis of OWL 2 QL, one of the profiles of OWL 2, the ontology specification language of the World Wide Web Consortium (W3C). *DL-Lite*$_A$ distinguishes concepts from *value-domains*, which denote sets of (data) values, and roles from *attributes*, which denote binary relations between objects and values. Concepts, roles, attributes, and value-domains in this DL are formed according to the following syntax:

$$\begin{array}{ll} B \longrightarrow A \mid \exists Q \mid \delta(U) & E \longrightarrow \rho(U) \\ C \longrightarrow B \mid \neg B & F \longrightarrow \top_D \mid T_1 \mid \cdots \mid T_n \\ Q \longrightarrow P \mid P^- & V \longrightarrow U \mid \neg U \\ R \longrightarrow Q \mid \neg Q & \end{array}$$

In such rules, A, P, and U respectively denote an atomic concept (i.e., a concept name), an atomic role (i.e., a role name), and an attribute name, P^- denotes the inverse of an atomic role, whereas B and Q are called basic concept and basic role, respectively. Furthermore, $\delta(U)$ denotes the *domain* of U, i.e., the set of objects that U relates to values; $\rho(U)$ denotes the *range* of U, i.e., the set of values that U relates to objects; $\top_D$ is the universal value-domain; $T_1, \ldots, T_n$ are n pairwise disjoint unbounded value-domains. A *DL-Lite*$_A$ ontology is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is the TBox and $\mathcal{A}$ the ABox. The TBox $\mathcal{T}$ is a finite set of assertions of the form

$$B \sqsubseteq C \qquad Q \sqsubseteq R \qquad E \sqsubseteq F \qquad U \sqsubseteq V \qquad (\mathsf{funct}\ Q) \qquad (\mathsf{funct}\ U)$$

From left to right, the first four assertions respectively denote inclusions between concepts, roles, value-domains, and attributes. In turn, the last two assertions denote functionality on roles and on attributes. In fact, in *DL-Lite*$_A$ TBoxes we further impose that roles and attributes occurring in functionality assertions cannot be specialized (i.e., they cannot occur in the right-hand side of inclusions). In practice, the only difference between *DL-Lite*$_A$ and OWL 2 QL lies in the presence of functionality assertions (which

are not allowed in OWL 2 QL). Due to space limitations, we refer the reader to [7] for details on the semantics of *DL-Lite*$_A$.

We then briefly recall the IAR semantics for inconsistency-tolerance in DL ontologies (see [5] for more details). Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL ontology. Then, the IAR*-repair* of $\mathcal{K}$ is defined as the ABox corresponding to the intersection of all the maximal subsets of $\mathcal{A}$ that are consistent with $\mathcal{T}$. A first-order formula ϕ is entailed by $\mathcal{K}$ under the IAR semantics if ϕ is entailed by $\langle \mathcal{T}, \mathcal{A}_R \rangle$ under the standard DL semantics, where $\mathcal{A}_R$ is the IAR-repair of $\mathcal{K}$. We are interested in checking (Boolean) unions of conjunctive queries (UCQs) over DL ontologies.

The technique for computing the IAR-repair of a *DL-Lite*$_A$ ontology $\langle \mathcal{T}, \mathcal{A} \rangle$ is based on the idea of deleting from $\mathcal{A}$ all the membership assertions participating in *minimal* conflict sets for $\mathcal{T}$. As shown in [5], this task is relatively easy (in particular, tractable) in *DL-Lite*$_A$ because the following property holds: for every *DL-Lite*$_A$ TBox $\mathcal{T}$, all the minimal conflict sets for $\mathcal{T}$ are either unary conflict sets or binary conflict sets. This property is actually crucial for tractability of reasoning under IAR semantics.

We now present a detailed algorithm for computing the IAR-repair of a *DL-Lite*$_A$ ontology. This algorithm exploits the techniques presented in [5], whose aim was only to provide PTIME upper bounds for the problem of computing such repairs. In particular, the present algorithms specify efficient ways of detecting minimal conflict sets. Instead, the previous techniques check all unary and binary subsets of the ABox for these purposes.

In the following, we call *annotated ABox assertion* an expression ξ of the form $\langle \alpha, \gamma \rangle$ where α is an ABox assertion and γ is a value in the set $\{cons, ucs, bcs\}$. Furthermore, we call *annotated ABox* a set of annotated ABox assertions. The intuition behind an annotated ABox assertion ξ is that its annotation γ expresses whether the associated ABox expression α does not participate in any minimal conflict set (*cons*) or participates in a unary conflict set (*ucs*) or to a binary conflict set (*bcs*).

The following algorithm QuID-IAR-repair computes the IAR-repair of a *DL-Lite*$_A$ ontology. For ease of exposition, the algorithm does not report details on the treatment of attributes, which are actually handled in a way analogous to roles. In the following, we denote concept names with the symbol A, role names with the symbol P, basic concepts (that is, a concept name A or the domain of a role $\exists P$ or the range of a role $\exists P^-$) with the symbols B_1, B_2, and basic roles (that is, either a role name P or the inverse of a role name P^-) with the symbols R, S. Moreover, the expression $B(a)$ with B basic concept denotes: the instance assertion $A(a)$ if $B = A$; an instance assertion of the form $P(a, b)$ if $B = \exists P$; an instance assertion of the form $P(b, a)$ if $B = \exists P^-$.

Algorithm QuID-IAR-repair($\mathcal{K}$)
input: *DL-Lite*$_A$ ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, **output**: IAR-repair of $\mathcal{K}$
begin
// STEP 1: create annotated ABox $\mathcal{A}_{ann}$
  $\mathcal{A}_{ann} = \emptyset$;
  for each $\alpha \in \mathcal{A}$ **do** $\mathcal{A}_{ann} = \mathcal{A}_{ann} \cup \langle \alpha, cons \rangle$;
// STEP 2: detect unary conflict sets in $\mathcal{A}_{ann}$
  for each concept name A s.t. $\mathcal{T} \models A \sqsubseteq \neg A$ **do**
    for each $\xi = \langle A(a), cons \rangle \in \mathcal{A}_{ann}$ **do** $\mathcal{A}_{ann} = \mathcal{A}_{ann} - \{\xi\} \cup \{\langle A(a), ucs \rangle\}$;
  for each role name P s.t. $\mathcal{T} \models P \sqsubseteq \neg P$ **do**

for each $\xi = \langle P(a,b), cons\rangle \in \mathcal{A}_{ann}$ **do** $\mathcal{A}_{ann} = \mathcal{A}_{ann} - \{\xi\} \cup \{\langle P(a,b), ucs\rangle\}$;
for each role name P s.t. $\mathcal{T} \models P \sqsubseteq \neg P^-$ or $\mathcal{T} \models \exists P \sqsubseteq \neg \exists P^-$ **do**
for each $\xi = \langle P(a,a), cons\rangle \in \mathcal{A}_{ann}$ **do** $\mathcal{A}_{ann} = \mathcal{A}_{ann} - \{\xi\} \cup \{\langle P(a,a), ucs\rangle\}$;
// STEP 3: detect binary conflict sets in $\mathcal{A}_{ann}$
for each disjointness $B_1 \sqsubseteq \neg B_2$ such that $\mathcal{T} \models B_1 \sqsubseteq \neg B_2$ **do**
for each pair $\xi_1 = \langle B_1(a), \gamma_1\rangle, \xi_2 = \langle B_2(a), \gamma_2\rangle \in \mathcal{A}'_{ann}$ such that $\gamma_1, \gamma_2 \neq ucs$ **do**
$\mathcal{A}_{ann} = \mathcal{A}_{ann} - \{\xi_1, \xi_2\} \cup \{\langle B_1(a), bcs\rangle, \langle B_2(a), bcs\rangle\}$;
for each disjointness $R \sqsubseteq \neg S$ such that $\mathcal{T} \models R \sqsubseteq \neg S$ **do**
for each pair $\xi_1 = \langle R(a,b), \gamma_1\rangle, \xi_2 = \langle S(a,b), \gamma_2\rangle \in \mathcal{A}'_{ann}$ such that $\gamma_1, \gamma_2 \neq ucs$ **do**
$\mathcal{A}_{ann} = \mathcal{A}_{ann} - \{\xi_1, \xi_2\} \cup \{\langle R(a,b), bcs\rangle, \langle S(a,b), bcs\rangle\}$;
for each functionality assertion (funct R) $\in \mathcal{T}$ **do**
for each pair $\xi_1 = \langle R(a,b), \gamma_1\rangle, \xi_2 = \langle R(a,c), \gamma_2\rangle \in \mathcal{A}'_{ann}$
such that $b \neq c$ and $\gamma_1, \gamma_2 \neq ucs$ **do**
$\mathcal{A}_{ann} = \mathcal{A}_{ann} - \{\xi_1, \xi_2\} \cup \{\langle R(a,b), bcs\rangle, \langle R(a,c), bcs\rangle\}$;
// STEP 4: extract the IAR repair from $\mathcal{A}_{ann}$
$\mathcal{A}' = \emptyset$;
for each $\langle \alpha, cons\rangle \in \mathcal{A}_{ann}$ **do** $\mathcal{A}' = \mathcal{A}' \cup \{\alpha\}$;
return $\mathcal{A}'$
end

The algorithm QuID-IAR-repair consists of four steps which can be informally described as follows.

step 1. *copy of $\mathcal{A}$ into an annotated ABox $\mathcal{A}_{ann}$*. In this step, the value of the annotation is initialized to *cons* for all ABox assertions.

step 2. *detection of the unary conflict sets in $\mathcal{A}_{ann}$*. For every assertion of the form $\xi = \langle \alpha, cons\rangle$, such that $\{\alpha\}$ is a unary conflict set for $\mathcal{T}$, $\mathcal{A}_{ann} = \mathcal{A}_{ann} - \{\xi\} \cup \{\langle \alpha, ucs\rangle\}$, i.e., the annotation relative to α is changed to *ucs*. Unary conflict sets are actually detected through TBox reasoning, by looking at empty concepts and roles in $\mathcal{T}$, as well as asymmetric roles, i.e., roles disjoint with their inverse.

step 3. *detection of the binary conflict sets in $\mathcal{A}_{ann}$*. For every pair of assertions of the form $\xi_1 = \langle \alpha_1, \gamma_1\rangle, \xi_2 = \langle \alpha_2, \gamma_2\rangle$ such that $\gamma_1 \neq ucs$ and $\gamma_2 \neq ucs$ and $\{\alpha, \beta\}$ is a binary conflict set for $\mathcal{T}$, $\mathcal{A}_{ann} = \mathcal{A}_{ann} - \{\xi_1, \xi_2\} \cup \{\langle \alpha, bcs\rangle, \langle \beta, bcs\rangle\}$, i.e., the annotation relative to α and β is changed to *bcs*. As in the case of unary conflict sets, to find binary conflict sets the algorithm looks for disjoint concepts and roles in $\mathcal{T}$, as well as functional roles.

step 4. *extraction of the IAR-repair from $\mathcal{A}_{ann}$*. The IAR-repair can be now simply extracted from the annotated ABox $\mathcal{A}_{ann}$, by eliminating both unary conflict sets and binary conflict sets. Therefore, for every assertion of the form $\langle \alpha, cons\rangle$ in $\mathcal{A}_{ann}$, α is copied into the (non-annotated) ABox $\mathcal{A}'$ which is finally returned by the algorithm.

Correctness of the above algorithm can be proved starting from the results in [5].

Theorem 1. *Let $\mathcal{K}$ be a DL-Lite$_A$ ontology and let $\mathcal{A}'$ be the ABox returned by QuID-IAR-repair($\mathcal{K}$). Then, $\mathcal{A}'$ is the IAR repair of $\mathcal{K}$.*

3 Perfect Reformulation of UCQs under IAR Semantics

We now briefly recall the query rewriting technique proposed in [6]. Such a technique computes a first-order query Q' starting from a union of conjunctive queries Q and a *DL-Lite*$_A$ TBox $\mathcal{T}$. The query Q' is a *perfect reformulation of* Q *with respect to* $\mathcal{T}$ *under the* IAR *semantics*, i.e., Q' is such that, for every ABox $\mathcal{A}$, the answers to Q over $\langle \mathcal{T}, \mathcal{A} \rangle$ under the IAR semantics correspond to to the answers to Q' computed over the ABox $\mathcal{A}$ only. Due to space limits, here we just report the main definitions of the query rewriting technique: we refer the reader to [6] for more details on the method.

The first definition that we give can be used to establish whether a certain atom is consistent with the TBox axioms. Let A be an atomic concept in $\Gamma_{\mathcal{O}}$ and t a term (i.e., either a constant or a variable symbol), we pose $ConsAt_A^{\mathcal{T}}(t) = false$ if $\mathcal{T} \models A \sqsubseteq \neg A$, $true$ otherwise. That is, $ConsAt_A^{\mathcal{T}}(t)$ is $false$ if and only if the concept A is unsatisfiable. For an atomic role $P \in \Gamma_{\mathcal{O}}$ and terms t, t', we define: (i) $ConsAt_P^{\mathcal{T}}(t, t') = false$ if $\mathcal{T} \models P \sqsubseteq \neg P$; (ii) $t \neq t'$ if $\mathcal{T} \models P \sqsubseteq \neg P^-$ or $\mathcal{T} \models \exists P \sqsubseteq \neg \exists P^-$; (iii) $true$ otherwise (an analogous definition holds for an attribute $U \in \Gamma_{\mathcal{O}}$ and terms t and t').

Now we deal with possible clashes involving negative inclusions, which are also called *disjointnesses*. Let B be a basic concept built from an atomic concept or an atomic role of $\Gamma_{\mathcal{O}}$, and let t be a term. Then, we define $NotDisjClash_B^{\mathcal{T}}(t)$ as the following FOL formula:

$$\bigwedge_{A \in DCN(B,\mathcal{T})} \neg(A(t) \wedge ConsAt_A^{\mathcal{T}}(t)) \wedge \bigwedge_{P \in DRD(B,\mathcal{T})} \neg(\exists y.P(t,y) \wedge ConsAt_P^{\mathcal{T}}(t,y)) \wedge$$
$$\bigwedge_{P \in DRR(B,\mathcal{T})} \neg(\exists y.P(y,t) \wedge ConsAt_P^{\mathcal{T}}(y,t)) \wedge \bigwedge_{U \in DAD(B,\mathcal{T})} \neg(\exists y.U(t,y) \wedge ConsAt_U^{\mathcal{T}}(t,y))$$

where y is a variable symbol such that $y \neq t$, DCN, DRD, DRR, and DAD are defined as follows:

$$DCN(B,\mathcal{T}) = \{A \mid A \text{ is an atomic concept of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models B \sqsubseteq \neg A\}$$
$$DRD(B,\mathcal{T}) = \{P \mid P \text{ is an atomic role of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models B \sqsubseteq \neg \exists P\}$$
$$DRR(B,\mathcal{T}) = \{P \mid P \text{ is an atomic role of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models B \sqsubseteq \neg \exists P^-\}$$
$$DAD(B,\mathcal{T}) = \{U \mid U \text{ is an attribute of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models B \sqsubseteq \neg \delta(U)\}$$

Let us now consider disjointness clashes for roles. Let P be a role name from $\Gamma_{\mathcal{O}}$ and let t, t' be terms, we define the formula $NotDisjClash_P^{\mathcal{T}}(t, t')$ as follows:

$$\bigwedge_{S \in DisjRoles(P,\mathcal{T})} \neg(S(t,t') \wedge ConsAt_S^{\mathcal{T}}(t,t')) \wedge NotDisjClash_{\exists P}^{\mathcal{T}}(t) \wedge$$

$$\bigwedge_{S \in DisjInvRoles(P,\mathcal{T})} \neg(S(t',t) \wedge ConsAt_S^{\mathcal{T}}(t',t)) \wedge NotDisjClash_{\exists P^-}^{\mathcal{T}}(t')$$

where, again, if either t or t' are variable symbols, then they are free variables, and the sets $DisjRoles(P,\mathcal{T})$ and $DisjInvRoles(P,\mathcal{T})$ are defined as follows:

$$DisjRoles(P,\mathcal{T}) = \{S \mid S \text{ is a role name of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models P \sqsubseteq \neg S\}$$
$$DisjInvRoles(P,\mathcal{T}) = \{S \mid S \text{ is a role name of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models P \sqsubseteq \neg S^-\}.$$

Intuitively, $NotDisjClash_P^{\mathcal{T}}(t, t')$ will be used in the reformulation to deal with possible violations of negative inclusions involving P. This means considering role inclusions, through the sets $DisjRoles(P,\mathcal{T})$ and $DisjInvRoles(P,\mathcal{T})$, and concept inclusions of the form $\exists P \sqsubseteq \neg B$ and of the form $\exists P^- \sqsubseteq \neg B$, through the use

of $NotDisjClash^{\mathcal{T}}_{\exists P}(t)$ and $NotDisjClash^{\mathcal{T}}_{\exists P^-}(t')$, respectively. $ConsAt^{\mathcal{T}}_S(t,t')$ plays here a role analogous to the one played by $ConsAt$ formulas in $NotDisjClash^{\mathcal{T}}_B(t)$. (The function $NotDisjClash^{\mathcal{T}}_U$ for attributes U is defined in an analogous way.)

Finally, we consider clashes on functionalities and define $NotFunctClash^{\mathcal{T}}_P(t,t')$ as the following FOL formula:

- if (funct P) $\notin \mathcal{T}$ and (funct P^-) $\notin \mathcal{T}$, then $NotFunctClash^{\mathcal{T}}_P(t,t') = \mathit{true}$;
- if (funct P) $\in \mathcal{T}$ and (funct P^-) $\notin \mathcal{T}$, then $NotFunctClash^{\mathcal{T}}_P(t,t') = \neg(\exists y.P(t,y) \wedge y \neq t' \wedge ConsAt^{\mathcal{T}}_P(t,y))$;
- if (funct P) $\notin \mathcal{T}$ and (funct P^-) $\in \mathcal{T}$, then $NotFunctClash^{\mathcal{T}}_P(t,t') = \neg(\exists y.P(y,t') \wedge y \neq t \wedge ConsAt^{\mathcal{T}}_P(y,t))$;
- if (funct P) $\in \mathcal{T}$ and (funct P^-) $\in \mathcal{T}$, then $NotFunctClash^{\mathcal{T}}_P(t,t') = \neg(\exists y.P(t,y) \wedge y \neq t' \wedge ConsAt^{\mathcal{T}}_P(t,y)) \wedge \neg(\exists y.P(y,t') \wedge y \neq t \wedge ConsAt^{\mathcal{T}}_P(y,t))$.

(The function $NotFunctClash^{\mathcal{T}}_U$ for attributes U is defined analogously.)

We are now able to define for each *DL-Lite*$_A$ construct the formula that combines together the various formulas we have introduced for dealing with the various possible clashes: (i) $NotClash^{\mathcal{T}}_A(t) = NotDisjClash^{\mathcal{T}}_A(t)$ for an atomic concept name A and term t; (ii) $NotClash^{\mathcal{T}}_Z(t,t') = NotDisjClash^{\mathcal{T}}_Z(t,t') \wedge NotFunctClash^{\mathcal{T}}_Z(t,t')$ for a role or attribute name Z and terms t, t'.

Let q be a CQ $\exists x_1, \ldots, x_k. \bigwedge_{i=1}^n A_i(t_i^1) \wedge \bigwedge_{i=1}^m P_i(t_i^2, t_i^3) \wedge \bigwedge_{i=1}^\ell U_i(t_i^4, t_i^5)$, where every A_i is an atomic concept, every P_i is an atomic role, every U_i is an attribute, and every $t_i^1, t_i^2, t_i^3, t_i^4, t_i^5$ is either a constant or a variable x_j with $1 \leq j \leq k$. Then, we define $\mathit{IncRewriting}_{IAR}(q, \mathcal{T})$ as the following FOL sentence

$$\exists x_1, \ldots, x_k. \bigwedge_{i=1}^n A_i(t_i^1) \wedge ConsAt^{\mathcal{T}}_{A_i}(t_i^1) \wedge NotClash^{\mathcal{T}}_{A_i}(t_i^1) \wedge$$
$$\bigwedge_{i=1}^m P_i(t_i^2, t_i^3) \wedge ConsAt^{\mathcal{T}}_{P_i}(t_i^2, t_i^3) \wedge NotClash^{\mathcal{T}}_{P_i}(t_i^2, t_i^3)$$
$$\bigwedge_{i=1}^\ell U_i(t_i^4, t_i^5) \wedge ConsAt^{\mathcal{T}}_{U_i}(t_i^4, t_i^5) \wedge NotClash^{\mathcal{T}}_{U_i}(t_i^4, t_i^5)$$

Informally, for each atom $A_i(t_i^1)$, each membership assertion of the ABox $\mathcal{A}$ constituting an image of $A_i(t_i^1)$ has not to be inconsistent with the TBox (condition $ConsAt^{\mathcal{T}}_{A_i}(t_i^1)$), and has not to be involved in any clash with some other assertion of $\mathcal{A}$ on any negative inclusion (condition $NotClash^{\mathcal{T}}_{A_i}(t_i^1)$). Similarly for atoms of the form $P_i(t_i^2, t_i^3)$ and $U_i(t_i^4, t_i^5)$.

Let Q be the UCQ $q_1 \vee \ldots \vee q_n$. Then, we define $\mathit{IncRewritingUCQ}_{IAR}(Q, \mathcal{T}) = \bigvee_{i=1}^n \mathit{IncRewriting}_{IAR}(q_i, \mathcal{T})$. Finally, we define $\mathit{PerfectRef}_{IAR}(Q, \mathcal{T})$ as $\mathit{IncRewritingUCQ}_{IAR}(\mathit{PerfectRef}(Q, \mathcal{T}), \mathcal{T})$, where $\mathit{PerfectRef}(Q, \mathcal{T})$ denotes the algorithm for computing a perfect reformulation of a UCQ Q with respect to a *DL-Lite*$_A$ TBox $\mathcal{T}$ under standard semantics [2,7] (the algorithm $\mathit{PerfectRef}(Q, \mathcal{T})$ returns a UCQ specified over $\mathcal{T}$). It can be shown (see [6]) that $\mathit{PerfectRef}_{IAR}(Q, \mathcal{T})$ constitutes a perfect reformulation of Q with respect to $\mathcal{T}$ under IAR semantics.

Therefore, using this technique, it is possible to solve query answering under IAR semantics in *DL-Lite*$_A$ as follows. Given the initial query Q and the ontology $\langle \mathcal{T}, \mathcal{A} \rangle$, the first-order query $\mathit{PerfectRef}_{IAR}(Q, \mathcal{T})$ is computed, and then such a first-order query is evaluated over the original ABox (which is in general inconsistent with $\mathcal{T}$). So, in this case no repair of the ABox is performed, differently from the algorithm presented in the previous section.

4 Experiments

We have implemented the techniques presented in the previous Section in the Quonto system, in a module called QuID (the QUonto Inconsistent Data handler). Essentially, QuID is a Java implementation of the above algorithms for ABox repair and for query rewriting. In fact, in the Quonto architecture, the management of the ABox is delegated to a relational database management system (DBMS): therefore, all the operations on ABox assertions of the algorithms for computing repairs are executed in QuID by the DBMS used by Quonto, through appropriate SQL scripts.

We have experimented QuID in order to answer several open questions about: (i) the computational cost of the various steps of the ABox cleaning algorithm and of the query rewriting algorithm; (ii) the scalability of such algorithms; (iii) the impact of the "degree of inconsistency" of the ABox on the computational cost of the algorithms; (iv) the practical difference between the ABox cleaning tecnhique and the purely intensional rewriting technique.

Experimenting the QuID-IAR-repair algorithm. We have experimented our implementation of the QuID-IAR-repair algorithm over the LUBM benchmark ontology[2] whose TBox has 43 concept names, 25 role names, 7 attribute names, and about 200 TBox assertions. We have generated 4 different ABoxes by means of the UBA Data Generator provided by the LUBM website, with an increasing number of assertions, and used such ABoxes in our experiments. It is important to note that the original LUBM ontology has no axioms which can generate inconsistency, and hence, no inconsistent data is contained in the generated ABoxes. So, we sligthly modified the LUBM ontology by adding some "inconsistency-generating" axioms and then added inconsistencies to the ABoxes. We created four different version for every original ABox with different percentages of ABox assertions involved in minimal conflict sets, in order to get ABoxes with respectively $1\%, 5\%, 10\%$ and 20% of inconsistent assertions, uniformly distributed among the axioms which might generate inconsistency. Figure 1 shows the size (number of instance assertions) of the ABoxes we used in the experiments: every column is labeled with the number of Universities the ABox data contains, and every row is labeled with the percentage of inconsistent facts added to the ABox itself.

Figure 2 report some of the experimental results that we have obtained. The table displayed presents the experimental results for QuID-IAR-repair using a PostgreSQL 9.1 instance as external DBMS. The results have been conducted on a Pentium i5 (2.4 GHz) CPU with 4GB RAM under Windows 7 (64 bit) operating system.

All the necessary software, as well as instructions on how to reproduce the experiments presented in this section, are publicly available at `http://www.dis.uniroma1.it/~ruzzi/quid/`. Further details on the ontology used in the experiments are also available there.

In the table displayed in Figure 2, the first column reports the number of universities represented in the ABox, while the second column reports the percentage of ABox assertions that participate in minimal conflict sets for the considered TBox. Moreover:

- T1 denotes the time to create the annotated ABox (step 1 of QuID-IAR-repair);

[2] `http://swat.cse.lehigh.edu/projects/lubm/`

– T2 denotes the time to detect unary and binary conflict sets (steps 2 and 3 of QuID-IAR-repair);
– T3 denotes the time to extract the IAR-repair from the annotated ABox (step 4 of QuID-IAR-repair);
– Total is the total time to compute the IAR-repair, i.e., T1+T2+T3.

		Number of Universities			
		1	5	10	20
Inc. Perc.	1	103765	631960	1285244	2711216
	5	109165	658980	1339304	2819337
	10	115845	692400	1406124	2952957
	20	130445	765380	1552104	3244937

Fig. 1. Size of the UBA generated ABoxes

#Univ	Inc%	T1 (ms)	T2 (ms)	T3 (ms)	Total (ms)
1	1	66908	2356	73617	142881
	5	69748	11559	71401	152708
	10	71402	24523	70231	166156
	20	85878	50014	68156	204048
5	1	414477	13416	418970	846863
	5	419298	60434	414854	894586
	10	412371	131805	403619	947795
	20	466363	254000	406880	1127243
10	1	968123	31060	953037	1952220
	5	945471	140447	917890	2003808
	10	936688	271830	884835	2093353
	20	987216	573020	873664	2433900
20	1	2381829	137327	2379121	4898277
	5	2485267	353486	2251335	5090088
	10	2233066	722468	2212381	5167915
	20	2297791	1417200	2090794	5805785

Fig. 2. Repair generation time

The above experimental results show that:

(i) the computation of the IAR-repair (column T1) seems really scalable, and grows almost linearly w.r.t. the size of the ABox.
(ii) the percentage of inconsistency, i.e., the fraction of ABox assertions that participate in minimal conflict sets, has a real impact only on the detection of minimal conflict sets (column T2);
(iii) most of the whole execution time of the QuID-IAR-repair algorithm is devoted to the creation of annotated ABox (T1) and of the final repair (T3): if this could be avoided (e.g., by just modifying the original database, as explained below), the algorithm would be much more efficient, since only time T2 would be consumed.

Experimenting the Consistent Query Rewriting Approach. As above observed, most of the execution time of the algorithm QuID-IAR-repair using a disk-resident DB is due to the creation of the annotated ABox (step 1) and to the creation of the IAR-repair (step 4). Thus, avoiding these steps would dramatically improve the efficiency of this algorithm.

To this aim, we observe that both the above steps could be completely avoided if the database schema used for representing the ABox would present an additional attribute for storing annotations in every relation (the usual DB representation of an ABox uses a unary relation for every concept and a binary relation for every role). This corresponds to the idea of directly using an annotated ABox instead of a standard ABox in the system. In this case, the computation of the IAR-repair could only consist of steps 2 and 3 of the algorithm QuID-IAR-repair. However, the choice of using an annotated ABox instead of a standard ABox could affect query answering, since the queries evaluated on an annotated ABox should be able to only consider the assertions whose annotation is equal to *cons*. Similarly, exploiting the query rewriting technique presented in Section 3, it is possible to completely avoid the computation of the annotated ABox, and could be able to evaluate the first-order query corresponding to the perfect reformulation of the original query directly over the original, inconsistent, ABox.

We have experimented whether this choice is actually feasible. In particular, we tested and compared three different approaches: (IAR) evaluation of the IAR perfect reformulation over the inconsistent ABox; (ANN) evaluation over the annotated ABox $\mathcal{A}_{ann}$ (produced by the QuID-IAR-repair algorithm) of the original query enriched with suitable conditions that are needed to filter out the assertions belonging to minimal conflict sets; (REP) evaluation of the original query over the repair using the standard query answering technique of QuOnto. Figure 3 presents a table showing the evaluation time of nine of the fourteen queries of the LUBM benchmark over all the ABoxes previously considered. We adopted a timeout (denoted by T.o. in the table) of 1 hour.[3]

Comparing the Two Approaches. These experimental results show that, in QuID, evaluating queries on the annotated ABox is computationally not harder than evaluating them on the standard ABox. Conversely, the evaluation of the IAR perfect reformulations is often more expensive (in particular, it is more expensive for queries Q5–Q9). This is due to the fact that we have built no repair and we are querying the inconsisent ABox: thus, as shown in the previous section, the IAR perfect reformulation essentially has to select only assertions of the ABox which do not participate in minimal inconsistent sets (with respect to the TBox). This makes the form of such queries quite involved: in particular, the SQL queries corresponding to the IAR perfect refomulations of UCQs may present several nesting levels, which makes such queries hard to evaluate by current DBMSs. This consideration is enforced, e.g., by the evaluation time of query Q5, which is greater than 1 hour on the ABox representing 5 universities. That is, in this case the time to evaluate the IAR perfect reformulation of this query over the original ABox is much greater than computing the IAR repair and then evaluating the original query on the repaired ABox.

Combining the results of Figure 2 and Figure 3, it seems that, in general, the ABox cleaning approach is more convenient than the consistent query rewriting approach. In other words, the cost of preprocessing the ABox is generally an acceptable one, and really pays off during the evaluation of the queries, especially when the annotated representation of the ABox is adopted.

[3] Further details on our experiments can be found at http://www.dis.uniroma1.it/~ruzzi/quid/.

		Q1			Q2			Q3		
#Univ	Inc%	IAR	ANN	REP	IAR	ANN	REP	IAR	ANN	REP
1	1	2324	37	31	31	2	0	32	7	16
	5	2340	36	16	32	3	0	31	8	0
	10	2325	40	16	31	0	0	31	10	0
	20	2340	36	16	32	3	0	31	7	0
5	1	905	2393	2808	31	238	405	63	874	999
	5	874	2882	3135	16	162	297	62	964	936
	10	561	3826	2668	32	113	218	78	882	936
	20	942	2968	4259	31	423	390	63	1063	1435
10	1	6661	11659	9531	32	162	390	3510	2441	2434
	5	4306	10878	9610	32	355	281	2122	2111	1966
	10	5663	8928	6926	47	437	406	1856	1977	2074
	20	4540	7677	7425	62	367	281	2871	1694	1701
20	1	11170	13625	20748	32	210	375	1639	1060	3229
	5	9859	18356	21887	63	317	390	2028	2198	3323
	10	9844	16870	14883	47	365	249	2075	2605	2996
	20	8783	18347	15725	63	482	296	2496	1486	2402

		Q4			Q5			Q6		
#Univ	Inc%	IAR	ANN	REP	IAR	ANN	REP	IAR	ANN	REP
1	1	78	4	0	4898	17	15	297	7	16
	5	78	0	15	4899	20	16	312	10	0
	10	78	0	0	4696	10	15	312	0	0
	20	62	4	15	4727	18	16	312	7	0
5	1	125	446	453	T.o.	15998	17659	1529	35	47
	5	203	351	421	T.o.	23721	14914	1529	37	47
	10	187	348	561	T.o.	15243	16521	1560	31	31
	20	202	444	749	T.o.	19612	22963	1748	30	15
10	1	3588	220	1372	T.o.	54142	52434	2995	66	62
	5	889	912	1185	T.o.	41709	54600	3167	66	62
	10	1701	237	936	T.o.	50099	48875	3229	70	63
	20	1607	771	843	T.o.	35762	40138	3448	73	78
20	1	1965	1398	1794	T.o.	99222	127796	6396	157	156
	5	2106	1121	1435	T.o.	112814	132288	6536	152	141
	10	2262	1238	1357	T.o.	113969	110622	6739	157	156
	20	3900	1273	1544	T.o.	103710	110339	7332	145	140

		Q7			Q8			Q9		
#Univ	Inc%	IAR	ANN	REP	IAR	ANN	REP	IAR	ANN	REP
1	1	4539	37	31	499	8	16	219	3	0
	5	4695	40	47	515	0	0	187	0	15
	10	4586	40	31	500	10	16	171	0	0
	20	4571	38	47	406	9	0	140	2	0
5	1	4652	557	453	1716	43	31	172	3	31
	5	4664	575	343	1731	39	31	156	29	0
	10	4648	533	515	1732	57	31	156	9	47
	20	4665	828	671	1731	40	16	141	30	31
10	1	4901	799	593	4025	71	62	234	17	16
	5	4790	894	655	3479	80	63	219	20	31
	10	4695	813	686	3339	88	63	234	18	15
	20	4477	833	483	3354	85	63	156	13	15
20	1	4508	590	577	5444	160	141	296	15	15
	5	4509	898	733	5553	160	140	312	10	16
	10	4430	753	437	5974	162	141	297	21	15
	20	4508	839	437	12215	171	156	156	9	16

Fig. 3. Query answering time (in milliseconds) for the various techniques

On the other hand, it is worth recalling that the ABox cleaning approach might not always be possible or easily realizable in real applications, especially in ontology-based data access (OBDA) scenarios where the ABox is actually a virtual object that is defined through virtual queries/views over one or more remote databases: (see e.g., [7]): in these cases, the OBDA system can typically only read such databases.

5 Conclusions

In this paper we have presented a practical approach to automatic the repair of inconsistent ontologies. The key features of our approach are the following: (i) the semantics of the repair are simple, intuitive, formally grounded, and defined for all DLs; (ii) such semantics allow for tractable automatic ABox cleaning and consistent query rewriting in the case of OWL 2 QL ontologies; (iii) our experiments show that the approach is really scalable, and that very large ABoxes can be effectively repaired.

The work presented in this paper can be extended in several directions. First, the present implementation can be certainly further optimized. For instance, besided working with an annotated ABox representation, other optimizations are possible: one possibility which seems worth exploring is employing summarization techniques for ABox representation, as in [3]. Also, the consistent query rewriting technique can be certainly optimized to the aim of reducing the size of the reformulated query. Then, it would be very interesting to see whether the techniques presented in this paper can be extended to other tractable OWL profiles.

Acknowledgments. This research has been partially supported by the ICT Collaborative Project ACSI (Artifact-Centric Service Interoperation), funded by the EU under FP7 ICT Call 5, 2009.1.2, grant agreement n. FP7-257593.

References

1. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The *DL-Lite* family and relations. J. of Artificial Intelligence Research 36, 1–69 (2009)
2. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. J. of Automated Reasoning 39(3), 385–429 (2007)
3. Dolby, J., Fan, J., Fokoue, A., Kalyanpur, A., Kershenbaum, A., Ma, L., Murdock, J.W., Srinivas, K., Welty, C.A.: Scalable Cleanup of Information Extraction Data Using Ontologies. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 100–113. Springer, Heidelberg (2007)
4. Hogan, A., Harth, A., Passant, A., Decker, S., Polleres, A.: Weaving the pedantic web. In: Proc. of 3rd Int. Workshop on Linked Data on the Web, LDOW 2010 (2010)
5. Lembo, D., Lenzerini, M., Rosati, R., Ruzzi, M., Savo, D.F.: Inconsistency-Tolerant Semantics for Description Logics. In: Hitzler, P., Lukasiewicz, T. (eds.) RR 2010. LNCS, vol. 6333, pp. 103–117. Springer, Heidelberg (2010)

6. Lembo, D., Lenzerini, M., Rosati, R., Ruzzi, M., Savo, D.F.: Query Rewriting for Inconsistent DL-Lite Ontologies. In: Rudolph, S., Gutierrez, C. (eds.) RR 2011. LNCS, vol. 6902, pp. 155–169. Springer, Heidelberg (2011)
7. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking Data to Ontologies. In: Spaccapietra, S. (ed.) Journal on Data Semantics X. LNCS, vol. 4900, pp. 133–173. Springer, Heidelberg (2008)
8. Wang, Z., Wang, K., Topor, R.W.: A new approach to knowledge base revision in *DL-Lite*. In: Proc. of AAAI 2010, AAAI Press (2010)

Evaluating Entity Summarization Using a Game-Based Ground Truth

Andreas Thalhammer[1], Magnus Knuth[2], and Harald Sack[2]

[1] University of Innsbruck, Technikerstr. 21a, A-6020 Innsbruck
andreas.thalhammer@sti2.at
[2] Hasso Plattner Institute Potsdam, Prof.-Dr.-Helmert-Str. 2–3, D-14482 Potsdam
{magnus.knuth,harald.sack}@hpi.uni-potsdam.de

Abstract. In recent years, strategies for Linked Data consumption have caught attention in Semantic Web research. For direct consumption by users, Linked Data mashups, interfaces, and visualizations have become a popular research area. Many approaches in this field aim to make Linked Data interaction more user friendly to improve its accessibility for non-technical users. A subtask for Linked Data interfaces is to present entities and their properties in a concise form. In general, these summaries take individual attributes and sometimes user contexts and preferences into account. But the objective evaluation of the quality of such summaries is an expensive task. In this paper we introduce a game-based approach aiming to establish a ground truth for the evaluation of entity summarization. We exemplify the applicability of the approach by evaluating two recent summarization approaches.

Keywords: entity summarization, property ranking, evaluation, linked data, games with a purpose.

1 Introduction

The main idea of the Semantic Web is to make implicit knowledge explicit and machine processable. However, machines that process knowledge are not a dead end. In fact, after processing the returned results are either consumed by another machine or by human users. In this paper, we focus on the latter: the consumption of machine processed data by human users. A lot of efforts in the Semantic Web currently focus on Linked Data interfaces and Linked Data visualization. As for the former, most interfaces have been developed by the Linked Data community and usually show all information (usually as property-value pairs) that is available for an entity (e. g. Pubby[1], Ontowiki[2], etc.) and leave it to the user to decide which of the information is important or of interest. In May 2012, Google[3] introduced its "Knowledge Graph" (GKG), which produces summaries for Linked Data entities. While it is not the first approach to rank properties or

[1] Pubby – http://www4.wiwiss.fu-berlin.de/pubby/
[2] Ontowiki – http://ontowiki.net/
[3] Google – http://google.com/

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 350–361, 2012.

features of Linked Open Data according to their relevance [9,11,3] the uptake by industry certainly gives incentives for further investigation in this subject. This has to be considered in line with the fact that Google processed 87.8 billion queries in December 2009 [4] which makes roughly 2.8 billion queries per day. Keeping the huge number of daily searches in mind, it was an interesting move by Google to devote a big part of its result pages to the GKG summaries. Having an average of 192 facts attached to an entity [3], producing a concise summary that is shaped to an entity's individual characteristics states an interesting research problem.

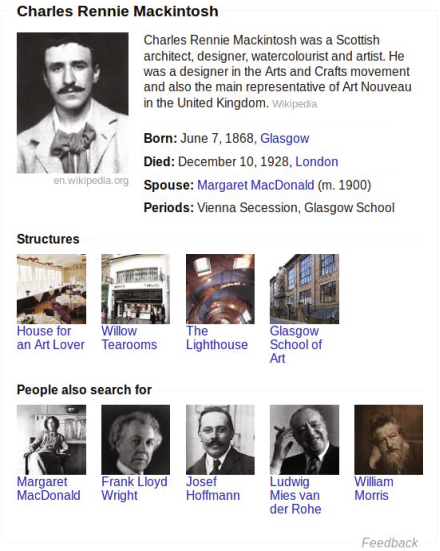
In this paper we will discuss current developments in Linked Data entity summarization and fact ranking as well as the need for a gold standard in form of a reference dataset which makes evaluation results comparable. We introduce a novel application of games with a purpose (GWAPs) that enables us to produce a gold standard for the evaluation of entity summarization. We demonstrate the applicability of the derived data by evaluating two different systems that utilize user data for producing summaries (one of which is GKG). In the course of our explanations we will emphasize on the complete and correct description of our test settings and stress that all data (that does not violate the privacy of our users) is made publicly available.

The remainder of this paper is structured as follows: Section 2 gives a description of the state-of-the-art in Linked Data entity summarization including the Google Knowledge Graph. In Section 3 the processed data sets, the quiz game and the evaluated systems are explained in detail, while Section 4 reports the achieved results. Section 5 concludes the paper with a brief summary and an outlook on future work.

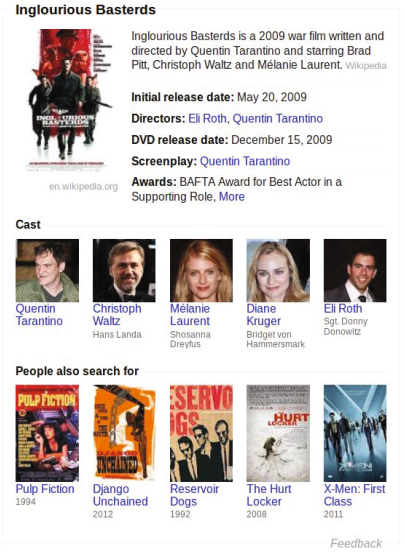
2 Background

In recent years, four approaches to Linked Data entity summarization have emerged including the one adopted by GKG. In the following, we will discuss all of those approaches and - in addition - present methods used for evaluating text summarization.

Google has introduced the "Knowledge Graph" in May 2012 [8]. The main idea is to enrich search results with information about named entities. In case of ambiguous queries, such as "lion king" (currently a musical and a film are returned), Google lists also different possibilities. Two examples for GKG summaries are shown in Fig. 1. Google's summaries are usually structured as follows: After presenting the name of the entity and an attached plot (usually taken from Wikipedia) next to a picture, up to five "main facts" are listed. These facts differ heavily between entities of different RDF types but also – to a certain extent – between entities of the same RDF type. After that, for certain RDF types like architects or movies, domain-specific attributes such as *'Structures'* (architects) or *'Cast'* (movies) are presented. For those, Google also defines a ranking e. g. from left to right for the *'Cast'* lists. In addition, a range of related entities is displayed (Google introduces this list with *'People also search for'*). In their blog, Google



(a) GKG: architect and designer Charles Rennie Mackintosh.



(b) GKG: movie titled "Inglourious Basterds".

Fig. 1. Examples for GKG summaries (Source: http://google.com/)

developers describe summaries as one of "three main ways" to enhance search results with GKG information [8]. To automatically generate summaries, Google utilizes the data of their users, i. e. the queries, "[...] and study in aggregate what they've been asking Google about each item" [8]. We assume that these queries are in most cases "subject+predicate" queries, such as "lake garda depth", or "subject+object" queries such as "the shining stanley kubrick". In some cases also "subject+predicate+object" queries might make sense such as "jk rowling write harry potter"[4]. It is worth mentioning that using queries for determining the users' average interest in facts also has some pitfalls. For example, the query "inglourious basterds quentin tarantino" (querying for a movie and one of its directors) not only boosts the 'directed by' property but also the 'starring' property for the movie's relation to the person Quentin Tarantino. Unfortunately, this leads to the situation that the main actor (namely Brad Pitt) is not mentioned in the cast list while the director – who is known for taking minor roles in his movies and is doing so in this particular one – takes his position (see Fig. 1b).

Thalhammer et al. [9] explain how entity neighborhoods, derived by mining usage data, may help to discover relevant features of movie entities. The authors outline their idea that implicit or explicit feedback by users, provided by consuming or rating entities, may help to discover important semantic relationships between entities. Having established the neighborhood of an entity with

[4] In fact, this query was suggested by Google Instant (http://www.google.com/insidesearch/features/instant/about.html).

methods adopted from item-based collaborative filtering [7], the frequency of a feature that is shared with its neighbors is likely to give an indication about the feature's importance for the entity. A TF-IDF-related weighting scheme is also adopted as some features are generally very common (e. g., provenance statements). Unfortunately, the authors do not provide an evaluation of their system and only provide some preliminary results. In the later sections, we will refer to this approach as UBES (usage-based entity summarization).

The term of "entity summarization" was initially introduced by [3]. According to the authors, entity summarization is the task of identifying features that "not just represent the main themes of the original data, but rather, can best identify the underlying entity" [3]. We do not fully agree with this definition. Rather than selecting features that unambiguously identify an entity, we suggest to select features that are most interesting to present to a user. Of course, for many entities there is a significant overlap between the features that best identify an entity and features that are most interesting for the users. As a further contribution, the authors introduce the term "feature" as a property-value pair. The approach presented in [3] applies a "goal directed surfer" which is an adapted version of the random surfer model that is also used in the PageRank algorithm. The main idea is to combine informativeness and relatedness for the ranking of features. In the conclusion of [3], the authors state that "user-specific notion of informativeness [...] could be implemented by leveraging user profiles or feedback" in order to mitigate the problem of presenting summaries that help domain experts but are not as useful for average users. The presented approach does not utilize user or usage data in order to provide summaries. However, this information could be given implicitly by the frequency of in and out links.

Waitelonis and Sack explain how exploratory search can be realized by applying heuristics that suggest related entities [11]. Assume that a user is currently browsing the current US president's Linked Open Data description. Attached to the president's URI are properties such as `dbpedia-owl:residence`, `dbpprop:predecessor`, or `dbpedia-owl:party`. Obviously, these links are useful to show in the context of exploratory search. However, as there are more than 200 facts attached to the entity, the authors propose to filter out less important associations (i. e., provide summaries). To achieve this, they propose and evaluate eleven different heuristics and various selected combinations for ranking properties. These heuristics rely on patterns that are inherent to the graph, i. e. they do not consider usage or user data. The authors conduct a quantitative evaluation in order to find out which heuristic or combination performs best. The results show that some heuristics, such as the Wikilink and Backlink-based ones, provide high recall while Frequency and Same-RDF-type-based heuristics enable high precision. Trials with blending also showed that either precision or recall can be kept at a significant high level, but not both at the same time. Like in the approach of GKG, the predicate and the object are decoupled. While the introduced heuristics address the predicates, the data gathering for the evaluation focuses on the objects. As exemplified above, this leaves space for ambiguity. In the discussion, the authors argue that summaries should be considered in

a specific context (i. e., "what is the search task?") and therefore quantitative measures might not provide the right means to evaluate property rankings.

[3] and [11] provide evaluations of their approaches. Both provide a quantitative as well as a qualitative evaluation. In the quantitative evaluation, both approaches base their evaluation on DBpedia[5] excerpts comprised of 115 [11] and 149 [3] entities. These entities were given to a sufficient amount of users in order to establish a ground truth with human created summaries. To the best of our knowledge, the results of these efforts are not publicly available.

In the field of automatic text summarization, [1] discusses two possible ways for evaluating summaries: *human assessments* and *proximity to a gold standard*. Thus, in this area, not only a gold standard had to be created but also a way to measure closeness to such a reference. As entity summarization deals with structured data only, such proximity measures are not needed: to measure the similarity between a summary and a ground truth, we can make use of classic information retrieval methods such as precision/recall, Kendall's τ and Spearman's rank correlation coefficient.

3 Evaluating Entity Summarization

We attempt to create a ground truth for the task of entity summarization by utilizing data gained from a game with a purpose. We exemplify our approach in the domain of movies. Thus, our research hypothesies is as follows:

> *A game-based ground truth is suitable for evaluating the performance of summarization approaches in the movie domain.*

Our assumption is that implemented approaches that provide summaries should perform significantly better than randomly generated summaries when measuring the correlation to the established ground truth. It is important to note that the relevance of facts for the task of summarization will be evaluated on the entity level. This means that the same properties, objects, or even property-value pairs are of different importance for different subjects. As a matter of fact, the importance of facts for an entity might vary given different contexts and summarization purposes. However, summarization also involves a certain level of pragmatics, i. e. trying to capture the common sense to address as many users as possible.

In the following we detail the restraints for the chosen domain, the design of the quiz game, the interpretation of the gained data, and the experimental setup for the evaluated systems.

3.1 Employed Dataset

In our evaluation, we focus on movie entities taken from Freebase[6]. This dataset contains a large amount of openly available data and – in contrast to DBpedia

[5] DBpedia - http://dbpedia.org/
[6] Freebase – http://www.freebase.com/

Listing 1. Property chain for defining a "hasActor" property.

```
<http://some-name.space/hasActor>
<http://www.w3.org/2002/07/owl#propertyChainAxiom> (
<http://rdf.freebase.com/ns/film.film.starring>
<http://rdf.freebase.com/ns/film.performance.actor> ).
```

and the Linked Movie Database (LinkedMDB)[7] – very detailed and well curated information. Large parts of this dataset are also used by Google for its summaries [8]. For the evaluation, we have randomly selected 60 movies of the IMDb Top 250 movies[8] and derived the Freebase identifiers by querying Freebase for the property `imdb_id`. With facts about 250 movies, it is difficult to achieve the mandatory number of game participants for sufficient coverage. Therefore, we have restricted the number of movies to 60. We have downloaded RDF descriptions of the movies and stored them in an OWLIM[9] triple store with OWL2 RL[10] reasoning enabled. This enables us to connect properties (such as actors) that are linked via reification (such as the 'film-actor-role' relationship) directly with property chain reasoning. An example for creating such an axiom is provided in Listing 1. We have created such direct links for actors, role names, achieved awards, budgets, and running times. As a matter of fact, not all properties are useful to be questioned in a game. Therefore, we make use of a white list. The list of selected movies, the used property chain rules as well as the property white list are available online (cf. Sec. 4.3).

3.2 *WhoKnows?Movies!* – Concept and Realization

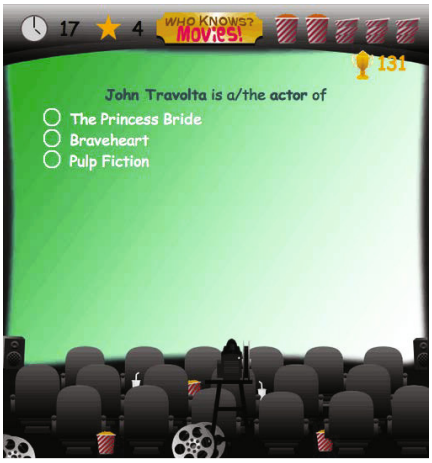
We developed *WhoKnows?Movies!* [10], an online quiz game in the style of *'Who Wants to Be a Millionaire?'*, to obtain a ground truth for the relevance of facts. The principle of the game is to present multiple choice questions to the player that have been generated out of the respective facts about a number of entities. In this case we limited the dataset as described in Sec. 3.1. The players can score points by answering the question correctly within a limited period of time and lose points and lives when giving no or wrong answers.

As an example, Fig. 2 shows the question *'John Travolta is the actor of ...?'* with the expected answer *'Pulp Fiction'*, which originates from the triple

```
fb:en.pulp_fiction test:hasActor fb:en.john_travolta .
```

and is composed by turning the triple's order upside down: *'Object is the property of: subject1, subject2, subject3...'*. The remaining options are selected from entities that apply the same property at least once, but are not linked to the object of the question. In this way we assure that only wrong answers are presented as alternative choices. There are two variants of questions: *One-To-One*

[7] LinkedMDB – `http://www.linkedmdb.org/`
[8] IMDB Top 250 – `http://www.imdb.com/chart/top`
[9] OWLIM – `http://www.ontotext.com/owlim`
[10] OWL2 RL – `http://www.w3.org/TR/owl2-profiles/#OWL_2_RL`



Subject	Property	Object
Pulp Fiction	**actor**	**John Travolta**
	actor	Uma Thurman
	actor	...
Braveheart	actor	Mel Gibson
	actor	Sophie Marceau
	actor	...
The Princess Bride	actor	Robin Wright
	actor	Annie Dyson
	actor	...

Fig. 2. Screenshot and triples used to generate a One-To-One question

where exactly one answer is correct and *One-To-N* where one or more answers are correct.

When the player answers a question correctly he scores points and steps one level up, while incorrect answer will be penalized by loosing points and one live. The earned score depends on the correctness of the answer and the time needed for giving the answer. With growing level the number of options raises, so correct answers are getting harder to guess. It has to be noted that the probability for a fact to appear in a question with many or few choices is equal for all facts. This ensures that the result is not skewed, for example by putting some facts in questions with two choices only. When submitting an answer, the user receives immediate feedback about the correctness of his answer in the result panel, where all choices are shown once again and the expected answer is highlighted. Given answers will be logged for later traceability and the triple's statistics are updated accordingly. The game finishes when the player lost all of his five lives.

Applying the white list described in Sec. 3.1, 2,829 distinct triples were produced in total. For each triple a set of wrong answers is preprocessed and stored into a database. When generating a question for a specific triple, a number of false subjects is randomly selected from this set.

3.3 What Are *Interesting Facts*?

The answer patterns of quiz games can tell a lot about what is generally interesting about an entity and what is not. One of the questions in the quiz game of Sec. 3.2 is *'What is the prequel of Star Wars Episode VI?'* with one of the answer options being *'Star Wars Episode V'*. Of course, most of the players were right on this question. On the other hand fewer players were right on the question whether *'Hannibal rising'* is a prequel of *'The silence of the lambs'*. The idea of

a good general[11] summary is to show facts that are common sense but not too common. This is related to Luhn's ideas about "significance" of words and sentences for the task of automatically creating literature abstracts [6]. Transferring the idea about "resolving power of words" to the answer patterns of the quiz game, we can state that neither the most known nor the most unknown facts are relevant for a good summary, it is the part between those two. Unfortunately, we have not been able to accumulate enough data to provide a good estimation for fine grained upper and lower cut-off levels. Therefore, in Sec. 4 we measure the relevance correlation with a pure top-down ranking.

In addition, there might be questions, where not knowing the right answer for a given fact does not necessarily mean that this fact does not have any importance. For our movie quiz game, participants are also asked for actors of a given movie. First of all, Freebase data does not distinguish between main actors and supporting actors. Thus, the property actor might not be in general considered as an important property, because most people do not know many of the supporting actors. Furthermore, an actor might play a very important role in a movie, but the game players do not know his name, because they only remember the face of the actor from the movie. The same holds for music played in the movie, where the participants might not know the title but are familiar with the tune. Thus, for future use, also the use of multimedia data should be considered to support the text-based questions of the quiz game.

3.4 Evaluated Systems

We exemplify the introduced evaluation approach to the summaries produced by GKG [8] and UBES [9]. For both approaches the additional background data stems from user behavior or actions. In addition, the rationale of both systems is to present useful information to the end users in a concise way. These similarities guarantee a comparison on a fairly equal level. In this section, we will detail the experimental setup and the data acquisition[12].

Usage-Based Entity Summarization (UBES)
In addition to Freebase, the UBES system utilizes the usage data of the HetRec2011 MovieLens2k dataset [2]. With a simple heuristic based on IMDb identifiers, more than 10,000 out of 10,197 HetRec2011 movies have been matched to Freebase identifiers (cf. [9] for more information). Based on the rating data provided by HetRec2011, the 20 nearest neighbors for each of the 60 selected movies were derived with the help of the Apache Mahout[13] library. It has to be noted that the actual numerical ratings were not used due to utilization of the log-likelihood similarity score [5]. This similarity measure only uses binary information (i. e., rated and not rated). With two SPARQL queries per movie, the

[11] As opposed to contextualized and/or personalized.
[12] The final results of the UBES and GKG summaries, both using Freebase URIs, can be found in the dataset, cf. Sec. 4.3.
[13] Apache Mahout – http://mahout.apache.org/

number of shared features was estimated once in combination with the neighbors and once considering the whole dataset. These numbers enable to apply the TF-IDF-related weighting for each property as it is described in [9]. Finally, the output has been filtered with the white list described in Sec. 3.1 in order to fit with the properties of the game and GKG.

Google's Knowledge Graph (GKG) Summaries
The 60 movie summaries by Google have been processed in a semi-automatic way to fit with the Freebase URIs. The first step was to retrieve the summaries of all 60 movies and storing the according HTML files. While the Freebase URIs for properties such as "Director" had to be entered manually, most objects could be linked to Freebase automatically. For this, we made use of the GKG-Freebase link[14]. The ranking of the five main facts is to be interpreted in a top-down order while Google's ordering of 'Cast' members follows a left to right orientation.

4 Results

At present, our quiz has been played 690 times by 217 players, while some players have played more frequently and the majority of 135 players has played only once. All 2,829 triples have been played at least once, 2,314 triples at least three times. In total 8,308 questions have been replied of which 4,716 have been answered correctly. The current results have to be regarded with care, since the absence of multiple opinions about a portion of the facts increases the probability for outliers. The random summaries were generated in accordance to the white list (cf. Sec. 3.1). In order to gain real randomness, we averaged the scores of 100 randomly generated summaries.

The ratio of correctly answered questions varies depending on the property that has been used in the question. As shown in table 1, to determine a movie according to its *prequel*, *film series*, or *sequel* is rather obvious, whereas a *film festival* or *film casting director* does not give a clear idea of the movie in question.

4.1 Evaluation of Property Ranking

To evaluate the ranking of properties for a single movie, we have determined the ranking of properties according to the *correct answer ratio*. The GKG movie representation lists general facts in an ordered manner, whereas the cast of the movie is displayed separately. Accordingly, only the remaining 24 properties are used for this evaluation. Properties that do not occur in the systems' results are jointly put in the bottom position. For benchmarking the ordering of both summaries, Kendall rank correlation coefficient is applied. For each movie τ is determined over the set of its properties. Table 2 shows the average, minimum, and maximum findings of τ. It can be seen, that both systems as well as random

[14] http://lists.w3.org/Archives/Public/semantic-web/2012Jun/0028.html

Table 1. Overall Relevance Ranking for Movie Properties

Rank	Property	Correct	Rank	Property	Correct
1	prequel	95.39%	14	production company	56.10%
2	film series	95.16%	15	runtime	54.52%
3	sequel	85.33%	16	music	54.11%
4	parodied	76.47%	17	award	53.41%
5	adapted original	74.32%	18	actor	52.86%
6	subject	73.91%	19	story writer	51.18%
7	genre	65.14%	20	editor	50.00%
8	initial release date	65.14%	21	event	50.00%
9	director	63.51%	22	cinematographer	44.20%
10	rating	61.61%	23	budget	42.78%
11	writer	61.61%	24	film festival	42.27%
12	featured song	60.00%	25	film casting director	41.32%
13	featured filming location	60.00%			

Table 2. Performance for Movie Property Ranking for Selected Movies

	τ_{avg}	τ_{min}	τ_{max}
UBES	0.045	-0.505 (The Sixth Sense)	0.477 (Reservoir Dogs)
GKG	0.027	-0.417 (The Big Lebowski)	0.480 (Reservoir Dogs)
Random	0.031	-0.094 (American Beauty)	0.276 (Monsters Inc)

perform equal in average. In each system, for about half of the movies the correlation is negative which means that the orderings are partly reverse compared ordering in the derived dataset. In general, none of the two systems' rankings differs significantly from a random ranking. This might be due to the sparsity of the dataset where most of the facts have been played only three times or less. Another negative influence might come from the fact that we aggregate on objects as we rank properties only and do not consider full property-value pairs.

4.2 Evaluation of Feature Ranking

For this evaluation the relevance ranking of the movie cast is compared to the user generated ground truth. Table 3 presents the average, minimum, and maximum findings of τ for the ranking of actors for a distinct movie. The results for the actor ranking are fairly equal for both systems in the average case. The average τ value differs from random scores. We have estimated that the difference to the random ranking is significant ($p < 0.05$) for both systems. This result provides an indication that the relative importance of property-value pairs can be captured by the statistics established through the game. It has to be mentioned, that - in some cases - the UBES heuristic provides none or very few proposals due to the required *'Cast'* overlap to neighboring movies.

Table 3. Performance for Actor Ranking for Selected Movies

	τ_{avg}	τ_{min}	τ_{max}
UBES	0.121	-0.405 (The Princess Bride)	0.602 (Indiana Jones and the last Crusade)
GKG	0.124	-0.479 (The Princess Bride)	0.744 (The Matrix)
Random	0.013	-0.069 (Fargo)	0.094 (Good Will Hunting)

4.3 Published Dataset

By publishing the data collected within the game[15], we encourage other researchers to apply this information for their purposes. The dataset consists of two main parts: first the aggregated statistics, which comprises the selected RDF triples and the respective players' performance. And second an anonymized log about the completed games that allows replay of user sessions with complete questions and results. Updates of these files will be published on a regular basis.

5 Conclusion and Future Work

In this paper a crowd sourcing approach implemented as a game with a purpose is demonstrated to gather relevance information about facts within a knowledge base and to establish ground truth data for evaluating summarization. We found indications that such a dataset can fulfill this purpose. However, the established dataset in its current state is too sparse to make valid assumptions about the importance of single facts.

Future development of the *WhoKnows?Movies!* game will also include images to help players to identify persons related to a movie, or other composed information artifacts. We also consider scoring properties that were listed in combination with an incorrect object while the user did not vote for this answer possibility. This is due to the fact that the user probably could exclude this possibility as he knew the correct object(s). Further research directions are increasing the number of movies and exploiting further domains. As for the latter, we consider the domains of books, music, places, and people. In principle, any domain where general knowledge is widely spread can be targeted with the game.

Acknowledgements. The authors would like to thank Ontotext AD for providing OWLIM-SE 5.0. This research was partly funded by the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 257790 (RENDER project).

[15] The dataset is available at http://yovisto.com/labs/iswc2012/

References

1. Amigó, E., Gonzalo, J., Peñas, A., Verdejo, F.: Qarla: a framework for the evaluation of text summarization systems. In: Proc. of the 43rd Annual Meeting on Association for Computational Linguistics, ACL 2005, pp. 280–289. Association for Computational Linguistics, Stroudsburg (2005)
2. Cantador, I., Brusilovsky, P., Kuflik, T.: 2nd ws. on information heterogeneity and fusion in recommender systems (hetrec 2011). In: Proc. of 5th ACM Conf. on Recommender systems, RecSys 2011. ACM, New York (2011)
3. Cheng, G., Tran, T., Qu, Y.: RELIN: Relatedness and Informativeness-Based Centrality for Entity Summarization. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 114–129. Springer, Heidelberg (2011)
4. comScore. comscore reports global search market growth of 46 percent in 2009 (2010), http://www.comscore.com/Press_Events/Press_Releases/2010/1/Global_Search_Market_Grows_46_Percent_in_2009
5. Dunning, T.: Accurate methods for the statistics of surprise and coincidence. Computational Linguistics 19(1), 61–74 (1993)
6. Luhn, H.P.: The automatic creation of literature abstracts. IBM J. Res. Dev. 2(2), 159–165 (1958)
7. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proc. of the 10th Int. Conf. on World Wide Web, WWW 2001, pp. 285–295. ACM, New York (2001)
8. Singhal, A.: Introducing the knowledge graph: things, not strings (2012), http://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html
9. Thalhammer, A., Toma, I., Roa-Valverde, A.J., Fensel, D.: Leveraging usage data for linked data movie entity summarization. In: Proc. of the 2nd Int. Ws. on Usage Analysis and the Web of Data (USEWOD 2012) co-located with WWW 2012, Lyon, France, vol. abs/1204.2718 (2012)
10. Waitelonis, J., Ludwig, N., Knuth, M., Sack, H.: WhoKnows? – evaluating linked data heuristics with a quiz that cleans up DBpedia. Int. Journal of Interactive Technology and Smart Education (ITSE) 8(3), 236–248 (2011)
11. Waitelonis, J., Sack, H.: Towards exploratory video search using linked data. Multimedia Tools and Applications 59, 645–672 (2012), 10.1007/s11042-011-0733-1

Evaluation of a Layered Approach to Question Answering over Linked Data

Sebastian Walter[1], Christina Unger[1], Philipp Cimiano[1], and Daniel Bär[2,⋆]

[1] CITEC, Bielefeld University, Germany
http://www.sc.cit-ec.uni-bielefeld.de/
[2] Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technische Universität Darmstadt
http://www.ukp.tu-darmstadt.de/

Abstract. We present a question answering system architecture which processes natural language questions in a pipeline consisting of five steps: i) question parsing and query template generation, ii) lookup in an inverted index, iii) string similarity computation, iv) lookup in a lexical database in order to find synonyms, and v) semantic similarity computation. These steps are ordered with respect to their computational effort, following the idea of layered processing: questions are passed on along the pipeline only if they cannot be answered on the basis of earlier processing steps, thereby invoking computationally expensive operations only for complex queries that require them. In this paper we present an evaluation of the system on the dataset provided by the 2nd Open Challenge on Question Answering over Linked Data (QALD-2). The main, novel contribution is a systematic empirical investigation of the impact of the single processing components on the overall performance of question answering over linked data.

Keywords: question answering, linked data, layered approach, experimental evaluation.

1 Introduction

Question answering over linked data has recently emerged as an important paradigm allowing non-expert users to access the steadily growing amount of data available as linked data (see [9] for a recent overview). One of the main challenges in question answering over linked data is mapping natural language questions into appropriate SPARQL queries or graph patterns that yield an appropriate and correct answer when evaluated. A crucial subtask to this end is to map words in the query to appropriate URIs representing their meaning. For example, when interpreting the question When was Abraham Lincoln born? with respect to the DBpedia dataset, the name Abraham Lincoln needs to be mapped

⋆ Part of this work has been supported by the Klaus Tschira Foundation under project No. 00.133.2008.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 362–374, 2012.

to the resource <http://dbpedia.org/resource/Abraham_Lincoln>, and born needs to be mapped to <http://dbpedia.org/ontology/birthplace>.

In this paper, we present a layered approach to question answering over linked data. The main intuition underlying this layered approach is the idea that a question answering system should be sensitive to the complexity of the question, in the sense that it applies certain processing steps only if the question cannot be answered using simpler mechanisms.

To give an idea of the various levels of difficulty, consider the following three questions taken from the DBpedia training questions of the 2nd Open Challenge on Question Answering over Linked Data (QALD-2, see Section 3.1 below):[1]

1. (a) What is the currency of the Czech Republic?
   (b)
```
SELECT DISTINCT ?uri WHERE {
   res:Czech_Republic dbo:currency ?uri .
}
```
2. (a) Who was the wife of U.S. president Lincoln?
   (b)
```
SELECT DISTINCT ?uri WHERE {
   res:Abraham_Lincoln dbo:spouse ?uri.
}
```
3. (a) Was Natalie Portman born in the United States?
   (b)
```
ASK WHERE {
   res:Natalie_Portman dbo:birthPlace ?city .
   ?city dbo:country res:United_States .
}
```

Question 1a exemplifies the simplest case: All natural language expressions can be mapped to DBpedia resources in the target SPARQL query 1b by simply matching the expressions (currency and Czech Republic) with the resources' labels, in this case `currency` and `Czech Republic`. Furthermore, the resources are directly related, so that the SPARQL query consists of one triple relating the entity `Czech Republic` with its currency. This is also the case for query 2b: The entity `Abraham Lincoln` is directly connected to his wife. However, matching the expressions used in the question 2a with DBpedia concepts (U.S. president Lincoln with `Abraham Lincoln`, and wife with `spouse`) is not straightforward but requires searching for synonyms and lexical variants. Similarly in example 3, where the natural language term born needs to be matched with the ontology label `birth place`. Moreover, the property `birth place` does not directly connect the occurring entities, Natalie Portman and the United States; instead they are connected via an intermediate node that is not expressed in the natural language question, the SPARQL query thus has a more complex structure.

As main contribution, we present the results of a systematic evaluation of the contribution of different state-of-the-art processing components on the overall system. For this purpose, we use the benchmarking datasets provided by the QALD-2 challenge. A systematic evaluation of the impact of various components

[1] The following prefixes are used:
```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX res: <http://dbpedia.org/resource/>
```

on the task has so far not been provided. This layered approach also allows us to assess the complexity of the questions in the dataset in terms of which processing is required to actually find an answer. We report the results in Section 3 below. Furthermore, we compare our system to the question answering systems that participated in the QALD-2 challenge as well as to Wolfram Alpha[2].

The paper is structured as follows: In Section 2 we present the architecture of our system in detail. In Section 3 we report on experiments on the QALD-2 dataset, presenting our results in terms of standard precision, recall and F-measure figures for each processing layer, thus being able to quantify the impact in terms of effectiveness of each layer. We also report the average times the approach requires to answer questions depending on the processing depth. This allows for a discussion of the trade-off between effectiveness and efficiency. Finally, we compare our approach to related work in Section 4, before concluding in Section 5.

2 Layered Approach

The system we propose, BELA, takes a natural language question as input and produces a SPARQL query as well as corresponding answers as output. It is layered in the sense that it builds on a pipeline along which hypotheses for the meaning of a natural language question are iteratively added and refined by factoring in more and more expensive processing mechanisms. At each layer, the best hypotheses is determined. If the confidence of the system in the hypothesis is high enough and the constructed query actually produces answers, the processing stops and the answers are returned.

BELA processes an incoming natural language question along the following layers:

1 Parsing and template generation
2 Inverted index lookup
3 String similarity computation
4 Lexical expansion
5 Semantic similarity computation

We will describe each of them in more detail in the following sections.

2.1 Parsing and Template Generation

Each input question is parsed on the basis of its part-of-speech tags, employing a parser based on Lexical Tree Adjoining Grammars (LTAG), in order to produce several query templates for the question. The parser has been described in more detail in [13], so we limit our description to the output of the parser. Parsing a natural language question produces a set of SPARQL query templates corresponding to proto-interpretations of this question, which mirror the semantic structure of the questions and only leave open slots where appropriate URIs need to be inserted. An example is given in 4.

[2] http://www.wolframalpha.com

4. (a) What is the currency of the Czech Republic?
   (b) `SELECT ?y WHERE {`
   `?y -- ?p -- ?x`
   `}`
   Slots:
   – ⟨`?p`, unknown, currency⟩
   – ⟨`?x`, resource, Czech Republic⟩

For the question in 4a the template in 4b is constructed. It specifies the overall structure of the query, but leaves open slots for a resource expressed as Czech Republic, which is related to `?y` by means of some property denoted by the noun currency. The dashes indicate that it is left open whether `?y` is subject or object of the property, i.e. whether the triple is `?y ?p ?x .` or `?x ?p ?y .`

2.2 Index Lookup

Consider the example 4 above. The first step of processing this template consists in a simple lookup of all slot terms in an inverted index. For indexation, we extract all concepts from DBpedia 3.7 subsumed by the `ontology` and `property` namespace together with their `rdfs:label`, from which we build an inverted index that maps each label to a set of URIs. Additionally, we include Wikipedia re-directs, such that a range of labels, e.g. IBM, I.B.M., International Business Machine and IBM Corporation map to the same URI, in this case `<http://dbpedia.org/resource/IBM>`. The resulting index contains more than 8 million entries: 8,011,004 mappings of labels to resources, 785 mappings of labels to classes, and 92,910 mappings of labels to properties (3,362 from the `ontology` and 89,548 from the `property` namespace).

For the example in 4, the slot terms currency and Czech Republic are found in the index, therefore the following two hypotheses about possible instantiattion of the query slots with URIs are built:

5. – Slot: ⟨`?p`, property, currency⟩
   – URI: `<http://dbpedia.org/ontology/currency>`
   – Rank: 1.0
6. – Slot: ⟨`?x`, resource, Czech Republic⟩
   – URI: `<http://dbpedia.org/resource/Czech_Republic>`
   – Rank: 1.0

The rank is a confidence value between 0 and 1. Here the rank is set to 1, as we take a single direct match in the index to be a sure indicator for a successful mapping. If several mappings are found, a hypothesis for each of them is created (leaving disambiguation to the success or failure of these hypotheses). Also note that the previously unknown type of `?p` can now be specified as property, the type of the found URI.

Using the above hypotheses to instantiate the SPARQL template yields the following two alternative interpretations of the question corresponding to the interpretations of `?y` is subject or object, respectively:

7. (a)
```
SELECT ?y WHERE {
   ?y <http://dbpedia.org/ontology/currency>
   <http://dbpedia.org/resource/Czech_Republic> .
}
```
   (b)
```
SELECT ?y WHERE {
   <http://dbpedia.org/resource/Czech_Republic>
   <http://dbpedia.org/ontology/currency> ?y .
}
```

All generated queries are then sent to the SPARQL endpoint and the highest ranked query that actually returns an answers is selected as final output.[3] In our example case, the query in 7b does return an answers and thus seems to represent a valid interpretation of the natural language question.

In case none of the queries returns an answer, BELA proceeds with the next step.

2.3 String Similarity

In case that the basic mechanism of index lookup fails to find appropriate URIs for all slots to produce a completely instantiated SPARQL query, we use identified resources as starting point and retrieve all their properties. As an example, consider the following question and its corresponding template:

8. (a) How many employees does IBM have?
   (b)
```
SELECT COUNT(?y) WHERE {
   ?x -- ?p -- ?y
}
```
   Slots:
   - ⟨?p, property, employees⟩
   - ⟨?x, resource, IBM⟩

An index lookup retrieves `<http://dbpedia.org/resource/IBM>`, which now serves as starting point for finding possible instantiations for the property slot expressed by employees. To this end, we query the dataset for labels of all properties that connect the resource `<http://dbpedia.org/resource/IBM>` to other resources or to literals. For the above example, this yields a list of about 100 properties, including for example `products`, `industry`, `foundation place`, `company type`, `number of employees` and `num employees`.

[3] In the case of ASK queries, however, we cannot dismiss queries on the basis of an empty result set as they always return a boolean as answer. Since one concept found in the index is as good as any other concept found in the index, the decision to return a specific query as final result is postponed until the subsequent steps, when query ranks start to vary. Then the highest ranked query above a certain threshold (set to 0.9 in our case) is returned.

Next, all retrieved property labels are compared to the slot term, in our example employees, by means of the normalized Levenshtein distance *NLD* between two words w_1 and w_2, calculated as follows:

$$NLD(w_1, w_2) = 1 - \frac{\text{number of letter changes between } w_1 \text{ and } w_2}{\mathbf{max}(length(w_1), length(w_2))}$$

All properties that have a label with a Levenshtein distance above a certain threshold, in our case established as 0.95, is added as a new hypothesis with the *NLD* value as its rank. In our case the best matching property is `num employees` with a Levenshtein score of 0.73. Although this is below the threshold, the property label bears strong similarity with the slot term employees, we therefore want to permit it as a hypothesis. To this end, we apply an additional heuristic that assigns rank 1 to a property if its label contains the slot term as a substring.[4] Therefore both the property `<http://dbpedia.org/ontology/numberOfEmployees>` as well as the property `<http://dbpedia.org/property/numEmployees>` is added as hypotheses with rank 1.

Finally, this processing step yields the following query for the question How many employees does IBM have, which retrieves the correct answer:

```
9. SELECT ?y WHERE {
      <http://dbpedia.org/resource/IBM>
      <http://dbpedia.org/ontology/numberOfEmployees> ?y .
   }
```

2.4 Lookup in Lexical Database

Now consider the question Who is the mayor of Berlin. Both layers described above—index lookup and string similarity—do not find an answer to this question as the right interpretation involves the property `<http://dbpedia.org/ontology/leader>` rather than a property with label `mayor`. Therefore, in a third processing step, we use a lexico-semantic resource, in this case WordNet, to retrieve synonyms for slot terms. The slot term mayor, for example, leads to a list containing civil authority, politician, ex-mayor and city manager, among others. While tuning BELA on the QALD-2 training question set for DBpedia, we found that the overall results improve if this list is further expanded with the synonyms, hypernyms and hyponyms of all list elements; in our example this adds authority, leader, governor and judge, among others. These synonyms are matched to all properties retrieved for the resources explicitly mentioned in the question (`<http://dbpedia.org/resource/Berlin>`in the example), and in case of a match, an appropriate hypothesis is generated. In the case of the above question, this leads to the following correct SPARQL query:

```
10. SELECT ?y WHERE {
       <http://dbpedia.org/resource/Berlin>
       <http://dbpedia.org/ontology/leader> ?y .
    }
```

[4] The rank is set to 1 in order to push these hypotheses above the Levenshtein threshold of 0.95 and to make them fare better than purely string similar hypotheses.

2.5 Semantic Similarity

In case the string similarity and lexical expension steps do not find sufficiently high ranked hypotheses, BELA tries to find suitable hypotheses by means of *Explicit Semantic Analysis* (ESA).[5]

ESA is a method introduced by Gabrilovich and Markovitch [5] in order to represent and compare texts of any length in a high-dimensional vector space. The vector space is constructed based on a given document collection D, where the documents are assumed to describe natural concepts such as *cat* or *dog* (a so-called concept hypothesis). In the construction phase, a term-document matrix is built with a *tf.idf* weighting scheme [12] of terms w.r.t. the documents $d \in D$. A semantic interpreter then allows to map any given natural language text t onto concept vectors: Each word $w \in t$ is represented by the concept vector $\boldsymbol{c}(w)$ of the corresponding row in the term-document matrix, where each vector element denotes the strength of association with a particular document $d \in D$. For $|t| > 1$ (i.e., texts rather than single words) the vector $\boldsymbol{c}(t)$ constitutes the sum of the individual word vectors $\boldsymbol{c}(w)$ for all $w \in t$. Finally the two concepts vectors are compared using cosine similarity, thus yielding a semantic similarity score for the compared texts. While in the original work of Gabrilovich and Markovitch (2007) Wikipedia was used as background knowledge source, recent work has shown that also Wiktionary[6] and WordNet [4] can be used as background document collections. Initial experiments showed that using Wikipedia as background document collection produces the best results on tour task. In the following, all experiments involving ESA are thus carried out using Wikipedia as background knowledge base.[7]

Applying ESA to the question answering task allows us to relate, e.g., the expression painted and the ontology label `artist`, which fail to be connected by both string similarity and WordNet expanison.

3 Experiments

3.1 Evaluation Set-Up

BELA has been evaluated on the DBpedia training and test question sets provided by the 2nd Open Challenge on Question Answering over Linked Data[8] (QALD-2). A more detailed description of these datasets and the procedure for constructing it can be found in [8]. Both datasets comprise 100 natural language questions annotated with SPARQL queries and answers. From these questions we removed all out-of-scope questions (questions that cannot be answered within

[5] The implementation we used is available at: `http://code.google.com/p/dkpro-similarity-asl/`.
[6] `http://www.wiktionary.org`
[7] Results for all tested dictionaries can be found at `http://www.sc.cit-ec.uni-bielefeld.de/bela`.
[8] `http://www.sc.cit-ec.uni-bielefeld.de/qald-2`

the dataset) as well as questions relying on namespaces not yet part of our index, namely YAGO and FOAF. This filtering led to remaining 75 training and 72 test questions. In order to ensure a fair evaluation, we used only the training set for developing and fine-tuning the system, e.g. for determining the Levenshtein and ESA thresholds, and used the test set for the purpose of evaluation only. By manual tuning on the training dataset, the threshold for the normalized Levenshtein distance was set to 0.95, while the threshold for ESA was set to 0.4.

For evaluation we used the tool provided by the QALD-2 challenge. For each question q, precision, recall and F-measure are computed as follows:

$$Recall(q) = \frac{\text{number of correct system answers for } q}{\text{number of gold standard answers for } q}$$

$$Precision(q) = \frac{\text{number of correct system answers for } q}{\text{number of system answers for } q}$$

$$F\text{-}Measure(q) = \frac{2 * Precision(q) \times Recall(q)}{Precision(q) + Recall(q)}$$

On the basis of these, overall precision and recall values P and R, as well as an overall F-measure value F are computed as the average mean of the precision, recall and F-measure values for all questions. Additionally, we compute coverage as the percentage of questions for which an answer was provided: $Coverage = \frac{\text{number of queries with answer}}{|Q|}$. In order to also take into account the balance between F-measure and coverage, we introduce an F-measure F' as the harmonic mean of the coverage and the overall F-Measure $F' = \frac{2 \times Coverage \times F}{Coverage + F}$.

3.2 Results

Table 1 shows the results on the DBpedia training and test sets. It lists the number of answered queries, the coverage, the number of questions that were

Table 1. Results over the 75 DBpedia train and 72 DBpedia test questions

Module	Answered	Coverage	Correct	R	P	F	F'
DBpedia Train							
Index lookup	15	0.2	7	0.67	0.61	0.64	0.30
+ String similarity	29	0.38	16	0.77	0.73	0.75	0.50
+ Lexical expansion	37	0.49	20	0.74	0.69	0.71	0.57
+ Semantic similarity	39	0.52	22	0.75	0.71	0.73	0.60
DBpedia Test							
Index lookup	11	0.15	9	0.909	0.84	0.87	0.25
+ String similarity	20	0.27	13	0.85	0.74	0.79	0.40
+ Lexical expansion	29	0.4	16	0.71	0.63	0.67	0.50
+ Semantic similarity	31	0.43	17	0.73	0.62	0.67	0.52

Table 2. How many questions require the pipeline up to which module to be answered?

Module	Train	%	Test	%
String similarity	24	61	20	64
Lexical expansion	10	25	11	35
Semantic similarity	2	5	2	6

answered perfectly as well as the average precision, recall and F-measures F and F'.[9] The behavior of the system is as expected in the sense that the overall performance (F') increases with each processing step in the pipeline, where string similarity computation clearly has the most impact on the results, increasing performance by 20% on train and 15% on test. The use of a lexical database (in our case WordNet) increases the results by 7% on train and 10% on test, followed by the semantic similarity component, which increases results by 3% on train and 2% on test. Thus all components provide an added value to the overall pipeline. Table 2 lists the number of questions that can be answered at a certain processing step in the pipeline but could not be answered earlier.

3.3 Comparison with State-of-the-Art Systems

Table 3 compares the results of our system BELA (traversing the full pipeline) with the results of the systems that participated in the QALD-2 challenge and with Wolfram Alpha.[10] In addition to the number of correctly answered questions, we list the number of questions for which a partially correct answer was provided, i.e. questions with an F-measure strictly between 0 and 1.

Table 3. Results compared with results from the participants of the QALD-2 challenge (with coverage calculated over all 100 questions)

System	Answered	Coverage	Correct	Partially	R	P	F	F'
SemSeK	80	0.8	32	7	0.44	0.48	0.46	**0.58**
MHE	**97**	**0.97**	30	12	0.36	0.4	0.38	0.54
BELA	31	0.31	17	5	**0.73**	**0.62**	**0.67**	0.42
WolframAlpha	51	0.51	15	2	0.32	0.3	0.309	0.38
QAKiS	35	0.35	11	4	0.39	0.37	0.38	0.36
Alexandria	25	0.25	5	10	0.43	0.46	0.45	0.32

The comparison shows that BELA ranges, from the point of view of overall performance, in the middle field, outperforming Wolfram Alpha in particular. The main difference between our system and the two systems that outperform it—SemSeK and MHE—is that the latter achieve a much higher coverage at the price of a much lower precision.

[9] A more detailed listing of the results for each question can be found at http://www.sc.cit-ec.uni-bielefeld.de/bela

[10] In order to allow for a comparison with Wolfram Alpha, we submitted the test questions to the Wolfram Alpha portal and extracted and verified the results manually.

3.4 Performance

The following table shows the average time for answering a question (in seconds, calculated over the train and test dataset):[11]

Index lookup	+ String similarity	+ Lexical expansion	+ Semantic similarity
4.5	5.2	5.4	16.5

The average time for answering a question, to no or little surprise, increases when increasing the number of modules used by the system. However, the average cost of the index lookup, string similarity and lexical expansion steps is very similar; a significant increase in fact arises only when adding semantic similarity to the computation, raising the average time per second by around 11 seconds, while providing only a 2% performance increase.

The parsing and template generation step takes an average of 1.7 seconds per question. In future work, we will optimize the index lookup and pre-caching mechanism, now taking up an average of two seconds per questions. Saving the pre-cached informations after an experiment, the average time in the next experiment drops to around 4.6 second per question for the second and third step of the pipeline.

3.5 Manual, Query-Driven Extension of Lexical Coverage

Although similarity and relatedness measures can bridge the gap between natural language terms and ontology labels to a certain extent, they fail when the gap is too big. For example, all modules included in BELA failed to relate `created` and `author`, or `die` and `deathCause`. Now, mappings that are notoriously difficult to find for a machine could be easy to create by someone with basic domain knowledge. Considering, for example, a question answering system that logs the questions it fails to answer, a maintainer could manually specify index mappings for natural language expressions that are often used.

In order to show how little manual effort is required to increase precision and recall, we additionally report on a run of the full pipeline of the system enriched with an additional, manually created index that contains 14 mappings from natural language terms to URIs which BELA failed to identify, for example `high` → `<http://dbpedia.org/ontology/elevation>`.[12] Given such a manual index with 14 entries, the results increase, as shown in Table 4. Note that even the results on the test question set slightly increase, although when building the manual index only training questions were taken into account. Thus a relatively small manual effort can help bridging the gap between natural language expressions and ontology labels in case similarity and relatedness measures fail.

4 Discussion and Related Work

We can identify two major challenges when constructing SPARQL queries for natural language questions:

[11] Performed on a machine with a Intel® Core™ i3-2310M CPU @ 2.10GHz.

[12] The complete list can be found at `http://www.sc.cit-ec.uni-bielefeld.de/bela`

Table 4. Results of full pipeline with manually extended index

	Answered	Coverage	Correct	Partially	R	P	F	F'
Train (without)	39	0.52	22	11	0.75	0.71	0.73	0.60
Train (with)	40	0.53	26	10	0.80	0.81	0.80	0.63
Test (without)	31	0.43	17	6	0.73	0.62	0.67	0.52
Test (with)	32	0.44	18	6	0.74	0.639	0.688	0.53

– Briding the *lexical gap*, i.e. the gap between natural language expressions and ontology labels (e.g. mayor and `leader`, written and `author`)
– Bridging the *structural gap*, i.e. the gap between the semantic structure of the natural language question and the structure of the data

The lexical gap is quite well covered by the tools exploited in our pipeline, i.e. string similarity as well as lexico-semantic resources and semantic relatedness measures, all of which are quite standard in current Semantic Web question answering systems. An additional, recently emerging tool for bridging the lexical gap are repositories of natural language representations of Semantic Web predicates, acquired from a structured data repository together with a text corpus. Examples are the BOA pattern library [6] (used, e.g., in TBSL [13]) and the WikiFramework repository [10] (used, e.g. in QAKiS [3]). Both go beyond semantic similarity measures in also involving co-occurence patterns.

The structural gap, on the other hand, is less often addressed. Most systems map natural language questions to triple-based representations and simply fail if this representation does not match the actual data. A simple example is the query Give me all cities in Germany. Our system starts looking for resources of class city that are directly related to the entity Germany; in the actual data, however, some cities are only indirectly connected to their country, e.g. through their federal state. Such a case requires a search for indirect relationships in case direct ones cannot be found. *PowerAqua* [7], an open-domain question answering system for the Semantic Web, does exactly this. After mapping natural language questions to a triple-based representation and discovering relevant ontologies, PowerAqua first tries to find entity mappings, exploiting different word sense disambiguation techniques. Then it searches for direct relationships between the candidate entities, using WordNet expansion and also different filtering heuristics to limit the search space. If no direct relationships are found, indirect relationships are explored.

A slightly more difficult example is the question When did Germany join the EU. Our template generation process assumes a representation with two entities, Germany and the EU, and a relation join connecting them; PowerAqua[13] assumes a triple representation of form ⟨`date,join,Germany`⟩,⟨`Germany,?,EU`⟩. The actual DBpedia data, however, relates Germany to a date literal via the property

[13] Accessed through the online demo at `http://poweraqua.open.ac.uk:8080/poweraqualinked/jsp/`

`accessioneudate`, thus both representations fail to match it. Such cases therefore require more sophisticated techniques for inferring or learning the target triple structure from the data or an underlying ontology. In order to bridge the structural gap, a system architecture like ours would therefore require further iterations: Once the whole pipeline is traversed without having constructed a successful query, the template structure needs to be adapted or extended, triggering a new pipeline cycle.

We conjecture that a proper approach to bridging the structural gap is necessary to further increase the coverage and performance of question answering systems significantly, and that without such an approach, comprehensive question answering over linked data will fail, just like without a proper approach to bridging the lexical gap.

Another major challenge for question answering over linked data is the processing of questions with respect to not only one but several datasets (ultimately the whole linked data cloud), which includes the search for relevant ontologies as well as the integration of query parts constructed from different sources. This challenge has so far only been taken up by PowerAqua. Also, evaluating and comparing question answering systems in such an open-domain scenario is inherently difficult.

5 Conclusion

We have presented a layered architecture for question answering over linked data that relies on an ordered processing pipeline consisting of the following steps: an inverted index lookup, the computation of string similarities, a lookup in a lexical database such as WordNet and a semantic similarity computation step based on Explicit Semantic Analysis. We have systematically evaluated the contribution of each of these component on the benchmarking dataset provided by the 2nd Open Challenge on Question Answering over Linked Data (QALD-2), showing that each of these processing components has an important impact on the task, increasing coverage and F-measure while obviously increasing the overall processing time. We have also shown that our approach can compete with other state-of-the-art systems, e.g. clearly outperforming Wolfram Alpha. Finally, we have shown how an iterative improvement lifecycle that adds additional mappings to the system can substantially improve the performance of the system. Future work will consider adding additional lexical knowledge to the system (e.g. Wiktionary and lexical pattern libraries), and will especially focus on adding iterations that adapt the structure of the query templates, in order to bridge the gap between the semantic structure of the natural language question and the structure of the dataset.

References

1. Anderka, M., Stein, B.: The ESA Retrieval Model Revisited. In: Proc. of the 32th Annual International ACM SIGIR Conference, pp. 670–671 (2009)

2. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data – The Story So Far. Int. Journal on Semantic Web and Information Systems 5 (2009)
3. Cabrio, E., Palmero Aprosio, A., Cojan, J., Magnini, B., Gandon, F., Lavelli, A.: QAKiS @ QALD-2. In: Proc. of ILD 2012 (2012), http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/2/proceedings_ILD2012.pdf
4. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press (1998)
5. Gabrilovich, E., Markovitch, S.: Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In: Proc. of the 20th International Joint Conference on Artificial Intelligence, pp. 1606–1611 (2007)
6. Gerber, D., Ngonga Ngomo, A.-C.: Bootstrapping the Linked Data Web. In: Proc. of WekEx at ISWC 2011 (2011)
7. Lopez, V., Fernández, M., Motta, E., Stieler, N.: PowerAqua: supporting users in querying and exploring the Semantic Web content. Semantic Web Journal (to appear), http://www.semantic-web-journal.net/
8. Lopez, V., Unger, C., Cimiano, P., Motta, E.: Evaluating Question Answering over Linked Data. Journal of Web Semantics (under review)
9. Lopez, V., Uren, V., Sabou, M., Motta, E.: Is Question Answering fit for the Semantic Web? A Survey. Semantic Web Journal 2, 125–155 (2011)
10. Mahendra, R., Wanzare, L., Bernardi, R., Lavelli, A., Magnini, B.: Acquiring Relational Patterns from Wikipedia: A Case Study. In: Proc. of the 5th Language and Technology Conference (2011)
11. Miller, G.A.: WordNet: A Lexical Database for English. Communications of the ACM 38, 39–41 (1995)
12. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill (1983)
13. Unger, C., Bühmann, L., Lehmann, J., Ngonga Ngomo, A.-C., Gerber, D., Cimiano, P.: Template-Based Question Answering over RDF data. In: Proc. of WWW 2012 (2012)
14. Zesch, T., Müller, C., Gurevych, I.: Using Wiktionary for Computing Semantic Relatedness. In: Proc. of the 23rd AAAI Conference on Artificial Intelligence, pp. 861–867 (2008)

Cross Lingual Semantic Search by Improving Semantic Similarity and Relatedness Measures

Nitish Aggarwal*

Unit for Natural Language Processing, Digital Enterprise Research Institute,
National University of Ireland, Galway
firstname.lastname@deri.org

Abstract. Since 2001, the semantic web community has been working hard towards creating standards which will increase the accessibility of available information on the web. Yahoo research recently reported that 30% of all HTML pages contain structured data such as microdata, RDFa, or microformat. Although multilinguality of the web is a hurdle in information access, the rapid growth of the semantic web enables us to retrieve fine grained information across the language barrier. In this thesis, firstly, we focus on developing a methodology to perform cross-lingual semantic search over structured data (knowledge base), by transforming natural language queries into SPARQL. Secondly, we focus on improving the semantic similarity and relatedness measures, to overcome the semantic gap between the vocabulary in the knowledge base and the terms appearing in the query. The preliminary results are evaluated against the QALD-2 test dataset, which achieved a F1 score of 0.46, an average precision of 0.44, and an average recall of 0.48.

1 Introduction

The rapid growth of the semantic web offers a wealth of semantic knowledge for facilitating an interactive way to access the information, by providing structured metadata[1] in a standard format such as microdata, RDFa or microformat. This structured data facilitates the possibility of automatic reasoning and inferencing. Thus, by embedding such knowledge within web documents, additional key information about the semantic relations among data objects can be captured.

People desire to access the multilingual information available on the web, while querying in their native language. To address this issue, we present cross-lingual semantic search, which aims to retrieve all the relevant information even if it is available in languages different from the query language. Translating search queries ([17], [10]) into the corresponding languages of the documents is the current approach for cross-lingual information retrieval. However, the poor accuracy of translation of short texts like queries, poses a certain problem to

* Supervisor: Dr. Paul Buitelaar.

[1] http://events.linkeddata.org/ldow2012/slides/Bizer-LDOW2012-Panel-Background-Statistics.pdf

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 375–382, 2012.

this method. Hence, using large knowledge bases as an interlingua [23] may prove beneficial.

The approach discussed here considers DBpedia [3] as the structured knowledge base. DBpedia contains a large ontology describing more than 3.5 millions instances extracted from Wikipedia info-boxes, forming a good and general structured knowledge source. Also, it is very well-connected to several other linked data repositories in the Semantic Web. DBpedia contains a huge number of instances in many languages, however, the ontology (properties & classes) is mainly covered in English. Thus, querying this knowledge base is not possible in other languages even if the instances are multilingual. Cross-lingual search is required to query this structured knowledge base, which is the major goal of this work.

In order to query a structured knowledge base, one requires a structured query to start with. Therefore, the conversion of a natural language query (NL-query) to a structured query is required. There are several efforts ([6], [15], [14]) to convert a NL-query to SPARQL[2] in the monolingual scenario. In particular, Freitas et al. [6] proposed an approach based on the combination of entity search, a Wikipedia-based semantic relatedness (using the Explicit Semantic Analysis measure), and spreading activation. Our approach takes inspiration from Freitas et al. to perform search across different languages. We focus on better interpreting NL-queries in different languages, driven by traversal over the large structured knowledge base, and constructing a corresponding SPARQL query. However, the gap between the vocabularies used in NL-queries and the structured knowledge base makes this task challenging. This gap can be filled by calculating cross-lingual similarity and relatedness between these vocabularies, which is the key to our proposed approach. In particular, we present our approach for cross-lingual semantic search, which includes three components: entity search, linguistic analysis, and semantic similarity and relatedness.

Following this approach, cross-lingual document retrieval can also be performed if the documents are already marked-up with the knowledge base, for instance, Wikipedia articles are annotated with DBpedia.

2 Proposed Approach

The key to our approach for cross-lingual semantic search is the interpretation of NL-queries in different languages, driven by the traversal over the large structured knowledge base, and construction of the corresponding SPARQL query. Semantic and linguistic variations of natural language text can create a gap between terms appearing in NL-queries and the vocabulary of the knowledge base. A well-interpreted SPARQL query, which is formed from a given NL-query can overcome this gap, by referring to the knowledge base. Figure 1 shows the three components of our approach along with an example of a NL-query in German[3].

[2] http://www.w3.org/TR/rdf-sparql-query/

[3] Translated from the QALD-2 challenge dataset, which has 100 NL-queries in English, over DBpedia.

2.1 Entity Search

The query interpretation process starts by identifying the potential entities, i.e. the ontology concepts (classes and instances), present in the NL-query. A baseline entity search can be defined as the identification of an exact match between the label of an ontology concept and the query text segment by using a simple string similarity, for example, DBpedia: Bill_Clinton shown in Figure 1. However, more sophisticated identification is needed to handle a rich semantic and linguistic analysis of NL-queries, for example, the English NL-query "Give me the capitals of all countries in Africa" has multiple possible entities for the same text segment, "DBpedia: Africa", "DBpedia: Country" and "YAGO: African_Country". "DBpedia: Africa" and "DBpedia: Country" can be identified by the baseline, but these are not the most appropriate entities to link for this NL-query. Therefore, semantic and linguistic analysis are required to identify "YAGO: African_Country" for the text segment "countries in Africa". Semantic analysis provides that "Africa" and "African" are the same and linguistic analysis interprets that "countries in Africa" is equivalent to "African country" as will be explained in Section 2.2.

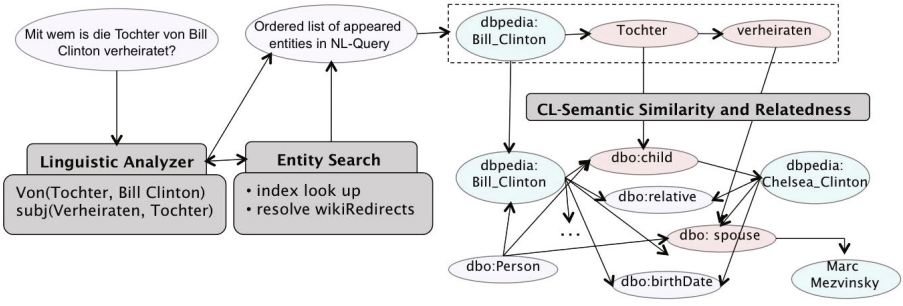


Fig. 1. Query interpretation pipeline for an example German NL-Query "Mit wem is die Tochter von Bill Clinton verheiratet?" which is "Who is the daughter of Bill Clinton married to?" in English

For languages other than English, entity search becomes more challenging as they may include richer linguistic variations such as compound words and gender specific articles.

2.2 Linguistic Analysis

A deep linguistic analysis of the NL-query is performed by generating a parse tree and typed dependencies, by using the Stanford parser.[4] The generated parse tree provides key phrase extraction for identifying potential ontology concepts. For instance, in the query "Who wrote the book The pillars of the Earth?", the

[4] `http://nlp.stanford.edu/software/lex-parser.shtml`

phrase “The pillars of the Earth” is identified as a noun phrase. This suggests that we should use the whole phrase to find an ontology concept, rather than separated search for each of the tokens. Linguistic analysis also provides entity recognition with linguistic variations. For instance, in the above discussed example, linguistic analysis interprets “Countries in Africa” as PP_in(countries, Africa), which means it is equivalent to YAGO: African_Country.

We convert the given NL-query into an ordered list of potential terms by using generated typed dependencies. To create this ordered list, first we select a central term among all the identified terms, where the central term is the most plausible term to start matching a NL-query to the vocabulary appearing in the knowledge base. This selection is performed by prioritising the ontology instances over classes. Then, we retrieve the directly dependent terms of the central term by following the generated typed dependencies, and add them into the ordered list. Similarly, we perform this action for all the other terms in the list. For instance, in our example NL-query shown in Figure 1, firstly, the system identifies “Bill Clinton” as a central term,[5] and then “Tochter” as direct dependent of “Bill Clinton” followed by “verheiratet” as direct dependent of “Tochter”.

2.3 Knowledge Base Graph Traversing Using Semantic Similarity and Relatedness

A knowledge base graph can be defined as the structured data of well-connected entities and their properties. Therefore, the next step is the traversal of the obtained ordered list of potential terms from the linguistic analysis step, over this knowledge base. For instance in Figure 1, the ordered list obtained from our example query “Mit wem is die Tochter von Bil Clinton verheiratet?” is <Bill Clinton, Tochter, verheiratet>. Firstly, we search for the Entity “Bill Clinton” in DBpedia as our approach takes DBpedia as knowledge base, and retrieve all of its properties. Then, we find the most semantically similar or related property of direct dependent term “Tochter” by calculating cross-lingual similarity between all the properties of Bill Clinton and the term “Tochter”. After obtaining relevant property, i.e. child, we find the entity DBpedia:Chelsea_Clinton, connected with entity Bill Clinton by property child. We perform the same steps with the retrieved entity for directly dependent term “verheiratet” of “Tochter”, and so on till end of the ordered list. Finally, we retrieved the relevant entity and also all the linked documents in different languages containing the description about this entity.

Our approach relies on semantic matching between recognised potential terms and properties in the knowledge base. Therefore, to find the most appropriate properties, a good cross-lingual semantic similarity and relatedness measure is required. We cannot rely solely on semantic similarity measures, as relatedness can better map the term “verheiratet” to the retrieved property “spouse”, because they are semantically related but not semantically similar. Therefore, to investigate different similarity and relatedness measures, we are building a Java

[5] The term to start the search around in whole DBpedia graph.

library which will include many structure-based and corpus-based similarity and relatedness measures. We are performing the experiments with several structure-based measures ([20], [25], [24], [12], [19]) and corpus-based measures ([11], [9], [7], [22]). However, corpus-based approaches rely on the assumption that related words would co-exist in the same document, which is normally not the case with the similar words, e.g. synonymy. Hence, towards the initial step for tuning the corpus-based relatedness to similarity [1], we combine the Explicit Semantic Analysis (ESA) [7] based relatedness score with the WordNet-based Lin [12] similarity scores calculated for the words falling under the corresponding syntactic role category, in both of the short phrases to be compared. We are further working on ESA and its variants (association strength, relevancy function and vector correlations) [22] to improve the corpus-based relatedness, and are planning to submit it in WWW-2013.

3 Evaluation

For the preliminary evaluation of our proposed approach, we examine it in the monolingual scenario. In this experiment, we used the WordNet-based similarity and relatedness proposed by Pirro [19], as it is computationally efficient in comparison to ESA. We performed the experiments [2] against the QALD-2 test dataset, which includes 100 NL-queries in English and their corresponding SPARQL, to retrieve the relevant entities from DBpedia. We calculated the average precision, average recall, and F1 score of the results obtained by our approach. Our approach does not completely explore all of the types of queries appearing in the dataset, as some of them are more challenging complex NL-queries, which would require SPARQL aggregation, and ask type queries. The results are shown in Table 1.

For testing our approach in a cross-lingual setting, we are preparing the benchmark by manually translating the English NL-queries of the QALD-2 test dataset into German.

Table 1. Evaluation on QALD-2 test dataset of 100 NL-queries over DBpedia

Total	Answered	Right	Partially right	Avg. Precision	Avg. Recall	F1
100	80	32	7	0.44	0.48	0.46

4 State of the Art

Most of the proposed approaches to address the task of Cross-Lingual Information Retrieval (CLIR), reduce the problem into the monolingual scenario, by translating the search query or documents in the corresponding language. Many of them perform query translation ([16], [18], [17], [10])) into the language of the documents. However, all of these approaches suffer from the poor performance of the machine translation on short texts (query). Jones et al. [10] performed

query translation by restricting the translation for the cultural heritage domain, while [17] makes use of the Wikipedia cross-lingual links structure.

Without relying on machine translation, some of the approaches ([13], [26], [21]) make use of distributional semantics. They calculate the cross lingual semantic relatedness measures between query and the documents. However, none of these approaches take any linguistic information into account, and do not make use of large available structured knowledge base. With an assumption that documents of different languages are already marked-up with the knowledge base (for instance, Wikipedia articles are annotated with the DBpedia), the problem of CLIR can be converted into the query over structured data. There is still a language barrier, as queries can be in different languages, while most of the structured data are only available in English. Qall-Me [5] performs NL-query over the structured information, by using the textual entailment to convert a natural language question into SPARQL. This system relies on availability of multilingual structured data. It can only retrieve the information which is available in the query language. Therefore, this system is not able to perform CLIR. Freitas et al. [6] proposed an approach for natural language querying over linked data, based on the combination of entity search, a Wikipedia-based semantic relatedness (using ESA) measure, and spreading activation. Our approach takes inspiration from the same.

Since our proposed approach mainly relies on good cross-lingual similarity and relatedness measures, we are working on improving the existing measures to reflect better similarity and relatedness. There are several structure-based methods ([20], [25], [24], [12], [19]), and corpus-based methods ([13], [26], [22]), to calculate similarity and relatedness. Although, structure-based methods require a structure predefined by experts, which is not a trivial task for a large number of language pairs. Corpus-based methods represent the semantics of a term by its distribution in large multilingual corpus, and calculate relatedness by taking correlation between distribution of terms to be compared. These approaches only require comparable multilingual corpus like Wikipedia. However, the corpus-based methods perform well for document similarity, but need to improve for short text or phrases. Therefore, we are working on improving these measures to reflect better similarity and relatedness scores.

5 Conclusion and Future Work

We presented our proposed approach for cross-lingual semantic search, which includes entity search, deep linguistic analysis, and cross-lingual semantic similarity and relatedness. With this approach, cross-lingual information retrieval at document level can also be performed, if the documents are already marked up with the structured knowledge base.

The next main steps are to develop the different components of our proposed approach for cross-lingual semantic search. All of these components mainly rely on better cross-lingual similarity and relatedness measures. Therefore, we are mainly concerned in improving the existing semantic relatedness measures to reflect higher accuracy in semantic matching for multiple languages. As discussed

in Section 2.3, we are working on ESA and its variants to improve the similarity and relatedness measures. Hence, we are evaluating it with different association strengths such as Latent Semantic Analysis [11] and Latent Dirichlet Allocation [4]. Thomas et al. [8] report significant improvement by taking probabilistic weighted association strength into account. However, one other major issue in corpus-based relatedness is that all the measures do not take the mutual relatedness of documents into account. Hence, we are planning to investigate the current ESA model by fusing it with other existing measures.

Acknowledgements. This work is supported in part by the European Union under Grant No. 248458 for the Monnet project and by the Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2).

References

1. Aggarwal, N., Asooja, K., Buitelaar, P.: DERI&UPM: Pushing corpus based relatedness to similarity: Shared task system description. In: SemEval-2012, SEM, First Joint Conference on Lexical and Computational Semantics, and co-located with NAACL, Montreal, Canada (June 2012)
2. Aggarwal, N., Buitelaar, P.: A system description of natural language query over dbpedia. In: 9th Extended Semantic Web Conference Interacting with Linked Data, ILD 2012 (May 2012)
3. Bizer, C., Cyganiak, R., Auer, S., Kobilarov, G.: Dbpedia.org - querying wikipedia like a database. In: Developers track at 16th International World Wide Web Conference (WWW 2007), Banff, Canada (May 2007)
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022 (2003), http://dl.acm.org/citation.cfm?id=944919.944937
5. Ferrández, Ó., Spurk, C., Kouylekov, M., Dornescu, I., Ferrández, S., Negri, M., Izquierdo, R., Tomás, D., Orasan, C., Neumann, G., Magnini, B., Vicedo, J.L.: The qall-me framework: A specifiable-domain multilingual question answering architecture. Web Semantics 9, 137–145 (2011)
6. Freitas, A., Oliveira, J.G., O'Riain, S., Curry, E., Pereira da Silva, J.C.: Querying Linked Data Using Semantic Relatedness: A Vocabulary Independent Approach. In: Muñoz, R., Montoyo, A., Métais, E. (eds.) NLDB 2011. LNCS, vol. 6716, pp. 40–51. Springer, Heidelberg (2011)
7. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 1606–1611 (2007)
8. Gottron, T., Anderka, M., Stein, B.: Insights into explicit semantic analysis. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM 2011 (2011)
9. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1999, pp. 50–57 (1999)
10. Jones, G., Fantino, F., Newman, E., Zhang, Y.: Domain-specific query translation for multilingual information access using machine translation augmented with dictionaries mined from Wikipedia. In: CLIA 2008, p. 34 (2008)
11. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse Processes 25, 259–284 (1998)

12. Lin, D.: An information-theoretic definition of similarity. In: Proc. of the 15th Int'l. Conf. on Machine Learning, pp. 296–304 (1998), http://portal.acm.org/citation.cfm?id=657297
13. Littman, M., Dumais, S.T., Landauer, T.K.: Automatic cross-language information retrieval using latent semantic indexing. In: Cross-Language Information Retrieval, ch. 5, pp. 51–62. Kluwer Academic Publishers (1998)
14. Lopez, V., Motta, E., Uren, V.S.: PowerAqua: Fishing the Semantic Web. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 393–410. Springer, Heidelberg (2006)
15. Lopez, V., Sabou, M., Motta, E.: PowerMap: Mapping the Real Semantic Web on the Fly. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 414–427. Springer, Heidelberg (2006)
16. Lu, C., Xu, Y., Geva, S.: Web-based query translation for english-chinese CLIR. Computational Linguistics and Chinese Language Processing (CLCLP) 13(1), 61–90 (2008)
17. Nguyen, D., Overwijk, A., Hauff, C., Trieschnigg, D.R.B., Hiemstra, D., De Jong, F.: Wikitranslate: query translation for cross-lingual information retrieval using only wikipedia. In: Proceedings of the 9th CLEF (2009)
18. Pirkola, A., Hedlund, T., Keskustalo, H., Jrvelin, K.: Dictionary-based cross-language information retrieval: Problems, methods, and research findings. Information Retrieval 4, 209–230 (2001)
19. Pirró, G., Euzenat, J.: A Feature and Information Theoretic Framework for Semantic Similarity and Relatedness. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 615–630. Springer, Heidelberg (2010)
20. Rada, R., Mili, H., Bicknell, E., Blettner, M.: Development and application of a metric on semantic nets. IEEE Transactions on Systems, Man and Cybernetics, 17–30 (1989)
21. Sorg, P., Braun, M., Nicolay, D., Cimiano, P.: Cross-lingual information retrieval based on multiple indexes. In: Working Notes for the CLEF 2009 Workshop, Cross-lingual Evaluation Forum, Corfu, Greece (September 2009)
22. Sorg, P., Cimiano, P.: An Experimental Comparison of Explicit Semantic Analysis Implementations for Cross-Language Retrieval. In: Horacek, H., Métais, E., Muñoz, R., Wolska, M. (eds.) NLDB 2009. LNCS, vol. 5723, pp. 36–48. Springer, Heidelberg (2010)
23. Steinberger, R., Pouliquen, B., Ignat, C.: Exploiting multilingual nomenclatures and language-independent text features as an interlingua for cross-lingual text analysis applications. In: Proc. of the 4th Slovenian Language Technology Conf. Information Society (2004)
24. Sun, P.R.: Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp. 448–453
25. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, ACL 1994, pp. 133–138. Association for Computational Linguistics, Stroudsburg (1994), http://dx.doi.org/10.3115/981732.981751
26. Zhang, D., Mei, Q., Zhai, C.: Cross-lingual latent topic extraction. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL 2010, pp. 1128–1137. Association for Computational Linguistics, Stroudsburg (2010), http://dl.acm.org/citation.cfm?id=1858681.1858796

Quality Reasoning in the Semantic Web

Chris Baillie, Peter Edwards, and Edoardo Pignotti

Computing Science & dot.rural Digital Economy Research,
University of Aberdeen, United Kingdom
{c.baillie,p.edwards,e.pignotti}@abdn.ac.uk

Abstract. Assessing the quality of data published on the Web has been identified as an essential step in selecting reliable information for use in tasks such as decision making. This paper discusses a quality assessment framework based on semantic web technologies and outlines a role for provenance in supporting and documenting such assessments.

Keywords: provenance, linked data, quality assessment.

1 Background

In recent years the World Wide Web has evolved from a collection of hyperlinked documents [3] to a vast ecosystem of interconnected documents, services and even people. Content on the Web suffers from a range of issues associated with data quality [5], as illustrated by this quote from one of the founders of the Internet:

> "The problem is - we don't know whether the information we find [on the Web] is accurate or not. We don't necessarily know what its provenance is. So we have to teach people how to assess what they've found. [...] there's so much juxtaposition of the good stuff and not-so-good stuff and flat-out-wrong stuff or deliberate misinformation or plain ignorance."
> *Vint Cerf, July 2010*

This highlights how the open nature of the Web enables anyone or any 'thing' to publish any content they choose. As a result, poor quality data can quickly propagate[1] and appropriate mechanisms to assess the quality of Web content are essential if agents (people or software) are to identify reliable information for use in tasks such as decision making and planning. Given the scope of the Web we have chosen to investigate these issues within the Web of Linked Sensor Data [11], a subset of the Web of Linked Data comprising semantic descriptions of sensors and their observations. Current examples of quality assessment frameworks such as Bizer and Cygniak's WIQA [2], and Lee et al's AIMQ [9] assess quality by examining data against a number of *quality dimensions* such as `accuracy`, `timeliness`, and `relevance` as defined by a number of *quality*

[1] http://www.w3.org/2005/Incubator/prov/wiki/Use_Case_Report#Information_Quality_Assessment_for_Linked_Data

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 383–390, 2012.

metrics. Assessments such as these often require additional metadata describing the context surrounding data (e.g. the characteristics of the sensor or the phenomenon measured by the observation), something that can be provided by publishing linked data [3]. We argue here that this context should also include provenance information, a record of the entities and processes involved in data derivation, as this has been identified as an essential step to support users to better understand, trust, reproduce, and validate the data available on the Web [10]. Provenance should therefore play a key role in evaluating quality as it provides information about data sources, the method used in data creation, and how the data has transformed over time - including who had access to the data, who processed it, and how the data was previously assessed.

Patel-Schneider and Fensel [12] describe Berners-Lee's vision of a semantic web language stack comprising different layers, each providing an intermediate language standard. This stack uses XML as a base standard for representing metadata, each layer above this base then adds new capabilities for expressing semantics. At the top of this stack is a layer dedicated to trust, famously illustrated by Berners-Lee's "*Oh, yeah?*"[2] button, which asks the Web "*how do I know I can trust this information?*". Richardson et al [13] describe trust as "*belief in a statement [...] A high value means that the statement is accurate, credible and/or relevant*", dimensions which are similar to those identified as important in evaluating the quality of data. This suggests that quality assessment should play an important role in the semantic web stack, either as a layer on its own or as a sub-component of the trust layer.

In our work to date we have investigated a number of application scenarios that employ sensors such as transport telematics, physiological monitoring in healthcare, and environmental conservation. In the first of these scenarios a crowdsourcing system is used to generate data describing the locations of public transport vehicles. The system relies on passengers activating a smartphone app that monitors their location using the phone's built-in GPS receiver. Other users can then use this system to discover when the next bus will arrive at their local bus stop. There are a number of possible sources for low quality data in this scenario, including poor mobile phone network coverage, degradation of the GPS signal, and malicious users. Being able to evaluate the quality of data is essential if this service is to be reliable and trustworthy.

To provide a focus for our research we have developed the following hypothesis: *publishing semantic descriptions of data and their provenance provides additional context that enhances quality assessment*. There are two key elements here: *context* refers to metadata describing the situation in which the observation was created and its provenance, such as the observed phenomenon (e.g. temperature), the feature of interest (e.g. a city), or the agent that controlled the sensing process; *enhancements* refer to how quality assessment is improved or new forms of assessment are enabled.

We have identified three potential enhancements: a) being able to evaluate a wider range of quality dimensions; b) being able to include a wider range of

[2] http://www.w3.org/DesignIssues/UI.html

data properties while evaluating individual quality dimensions; and c) being able to reduce the time taken to evaluate quality by re-using results from previous quality assessments. These are described in greater detail in section 4.

2 Related Work

Recent years have witnessed growing interest in semantic sensor networks. For example, the Open Geospatial Consortium ran a Sensor Web Enablement initiative [4] which aimed to develop a number of standard encodings for sensor measures. Le-Phouc and Hauswirth [8] built upon this, illustrating how linked sensor data can be published by following the linked data principles. This enables links to other datasets that provide additional contextual information about the original data. For example, observations from a GPS device can link to data describing the transport route that a vehicle should be using. There are a number of existing ontologies describing sensors and their observations [1,7]. The W3C Semantic Sensor Networks Incubator Group developed its own ontology[3] after a survey of these existing sensor ontologies and represents a state-of-the-art model describing sensor networks. However, while these ontologies are suitable for describing sensors and their observations, they provide only minimal observation provenance in the form of a description of the sensing method used to produce the observation. We argue that this is insufficient as there is more to provenance than just the process that created the observation, including details of the agent that controlled the process and the entities that were used by the process (e.g. the sensing device).

Quality assessment is the process of determining how suitable a piece of information is for a particular use and is performed by evaluating data against a number of *quality dimensions* such as `accuracy`, `timeliness`, and `relevance`. Bizer and Cygniak's WIQA framework [3] is a collection of software components that perform quality assessment using a number of *quality metrics* to examine data content, its context, and any associated external ratings. To our knowledge, the WIQA framework does not enable users to author their own policies to guide the information filtering process. We argue that this is key to any quality assessment framework because quality is highly subjective and task dependent.

Hartig and Zhao [6] present an approach to using provenance information about the data on the Web to assess its quality and trustworthiness. Their solution identifies provenance elements and the relationships between them. These elements represent specific provenance information such as the data producer or the process of data creation. Once the provenance graph has been generated, this data can be used in order to assess information quality by assigning *impact values* to the nodes, representing how processes and agents may have influenced data quality. Again, this solution does not enable users to define their own quality metrics.

There is no consensus on how quality metrics should be defined. Furber and Hepp [5] describe the use of SPARQL rules to guide quality assessment. Their

[3] `http://www.w3.org/2005/Incubator/ssn/`

model of quality assessment (DQM) has provision for a limited number of quality dimensions (currently `timeliness`, `accuracy`, `completeness`, and `uniqueness`). Having analysed a number of real application scenarios we have requirements for dimensions that are not defined in DQM such as `availability` (the time between the observation being created and published on the server) and `relevance` (the extent to which the observation describes the phenomenon in which we are interested).

3 Work to Date

To provide a realistic platform for our research we have developed a basic sensor network framework that can receive input from Arduino[4] based sensors and also smartphones. Observations are transmitted as a JSON[5] string to the observation web service, which uses the W3C Semantic Sensor Network Incubator Group ontology to create a semantic representation, in RDF, of the observation. The example in Figure 1 illustrates a sensor observation described using this ontology. Sensors are characterised using instances of `ssn:Sensor` and their observations using a combination of `ssn:Observation`, `ssn:ObservationValue`, and `ssn:SensorOutput`. The SSN ontology provides a number of properties that enable us to describe certain aspects of the context in which the observation was created. For example, `ssn:observationSamplingTime` allows us to describe when the observation value was originally measured and `ssn:observationResultTime` can describe when the observation was made available. We can also describe the `ssn:Property` (the phenomenon measured by the observation, e.g. speed or temperature) and the `ssn:FeatureOfInterest` (the entity to which the `ssn:Property` applies, e.g. a vehicle or location). In implementing the passenger information scenario, described earlier, we have extended the SSN ontology to enable us to capture more contextual information. As observations are transmitted to a server from a mobile phone we create the `_:serverTime` property to describe when observations are received by the server. The GPS observations we are working with detail latitude and longitude and so we capture these using the W3C Semantic Web Interest Group's Basic Geo Vocabulary[6] `geo:lat` and `geo:long`. We can also represent the error associated with the observation using `_:accuracy`, along with the vehicle's `_:speed` and `_:heading`. This extra metadata enables our framework to perform a more comprehensive assessment of quality, as described later.

We characterise the quality assessment process using Furber and Hepp's Data Quality Management (DQM) ontology (Figure 2). This ontology enables the definition of `dqm:DataRequirements` that specify how quality assessment should be performed (i.e. *quality metrics*). A number of basic quality rules are built into the model (e.g. legal and illegal values, and unique values). However, these are not capable of describing application-specific data requirements such as calculating

[4] http://www.arduino.cc
[5] http://www.json.org
[6] http://www.w3.org/2003/01/geo/

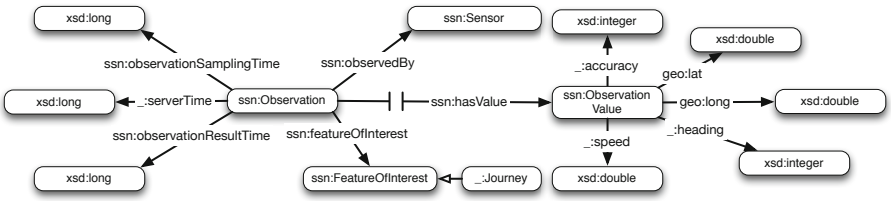


Fig. 1. An example sensor observation characterised using the SSN ontology

the distance between a GPS observation and a bus route. We have constructed a number of these requirements using the SPIN - SPARQL inferencing notation[7] which allow custom rules to be associated with `dqm:DataRequirement` instances (see example in Figure 2).

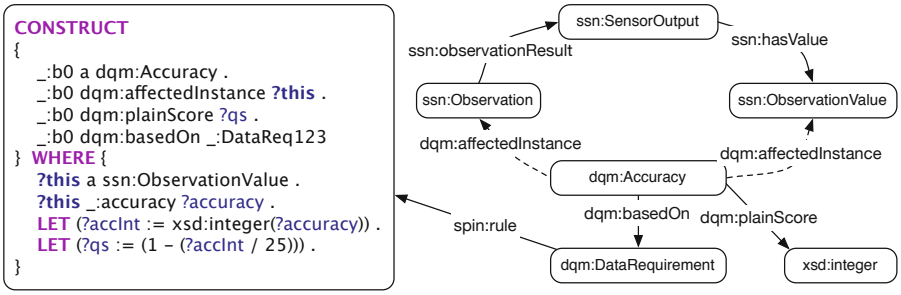


Fig. 2. Quality assessment characterised using the DQM ontology

We have also implemented a web based client that displays these sensor observations on a map based on the values of the `geo:lat` and `geo:long` properties. Clicking on these observations sends the observation's URI to the quality assessment service. This service employs a SPIN reasoner, guided by a number of rules, to evaluate the quality of selected observations which are returned to the web browser and displayed to the user. Figure 2 contains an example SPIN rule from the passenger information scenario that evaluates the accuracy of GPS observations, those with a low error are assigned a high quality score by the reasoner, which also annotates observations with the quality assessment results. Other examples include `timeliness` (observations older than 10 minutes are considered low quality), and `relevance` (observations farther than 250 metres from the expected route of travel are low quality). When assessment is complete instances of `dqm:QualityScore` are used to annotate the corresponding `ssn:Observation` or `ssn:ObservationValue` via the `dqm:affectedInstance` property.

[7] http://spinrdf.org/

4 Future Plans

At present our framework only examines the metadata describing the context surrounding sensor observations. We have already concluded that the SSN ontology is not sufficient to capture the provenance of sensor observations. We have identified the Prov-O[8] ontology as suitable for this task as it introduces a minimal set of concepts to represent provenance information in different application domains. Moreover, Prov-O conforms with the OWL 2 RL profile (scalable reasoning) which should facilitate the integration of provenance reasoning within our existing rule-based engine. This ontology is still being developed therefore we need to determine if Prov-O is capable of expressing the provenance of sensor observations. *Can the SSN and Prov-O ontologies be combined to represent the provenance of sensor observations?* For example, SSN can represent both the sensing process and the device that created an observation but with the inclusion of Prov-O we can also represent the agent that controlled the sensing process. *Can an SSN sensing process also be characterised as a Prov-O* `Activity`*?* We believe that this should be possible but need to investigate whether the semantics in both ontologies will permit this. The outcome of this investigation could be useful to the group developing Prov-O.

Another important question we need to address is: *Should the provenance of sensor observations be captured as they are created?* or *Should provenance be inferred only when a specific observation is requested?* Capturing the provenance of each observation could lead to the generation of large amounts of provenance data. Inferring provenance would avoid having to store much of this data but could increase the time taken to reason about its quality as the reasoner must perform two tasks (inferring provenance and performing quality assessment). We are also interested in answering the following question: *How can we use the provenance of existing quality scores to determine if these results can be reused?* This will involve either capturing or inferring the provenance of quality scores and authoring a number of new data requirements that can consider this provenance when performing new assessments. This raises the following issue: *Are DQM and Prov-O sufficient to characterise the provenance of quality scores?* For example, `dqm:DataRequirements` and `dqm:QualityScores` could both be characterised as a `prov:Entity` and so a combination of the two ontologies could potentially provide a complete account of quality score provenance.

We also have a number of questions relating to how reasoning is performed within our quality assessment framework. *What kind of rules (based on the Prov-O / SSN / DQM combination) can be used to support quality assessment?* We have already identified a number of ways in which the provenance of sensor observations can be used to support quality assessment. For example, we can examine the reputation of the agent associated with the sensing process, the type of device that created the observation, and how the observation has been transformed since it was created (e.g. converting location observations between certain co-ordinate systems can reduce the accuracy of observations). We have

[8] `http://www.w3.org/2011/prov/`

also identified a number of scenarios in which agents could re-use quality scores, e.g. Agent A could re-use Agent B's quality result because they are in the same social network and trust each other, or because Agent B's data requirement matches one of Agent A's. We will continue to identify more scenarios that will, in turn, inform new data requirements.

Our hypothesis, in section 1, states that publishing semantic representations of data and their provenance provide additional context that enhances quality assessment. We will measure the extent to which the provision of additional contextual information is useful to quality assessment by documenting the number of quality dimensions that can be evaluated with and without this metadata. For example, a description of an observed value associated with a timestamp can only be evaluated for `timeliness`. However, adding a description of the observation's associated error enables assessments of `accuracy`, and a description of the *feature of interest* allows the assessment of `relevance`. Furthermore, increasing the amount of contextual information enables quality assessment to consider more metadata while evaluating each quality dimension. For example, as part of the earlier `accuracy` example we could also explore observation provenance to identify where `accuracy` may be reduced (such as a change in coordinate system). To evaluate this, we will analyse the number of RDF triples used in assessing each quality dimension. Capturing the provenance of past quality assessments should enable us to improve the performance of future quality assessments through the re-use of existing quality results. We will determine if this is true by analysing the time taken to perform a new quality assessment with or without the provenance of past assessments. The data required by these evaluations will be collected by deploying our solution as part of a larger software infrastructure to address issues in the passenger information scenario[9] outlined earlier. This will enable us to evaluate how our solution performs with real data and real users. We aim to show that the use of our quality assessment framework enables a service to better select data for presentation to its users based on a number of quality rules. For example, the service in the passenger information scenario can evaluate quality to ensure that the sensor observations produced by GPS devices on public transport vehicles are accurate, timely, and relevant to the user.

Our approach will be deemed to be successful if we can demonstrate that it is possible to assess the quality of sensor observations by examining metadata describing their characteristics and provenance. A further indicator of success will be if the deployment of our quality assessment framework within the passenger information scenario can be shown to provide tangible benefits to users.

Acknowledgement. The research described here is supported by the award made by the RCUK Digital Economy programme to the dot.rural Digital Economy Hub; award reference: EP/G066051/1.

[9] `http://www.dotrural.ac.uk/irp/`

References

1. Bermudez, L., Graybeal, J., Arko, R.: A marine platforms ontology: Experiences and lessons. In: Proceedings of the Semantic Sensor Networks Workshop at the 5th International Semantic Web Conference (November 2006)
2. Bizer, C., Cygniak, R.: Quality-driven information filtering using the wiqa policy framework. Journal of Web Semantics 7, 1–10 (2009)
3. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. International Journal on Semantic Web and Information Systems 5, 1–22 (2009)
4. Botts, M., Percivall, G., Reed, C., Davidson, J.: Ogc sensor web enablement: Overview and high level architecture. In: Nittel, S., Labrinidis, A., Stefanidis, A. (eds.) Proceedings of Geosensor Networks 2006, pp. 175–190 (2008)
5. Furber, C., Hepp, M.: Swiqa - a semantic web information quality assessment framework. In: European Conference on Information Systems, p. 76 (2011)
6. Hartig, O., Zhao, J.: Using web data provenance for quality assessment. In: 1st Int. Workshop on the Role of Semantic Web in Provenance Management, vol. 526 (2009)
7. Kim, J., Kwon, H., Kim, D., Kwak, H., Lee, S.: Building a Service-Orient Ontology for Wireless Sensor Networks. In: 7th IEEE/ACIS International Conference on Computer and Information Science, pp. 649–654 (2008)
8. Le-Phouc, D., Hauswirth, M.: Linked open data in sensor data mashups. In: International Workshop on Semantic Sensor Networks 2009, vol. 552, pp. 1–16. CEUR (2009)
9. Lee, Y.W., Strong, D.M., Kahn, B.K., Wang, R.Y.: Aimq: a methodology for information quality assessmemt. Information and Management 40, 133–146 (2002)
10. Miles, S., Groth, P., Munroe, S., Moreau, L.: Prime: A methodology for developing provenance-aware applications. ACM Transactions on Software Engineering and Methodology 20(3), 39–46 (2009)
11. Page, K.R., Roure, D.C.D., Martinez, K., Sadler, J.D., Kit, O.Y.: Linked sensor data: Restfully serving rdf and gml. In: International Workshop on Semantic Sensor Networks 2009, vol. 522, pp. 49–63 (October 2009)
12. Patel-Schneider, P.F., Fensel, D.: Layering the Semantic Web: Problems and Directions. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 16–29. Springer, Heidelberg (2002)
13. Richardson, M., Agrawal, R., Domingos, P.: Trust Management for the Semantic Web. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 351–368. Springer, Heidelberg (2003)

Burst the Filter Bubble: Using Semantic Web to Enable Serendipity

Valentina Maccatrozzo

The Network Institute
Department of Computer Science
VU University Amsterdam, The Netherlands
v.maccatrozzo@vu.nl

Abstract. Personalization techniques aim at helping people dealing with the ever growing amount of information by filtering it according to their interests. However, to avoid the information overload, such techniques often create an over-personalization effect, *i.e.* users are exposed *only* to the content systems assume they would like. To break this "personalization bubble" we introduce the notion of *serendipity* as a performance measure for recommendation algorithms. For this, we first identify aspects from the user perspective, which can determine level and type of serendipity desired by users. Then, we propose a user model that can facilitate such user requirements, and enables serendipitous recommendations. The use case for this work focuses on TV recommender systems, however the ultimate goal is to explore the transferability of this method to different domains. This paper covers the work done in the first eight months of research and describes the plan for the entire PhD trajectory.

1 Research Problem

We are living the Information Age - previously unfindable or unreachable information is accessible instantly and the amount of it is constantly growing. Through personalization techniques we often get to see only the chunk that relates to our interests, preventing us from being overwhelmed. Various information providers typically gather user behavior and interests data to provide personalized recommendations, *e.g.* Amazon[1], Netflix[2]. However, such information filters have downsides too. On one hand, users are constantly missing something without noticing it, and, on the other, they are getting continuously the same type of recommendations. In 2011 Pariser [18] coined a new concept to describe this phenomenon: *the filter bubble*, *i.e.* personalization filters are building around us invisible barriers that keep away the content that does not fit completely with our profiles.

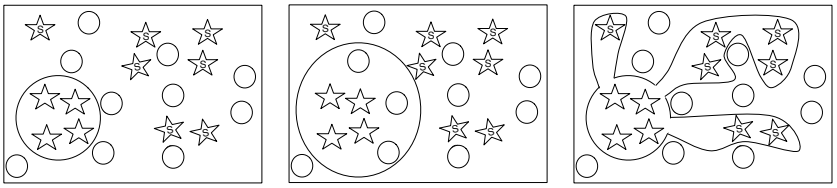
Think when you buy a book in a bookstore. You browse around the shelves letting titles and covers attract your attention. How many times it happens

[1] http://www.amazon.com
[2] http://www.netflix.com

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 391–398, 2012.

you found an interesting book on a shelve you look at only by chance? This is an *unexpected encounter*. The ability to make fortunate discoveries by accident is called *serendipity*. The word was coined by Horace Walpole in a letter he exchanged with Horace Mann in 1754 [24]. He describes serendipity as "*[...] making discoveries, by accidents and sagacity, of things which they were not in quest for [...]*". Personalization as we know it in the online bookstores does not allow this to happen anymore. It makes it difficult to discover what we did not know we were looking for.

This over-personalization problem cannot be solved by simply relaxing the filters, *i.e.* by keeping the bubble bigger, because of two reasons (see Fig. 1). First, browsing through irrelevant results in an online system is not as pleasurable as browsing through a physical store - the amount is way too big, and a bird-eye view is usually not available. Second, such approach does not account explicitly for the serendipity effect, *i.e.* as serendipity is subjective, the user model has to be able to surface items that are relevant but enough novel and diverse from the standard user interests.



(a) (b) (c)

Fig. 1. This example shows how over-personalization harms recommendations. In (1a) the circle indicates an over-personalized recommendation, which includes *only* core relevant items, (1b) shows a relaxed personalized recommendation, which contains *many* irrelevant items and (1c) shows the target recommendation, which contains *all* relevant items including the serendipitous ones.

While traditional personalization approaches focus mainly on getting results as close as possible to the user profile and do not account explicitly for the serendipity effect, more recently, a trend to focus more on approaches to get serendipitous results in recommendations [1,16,25] has developed.

2 Research Context

The focus of this work is on recommender systems in the TV domain (not only movies, but more about TV programs, *e.g.* talk shows, live shows, particular episodes of series). As described above, serendipity in the context of recommender systems is represented by a well balanced mix of diversity, novelty and relevance of the recommended items with respect to the users' interests. Thus, it can be measured only with respect to a given user profile, and the challenge

is how to determine the ideal distance from the user profile in a given context, that will be still relevant. Our proposal is to use Semantic Web techniques, in particular Linked Open Data (LOD), as a means to induce novel and relevant concepts in the user profile and thus explicitly support serendipity in recommender systems.

The rich link structure and the uniform representation make the LOD cloud a good candidate to explore for 'deep' and 'novel' connections between concepts. The LOD cloud can be seen as a structured knowledge space covering a multitude of different domains (many relevant to TV, *e.g.* music, books, movies, art), where each node in the graph is a separate knowledge element and the mechanism for discovery can be applied by creating bindings between different elements.

The goal of this research is to define and develop a method for an interactive recommendation approach, where the central novelty is the discovery and utilization of serendipitous bindings between the user profile and elements previously unlinked to it. We refer to such bindings as content patterns [19] - well connected concepts in one or across LOD datasets.

The requirements, as well as initial experiments with LOD patterns for this research have been gathered and performed during the NoTube project[3]. The next stage of this research will be performed in the context of the ViSTA-TV[4] European project. The consumers anonymized viewing behavior as well as the actual video streams from broadcasters and IPTV transmitters provided by ViSTA-TV will be used as training and test data for this PhD research. The ultimate goal is to integrate the serendipity-aware recommendation strategies together with a holistic live-stream data mining analysis in a personalized electronic program guide.

3 Research Questions

The central concept of this PhD research is serendipity and its utilization in serendipity-aware recommendation algorithms. Serendipity is typically an implicit user-subjective notion that is difficult to capture in objective terms. In order to realize it in a general recommendation approach we need to identify its objective characteristics in different user contexts, and define an explicit method to measure it. This guides our first research question:

> *Can we define a method to measure serendipity for individual content elements, as well as for the overall result of a recommendation system considering an explicit user profile? Which elements of this method are domain dependent and which could be generalized?*

As the serendipity level and its success should be assessed from a user perspective we use results from previous work, in the TV and cultural heritage domains, on identifying user needs and understanding of 'serendipity' as initial

[3] http://www.notube.tv
[4] http://www.vista-tv.eu

input for this research question. We have also explored several LOD sources, discovered relevant content patterns and analyzed their statistics as possible input for the serendipity measure. Further, user surveys in the context of the concrete ViSTA-TV use cases will be performed to gather additional requirements for the definition of serendipity and for the model to assess it. A number of experiments with the serendipity-aware recommendations will be needed in order to identify the optimal serendipity level in the different use cases. Finally, similar experiments will be performed in a different domain to investigate the cross-validity of our model.

The second challenge in this work relates to the use of LOD as a structured knowledge space to discover content patterns suitable to surface serendipitous recommendations. Considering the size of the LOD cloud and the diversity of domains, types of relationships and concepts it covers, keeping the right level of relevance in the recommendation results could be a tedious task. One way of addressing this issue could be through maintaining an up-to-date user context. Therefore, the second research question is:

> *Can we use social networks activities to form a continuously evolving and relevant context of the user interests? Can we map these user interests to LOD concepts in order to discover novel user interests through LOD content patterns?*

Results from previous and related research on social activities as input for a user profile were studied. An initial set of requirements for the user profile were derived. This set should be extended and finalized through experiments in the ViSTA-TV use cases, *i.e.* applying LOD browsing procedures (Section 4) guided by a user model. The NoTube mapping of LOD concepts to user interests is used as a baseline and further extended. Experiments will be performed to determine the impact of alternative user models and their LOD mappings on the serendipity level of the recommendation results and the user satisfaction.

What is serendipitous today, may not be true tomorrow, as it is with most of the user interests. In order to be sustainable over time, recommendation strategies need to account for the decay in user interests and changes in user context that determine the serendipity aspects. So, the third question is:

> *How does the time affect the serendipity function of a recommender system? What user feedback can help to determine a possible decay in the user interests?*

In order to measure the influence of time we need to perform long-term user tests monitoring the evolution of individual user interests, the context switching and the corresponding user feedback in the whole process. We envision comparing user profile states in different moments of time and applying a set of content patterns to analyze the differences in the serendipity perception.

In the next section we are discussing the overall approach to answering the research questions and implementing the solutions.

4 Approach

Our approach combines technologies from two fields, *i.e.* user modeling and semantic-based recommendation systems. According to André et al. [4], to induce serendipity we need a common language model, so that barriers between different fields can be removed and novel connections can be established. In other words, we need to express all the components involved in the same way. Thus, centrally to this approach is the *enrichment of our data with LOD concepts*. This includes both user activities, user interests and program metadata. The enriched data enables the alignment of concepts between the user profile and the program descriptions, and subsequently the querying for related users and programs, for example through analogy, metaphors, synonymy, homonymy, and hierarchy. Here we reuse existing metadata enrichment experiences in other domains, such as in cultural heritage for defining semantic search paths from experts behavior [14], for enriching museum metadata with historical events [23], and for recommendation-based browsing through museum collections [5]. In this project we use Web services, such as Lupedia[5], to realize the enrichment of the program metadata and the user activities.

The next major step in the approach is to *find the interesting paths* in these graphs (*i.e.* content patterns) that would lead to serendipitous recommendations. We identify three such 'routes' to serendipity, *i.e.* (1) variation & selection, (2) diverging & converging and (3) analogy.

Variation & Selection. According to Campbell [8], a combination of blind variation and selective retention of concepts is the process at the basis of creative thinking. We can apply this rationale to the querying of LOD sources by deriving new concepts from the ones that are present in the user-profile and then select those that are potentially serendipitous. The selection process needs to be trained by the feedback of the user, so that the serendipitous variations can be identified. In terms of content patterns: we select new concepts following a specific pattern, and if the feedback is positive we keep on applying it, otherwise we eliminate it. Once we have a list of serendipitous patterns, *i.e.* patterns that lead to serendipitous concepts, the identification of new ones is performed on the basis of their characteristics (*e.g.* same length, same predicates but different order).

Diverging & Converging. According to Guilford [13], divergent thinking is the capacity to consider different and original solutions to one problem and is the main component in the creativity process. Convergent thinking, instead, is the ability of bringing all the solutions together and elaborate a single one. Analogously, in querying the LOD we can first discover all possible paths starting from one node in the user profile (diverging phase). Then we can identify a new node that connects all (or the most of) these new concepts together (converging phase), and use it as a serendipitous candidate.

Analogy. According to Gentner [10], an analogy is a mapping of knowledge from one domain (the base) into another (the target). In other words, a system

[5] http://lupedia.ontotext.com

of relations that holds among the base objects also holds for the target objects. This process, called analogical mapping, is a combination of matching existing predicate structures and importing new predicates. Following the same reasoning, we can derive analogues LOD patterns using nodes (starting from the user profile) that share (the same or similar) predicates and exchange their predicates to define new connections.

5 Related Work

Serendipity has been recognized as an important component in many fields, such as scientific research [9], art [22] and humanistic research [20]. The main point of study, especially in creative thinking, has been the strive for understanding how different serendipitous encounters take place [7].

The role of serendipity in recommender systems has also been studied. Abbasi et al. [1] examine the over-specialization problem in recommenders. Similarly to our approach, they propose a system were items are grouped in regions and recommendations are built taking items also from regions under-exposed to the user. However, contrary to our approach, they do not exploit content semantics. Oku and Hattori [16] introduce serendipity in recommendations by selecting new items mixing the features of two user-input items. This approach measures serendipity only considering past activities of the users. This differs from our approach, that does not aim necessarily at improving accuracy with respect to other recommendation techniques, but improving the overall user experience. Zhang at al. [25] present a music recommender that combines diversity, novelty and serendipity of recommendation at a slightly cost of the accuracy.

On the side of semantic recommenders, Oufaida and Nouali [17] propose a multi-view recommendation engine that integrates collaborative filtering with social and semantic recommendation. They build users' profiles and neighborhoods with three dimensions: collaborative, socio-demographic and semantic. They show how semantics enhance precision and recall of collaborative filtering recommendations. However our approach aligns more with the work done in the CHIP project[6] on a content-based semantic art recommender, where [5] explores a number of semantic relationships and patterns that allow for introducing surprisingly interesting results. One of the aim addressed by researchers in the field of semantic recommender systems is the reliability and precision of the recommended items. To tackle this issue trust network have been used. For instance, Ziegler [26] proposes suitable trust metrics to build trust neighborhoods, and to make collaborative filtering approaches applicable to decentralized architecture, *i.e.* the Semantic Web. Golbeck and Hendler [12] propose a collaborative recommender system for movies, using FOAF [6] vocabulary as a base to build a social network of trust. An example of a semantic recommender for multimedia content is given by Albanese et al. [3] that computes customized recommendations using semantic contents and low-level features of multimedia objects, past behavior of individual users and behavior of the users community as a whole. The effectiveness of the approach is evaluated on the basis of user satisfaction.

[6] http://www.chip-project.org

Semantic user models to enhance personalized semantic search have been researched by Jiang and Tan [15]. They propose a user ontology model that utilizes concepts, taxonomic and non-taxonomic relations in a given domain ontology to capture the users interests. Ghosh and Dekhil [11], on the other hand, discuss accurate models of user profiles using Semantic Web technologies, by aggregating and sharing distributed fragments of user profile information spread over multiple services. Related to our proposal are also the semantic user modeling from social network. Abel et al. [2] introduce a framework for user modeling on Twitter which enriches the semantics of Twitter messages and identifies topics and entities mentioned in them and, similarly to van Aart at al. [21], shows how semantic enrichment enhances the variety and the quality of the generated user profiles.

6 Future Work and Conclusions

This PhD research is now approaching the second year. Current work involves analyzing specific techniques to select possible serendipitous patterns from different LOD datasets, namely *LinkedMDB*[7] and *DBpedia*[8]. We are also investigating different techniques of enrichment, exploring natural language processing methods. The plan for the near future is to start the users surveys to gather preliminary data about their serendipity perception. Afterwards, we will follow the steps presented in Section 4.

Acknowledgments. This research is supported by the FP7 STREP "ViSTA-TV" project, as well as partially supported by the FP7 IP "NoTube" project and the ONR Global NICOP "COMBINE" project.

References

1. Abbassi, Z., Amer-Yahia, S., Lakshmanan, L.V.S., Vassilvitskii, S., Yu, C.: Getting Recommender Systems to Think Outside the Box. In: RecSys 2009, pp. 285–288 (2009)
2. Abel, F., Gao, Q., Houben, G.-J., Tao, K.: Analyzing User Modeling on Twitter for Personalized News Recommendations. In: Konstan, J.A., Conejo, R., Marzo, J.L., Oliver, N. (eds.) UMAP 2011. LNCS, vol. 6787, pp. 1–12. Springer, Heidelberg (2011)
3. Albanese, M., d'Acierno, A., Moscato, V., Persia, F., Picariello, A.: A Multimedia Semantic Recommender System for Cultural Heritage Applications. In: ICSC 2011, pp. 403–410 (2011)
4. André, P., schraefel, mc., Dumais Teevan, S.T.: Discovery Is Never by Chance: Designing for (Un)Serendipity. In: C & C 2009, pp. 305–314 (2009)
5. Aroyo, L., Stash, N., Wang, Y., Gorgels, P., Rutledge, L.: CHIP Demonstrator: Semantics-Driven Recommendations and Museum Tour Generation. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 879–886. Springer, Heidelberg (2007)

[7] http://www.linkedmdb.com/
[8] http://www.dbpedia.org/

6. Brickley, D., Miller, L.: FOAF Vocabulary Specification 0.97. Namespace document, W3C (January 2010)
7. Chaomei, C.: Turning Points. The Nature of Creative Thinking. Springer (2011)
8. Campbell, D.T.: Blind Variation and Selective Retention in Creative Thought as in Other Knowledge Processes. Psychological Review 67, 380–400 (1960)
9. Garcia, P.: Discovery by Serendipity: a new context for an old riddle. Foundations of Chemistry 11, 33–42 (2009)
10. Gentner, D.: The mechanisms of analogical learning. In: Vosniadou, S., Ortony, A. (eds.) Similarity and Analogical Reasoning, pp. 199–241. Cambridge University Press (1989)
11. Ghosh, R., Dekhil, M.: Mashups for semantic user profiles. In: WWW 2008, pp. 1229–1230 (2008)
12. Golbeck, J., Hendler, J.: FilmTrust: movie recommendations using trust in web-based social networks. In: CCNC 2006, pp. 282–286 (2006)
13. Guilford, J.P.: The Nature of Human Intelligence. McGraw-Hill, New York (1967)
14. Hildebrand, M., van Ossenbruggen, J.R., Hardman, H.L., Wielemaker, J., Schreiber, G.: Searching In Semantically Rich Linked Data: A Case Study In Cultural Heritage. Technical Report INS-1001, CWI (2010)
15. Jiang, X., Tan, A.: Learning and inferencing in user ontology for personalized Semantic Web search. Information Sciences 179(16), 2794–2808 (2009)
16. Oku, K., Hattori, F.: Fusion-based Recommender System for Improving Serendipity. In: DiveRS 2011, pp. 19–26 (2011)
17. Oufaida, H., Nouali, O.: Exploiting Semantic Web Technologies for Recommender Systems: A Multi View Recommendation Engine. In: ITWP 2009 (2009)
18. Pariser, E.: The Filter Bubble. What the Internet is hiding from you. Penguin Press HC (2011)
19. Presutti, V., Aroyo, L., Adamou, A., Schopman, B., Gangemi, A., Schreiber, G.: Extracting Core Knowledge from Linked Data. In: COLD 2011 (2011)
20. Quan-Haase, A., Martin, K.: Digital Humanities: the continuing role of serendipity in historical research. In: iConference 2012, pp. 456–458 (2012)
21. van Aart, C., Aroyo, L., Brickley, D., Buser, V., Miller, L., Minno, M., Mostarda, M., Palmisano, D., Raimond, Y., Schreiber, G., Siebes, R.: The NoTube Beancounter: Aggregating User Data for Television Programme Recommendation. In: SDoW 2009 (2009)
22. van Andel, P.: Anatomy of the Unsought Finding. Serendipity: Origin, History, Domains, Traditions, Appearances, Patterns and Programmability. The British Journal for the Philosophy of Science 45(2), 631–648 (1994)
23. van Erp, M., Oomen, J., Segers, R., van de Akker, C., Aroyo, L., Jacobs, G., Legêne, S., van der Meij, L., van Ossenbruggen, J.R., Schreiber, G.: Automatic Heritage Metadata Enrichment With Historic Events. In: MW 2011 (2011)
24. Walpole, H.: To Mann, Monday 18 January 1754. In: Lewis, W.S. (ed.) Horace Walpole's Correspondence, vol. 20, pp. 407–411. Yale University Press (1960)
25. Zhang, Y.C., Séaghdha, D., Quercia, D., Jambor, T.: Auralist: Introducing Serendipity into Music Recommendation. In: WSDM 2012, pp. 13–22 (2012)
26. Ziegler, C.-N.: Semantic Web Recommender Systems. In: Lindner, W., Fischer, F., Türker, C., Tzitzikas, Y., Vakali, A.I. (eds.) EDBT 2004. LNCS, vol. 3268, pp. 78–89. Springer, Heidelberg (2004)

Reconstructing Provenance*

Sara Magliacane

Department of Computer Science
VU University Amsterdam
s.magliacane@vu.nl

Abstract. Provenance is an increasingly important aspect of data management that is often underestimated and neglected by practitioners. In our work, we target the problem of reconstructing provenance of files in a shared folder setting, assuming that only standard filesystem metadata are available. We propose a content-based approach that is able to reconstruct provenance automatically, leveraging several similarity measures and edit distance algorithms, adapting and integrating them into a multi-signal pipeline. We discuss our research methodology and show some promising preliminary results.

1 Problem Statement

The provenance of a data item is the metadata describing how, when and by whom the data item was produced. Provenance information is crucial for many applications, from data quality and aggregation to trust, and it has been researched from several perspectives (see surveys [8,11,20]).

In science, provenance helps scientists reproduce and repeat experiments. In business, understanding who made a decision, produced a document, or designed a product allows for effective accountability. However, since tracking provenance requires effort, it is often not done in real-world settings, resulting in collections of files with only basic metadata, e.g. timestamps. Thus, addressing these use cases becomes difficult or impossible.

In our work, we target the problem of reconstructing provenance of data in a shared folder setting, in which several authors can create or edit the data at different moments, and only standard filesystem metadata are available. Some of the data in the folder have been created by a sequence of operations on other data. The research questions we wish to answer are the following:

> How can one automatically, accurately and efficiently reconstruct the provenance of data in a shared folder, intended as the sequences of operations connecting the data?

A desirable solution should reconstruct provenance across multiple data types. It should be applicable also without domain-specific knowledge, while improving its accuracy in case this knowledge is available. Efficiency is intended both in

* Advisors: Paul Groth, Frank van Harmelen

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 399–406, 2012.

terms of run-time performance and scalability. An additional desirable property would be to produce the results with an anytime strategy, i.e. returning an approximated output at any time of the computation, in which the accuracy increases the longer we wait.

2 Related Work

As we pointed out in [15], recently the issue of missing or incomplete provenance has attracted the attention of the provenance community and lead to few initial attempts to address this problem. On the other side, there are also several other fields that face similar problems and propose approaches that could be adapted to reconstructing provenance.

Table 1. Classification matrix of the related work

Related Work	Reconstruction	Entities	Operations	Required knowledge
Provenance in reservoir engineering [26]	Generating Process	Instances of concepts	Processes in the system	Previous executions
Provenance in network setting [14,2]	Dependency	Nodes	Sending information	Network structure
Provenance in stream processing [18]	Sequence	Tuples in data streams	Processes in the system	Coarse-grained provenance
Monitoring at OS-level [16,12]	Sequence	Application data	Application	OS-level reads and writes
Provenance as data mining [9]	Dependency	Text	Any on text	None
Provenance discovery using semantic similarity [22]	Sequence	Named Entities in Documents	Replacement, Generalization, Specialization, Addition, Omission	None
Text-reuse [6,4]	Dependency	Text	Any on text	None
Image Mining for Historical Manuscripts [17]	Dependency (same manuscript)	Images of historical manuscripts	Distortions on images	Library of known images
Edit distance [5,13]	Sequence	Strings, trees and graphs	Few and simple	None
Change detection [7]	Sequence	Hierarchically structured data	Few and simple	None
Ontology change detection [23]	Sequence	Ontologies	Low-level operations are similar to graphs	Rules for inferring high level changes
Web Service Composition [3,24]	Sequence based on user requirements	Inputs and Outputs	Web Services	Formal description of Web Services
Learn data transformations [25]	Sequence	Instances of semantic types	Any defined by grammar	Grammars of operations, More examples
Workflow Mining [1]	Sequence	Inputs and Outputs	Workflow components	Execution Traces

In Table 1, we take a broad view of reconstructing provenance and present a classification of the related work, listing in some cases only few representative examples of a field. The type of provenance that is reconstructed (column Reconstruction in Table 1) between the entities (column Entities in Table 1) can be:

- Dependency - the dependency relationship between two entities;
- Sequence - the sequence of operations that connect two entities;
- Generating Process - the process which created the entity;

The type of entities involved in the reconstruction ranges from text to data structures like graphs and ontologies. The type of operations involved in the reconstruction varies accordingly from simple operations, like inserting a node in a graph, to an arbitrary complex operation as a web service. Finally, we have classified also the required knowledge in the case of each related work.

As can be seen from Table 1, most of the approaches in the provenance literature [26,14,2,18,16,12] require a lot of knowledge, leveraging the network structure or execution environment. There are two exceptions: Deolalikar et al. [9] who reconstruct dependency chains of documents using a basic text similarity metric, and Nies et al. [22] who reconstruct sequences of a limited set of operations on Named Entities in documents using semantic similarity, i.e. the cosine similarity of vectors of Named Entities contained in each document. Both of these approaches offer a partial solution to the problem of reconstructing provenance, since they consider only one type of entities (text) and few operations.

More refined similarity measures are used in the context of text-reuse (e.g. [6,4]) in order to detect content reuse between documents, which can be seen as a type of dependency relationship. There are also approaches that use image similarity to reconstruct dependencies between documents, e.g. Hu et al. [17], who consider several electronic versions of historical manuscripts. There exists extensive research on reconstructing sequences of operations based on input and output data, but either the entities and operations involved are very simple [5,13,7] or they are tailored to a specific situation [23]. Other approaches require a lot of knowledge, either a formal description of the operations and user requirements on the composite operation (e.g. [3,24]), grammars of edit operations and a number of examples [25], or execution traces for several executions [1].

While there is extensive related work, this specific problem is only beginning to be addressed in the provenance community (see [9,22]), thus there are still a wide variety of open issues and improvements to be made.

3 Proposed Approach

We propose a content-driven approach that reconstructs provenance using the contents of the files. Inspired by the DeepQA approach of IBM Watson [10], we aim at developing a multi-signal pipeline, which will combine several signals representing evidence on the relationships between files and propose a ranked list of plausible reconstructions. Our multi-signal pipeline consists of four stages, each containing several components that can be executed in parallel:

1. **Preprocessing phase:** contains the components that extract all available metadata from the files, infer the semantic types of the data, preprocess the content and index it in order to speed up the following phases.

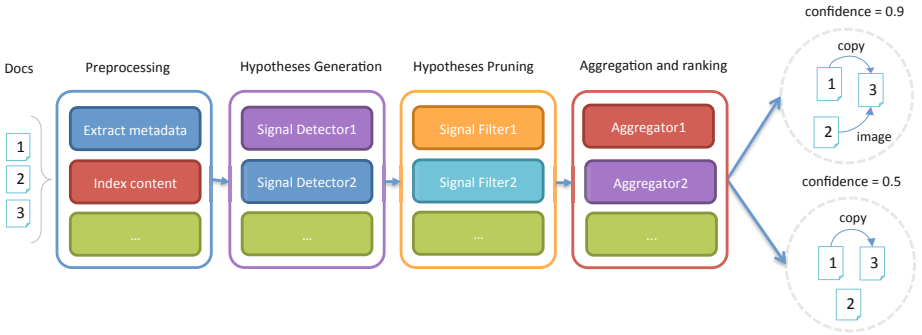


Fig. 1. Multi-signal pipeline for reconstructing provenance

2. **Hypotheses generation phase:** contains several *Signal Detectors*, which gather evidence of possible relationships between all the documents, generating several hypothesis graphs, that are expressed in the PROV-DM model [21]. Signal Detectors can be implemented using a number of existing techniques, from change detection algorithms to various types of similarity measures for different types of entities, e.g. text-reuse measures [4], image similarity [19] or semantic similarity [22].

   If domain-specific knowledge is available, we can integrate it into one or more Signal Detectors. Moreover, if there is a library of possible operations, an AI planning technique similar to [15] can be employed, parametrized with the appropriate domain-specific heuristics.

   The semantic type of data from the previous phase helps in deciding which Signal Detector to use. In order to speed up the evaluation, the cheapest Signal Detectors are executed first.
3. **Hypotheses pruning phase:** contains several *Signal Filters* that prune inconsistent or non-relevant hypotheses. One example is the Signal Filter that prunes temporally inconsistent edges in the hypotheses graphs or Signal Filters that enforce domain-specific rules triggered by the semantic type of the data. For example, if we are comparing two patient records, there could be a domain-specific rule that defines that two records can refer to the same patient only if they have the same identifier. For each hypotheses graph, the system executes all relevant Signal Filters in a cascade, but being independent one from the other, their order is not important. Therefore, we can devise a scheduling algorithm to parallelize their execution.
4. **Aggregate and ranking of hypotheses phase:** contains several *Aggregators* that aggregate the hypotheses, each with a confidence value that is based on the semantic type of data, e.g. a domain-specific aggregator has a greater confidence than a general one.

There are several challenges in this approach. The first major challenge involves finding the appropriate components for each of the phases in order to have results that are accurate enough for a broad range of domains and types of entities. We address this challenge by researching existing approaches in literature and

integrating them in our pipeline. Moreover, we plan to integrate some simple domain-specific components (e.g. for dealing with bio-medical publications).

Another important challenge is computational efficiency, due to the large number of components, which are possibly already computationally expensive. We propose to address this issue by parallelizing the execution of components as much as possible, and schedule the cheapest components in each phase first. Moreover, all the components should feed their outputs, i.e. the hypotheses graphs, to the next phase as soon as they are ready. The Aggregator components, which need to aggregate several hypotheses graphs, should implement an anytime strategy that is able to give an approximation of the results based on its inputs, and gets more and more refined as there are more inputs.

A possible approach that we are considering involves using some of the computationally cheaper Signal Detectors as an approximation of the dependencies between entities, in order to suggest which pairs of entities are more promising to be compared.

4 Planned Research Methodology

To address the reconstructing provenance problem, we will follow an iterative process and we will incrementally build a framework for reconstructing provenance. In particular, we will use an empirical approach, in which each iteration will be guided by the results of the evaluation of the previous iteration. Each iteration of the process will consist of three phases.

The first phase will be focused on analyzing the state of the art approaches in literature, that could be compatible and complementary to the ones already present in our framework.

In the second phase, we will adapt and integrate one of these approaches into a framework, possibly reusing existing open-source software.

In the third phase, we will evaluate the performance of the system, both in terms of correctness of predictions and computational efficiency, on benchmark corpora. In case there are no corpora available, we will construct one, either automatically or manually, depending on the case. In the evaluation and testing phase, we will follow and adapt the standard IR and NLP approaches.

5 Preliminary Results

The first approach for reconstructing provenance we devised was inspired by AI planning techniques and change detection algorithms, as described in [15]. The goal of this work was to reconstruct the sequence of transformations between entities by using the A* algorithm combined with a heuristic function based on the edit distance. In this case, there are three limitations :

- we need to define the library of possible operations;
- we need to define the heuristics;
- for each entity, we have to compute the edit distance - an expensive algorithm- for all entities, not only the more promising entities.

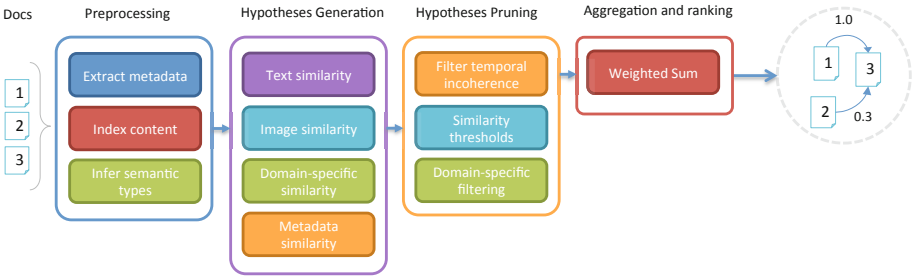


Fig. 2. Current system architecture

Therefore, we developed a complementary approach based on our multi-signal pipeline. As a first step, we considered the simpler problem of reconstructing provenance intended as dependencies between entities. There has been some prior work addressing reconstructing provenance as dependencies using text similarity (e.g., [9]), which we expanded by considering several multi-modal similarity measures. The setting we considered is reconstructing dependencies among a set of documents of different types (including images, Latex files, PDFs, MS Office documents, etc.) in a shared Dropbox folder.

We implemented a first prototype of the multi-signal pipeline by taking advantage of existing libraries and frameworks. Currently, all execution is sequential and we have not yet developed the anytime behavior.

In the Preprocessing stage, the system gets all available versions and metadata using the Dropbox API [1], extracts content (both text and images) and other metadata using Apache Tika[2]; and indexes the text using Apache Lucene[3] and images using LIRE [19].

We implemented four Signal Detectors: 1) text similarity using Lucene; 2) metadata similarity using SimMetrics[4]; 3) image similarity using LIRE [19]; 4) a simple domain-specific similarity, e.g. the exact match of the document name in the content.

We developed two Signal Filters: 1) filter dependencies using temporal information, e.g. a document in the past cannot depend on a document in the future; 2) filter dependencies with a score lower than a specified threshold;

The Aggregator we implemented is a weighted average of all the scores from the Signal Detectors and output a PROV-DM [21] graph using the Prov-toolbox[5].

We evaluated the prototype in a preliminary experiment on a Dropbox folder containing all data for a workshop paper, where the provenance of the files in the folder was manually annotated. With respect to our baseline, i.e. the approach described in [9], which uses only text similarity, our approach that combines

[1] https://www.dropbox.com/developers/reference/sdk
[2] http://tika.apache.org/
[3] http://lucene.apache.org/
[4] http://sourceforge.net/projects/simmetrics/
[5] https://github.com/lucmoreau/ProvToolbox

multi-modal similarities is able to increase the precision from 0.57 to 0.63 and the recall from 0.65 to 0.80, showing that even a simple approach can lead to significant improvements. More details on our experiment can be found in the Technical Report[6].

6 Conclusions and Future Work

In this paper, we have described the problem of reconstructing provenance, introducing a possible approach to address it using a multi-signal pipeline. The results we had obtained on the small test pilot are encouraging and we are currently creating a corpus for a more extensive evaluation.

The next step we will take is to implement the parallel and anytime behavior suggested in the proposed approach. Then we will extend the prototype with additional Signal Detectors, like text-reuse similarity measures (e.g. [4]), semantic similarity [22], normalized compression distance or other domain-specific Detectors as citation analysis techniques. We also plan to add domain-specific Signal Filters and to implement more Aggregators, possibly by using a supervised learning to assign the weights to the different Signal Detector scores. Moreover, we will integrate and adapt the approach presented in [15].

Acknowledgements. This publication was supported by the Data2Semantics project in the Dutch national program COMMIT.

References

1. van der Aalst, W., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G.: Workflow mining: A survey of issues and approaches. Data & Knowledge Engineering 47(2), 237–267 (2003)
2. Barbier, G., Liu, H.: Information Provenance in Social Media. In: Salerno, J., Yang, S.J., Nau, D., Chai, S.-K. (eds.) SBP 2011. LNCS, vol. 6589, pp. 276–283. Springer, Heidelberg (2011)
3. Baryannis, G., Plexousakis, D.: Automated Web Service Composition: State of the Art and Research Challenges. Tech. Rep. October, Tech. Rep. 409, ICS-FORTH (October 2010)
4. Bendersky, M., Croft, W.B.: Finding text reuse on the web. In: Proceedings of the Second ACM International Conference on Web Search and Data Mining (2009)
5. Bille, P.: A survey on tree edit distance and related problems. Theoretical Computer Science 337(1-3), 217–239 (2005)
6. Broder, A.Z.: On the resemblance and containment of documents. In: Compression and Complexity of Sequences, SEQUENCES 1997 (1997)
7. Chawathe, S., Garcia-Molina, H.: Meaningful change detection in structured data. ACM SIGMOD Record, 26–37 (1997)
8. Cheney, J., Chiticariu, L., Tan, W.C.: Provenance in databases: Why, how, and where. Found. Trends Databases 1, 379–474 (2009)

[6] http://www.few.vu.nl/~sme340/papers/techreport.pdf

9. Deolalikar, V., Laffitte, H.: Provenance as data mining: combining file system metadata with content analysis. In: First Workshop on Theory and Practice of Provenance, p. 10. USENIX Association (2009)
10. Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A.A., Lally, A., Murdock, J.W., Nyberg, E., Prager, J., Schlaefer, N., Welty, C.: Building Watson: An overview of the DeepQA project. AI Magazine 31(3), 59–79 (2010)
11. Freire, J., Koop, D., Santos, E., Silva, C.T.: Provenance for computational tasks: A survey. Computing in Science and Engg. 10, 11–21 (2008)
12. Frew, J., Metzger, D., Slaughter, P.: Automatic capture and reconstruction of computational provenance. Concurrency and Computation: Practice and Experience 20(5), 485–496 (2008)
13. Gao, X., Xiao, B., Tao, D., Li, X.: A survey of graph edit distance. Pattern Analysis and Applications 13(1), 113–129 (2009)
14. Govindan, K., Wang, X., Khan, M., Dogan, G., Zeng, K., Davis, C.: PRONET: Network Trust Assessment Based on Incomplete Provenance. In: IEEE The Premier International Conference for Military Communications (2011)
15. Groth, P., Gil, Y., Magliacane, S.: Automatic Metadata Annotation through Reconstructing Provenance. In: ESWC (2012)
16. Holland, D.A., Seltzer, M.I., Braun, U., Muniswamy-Reddy, K.K.: Passing the provenance challenge. Concurrency and Computation: Practice and Experience 20(5), 531–540 (2008)
17. Hu, B., Rakthanmanon, T., Campana, B., Mueen, A., Keogh, E.: Image mining of historical manuscripts to establish provenance. In: SIAM Conference on Data Mining, SDM (2012)
18. Huq, M.R., Wombacher, A., Apers, P.M.G.: Inferring Fine-Grained Data Provenance in Stream Data Processing: Reduced Storage Cost, High Accuracy. In: Hameurlain, A., Liddle, S.W., Schewe, K.-D., Zhou, X. (eds.) DEXA 2011, Part II. LNCS, vol. 6861, pp. 118–127. Springer, Heidelberg (2011)
19. Lux, M., Chatzichristofis, S.A.: Lire: lucene image retrieval: an extensible java cbir library. In: Proceedings of the 16th ACM International Conference on Multimedia, pp. 1085–1088 (2008)
20. Moreau, L.: The foundations for provenance on the web. Found. Trends Web Sci. 2, 99–241 (2010)
21. Moreau, L., Missier, P.: PROV-DM: The PROV Data Model, http://www.w3.org/TR/prov-dm/
22. Nies, T.D., Coppens, S., Deursen, D.V., Mannens, E., Walle, R.V.D.: Automatic Discovery of High-Level Provenance using Semantic Similarity. In: IPAW 2012 (2012)
23. Noy, N.F., Kunnatur, S., Klein, M., 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)
24. Rao, J., Su, X.: A Survey of Automated Web Service Composition Methods. In: Cardoso, J., Sheth, A.P. (eds.) SWSWPC 2004. LNCS, vol. 3387, pp. 43–54. Springer, Heidelberg (2005)
25. Wu, B., Szekely, P., Knoblock, C.A.: Learning data transformation rules through examples: Preliminary results. In: Ninth International Workshop on Information Integration on the Web, IIWeb 2012 (2012)
26. Zhao, J., Gomadam, K., Prasanna, V.: Predicting Missing Provenance using Semantic Associations in Reservoir Engineering. In: 2011 Fifth IEEE International Conference on Semantic Computing (ICSC), pp. 141–148. IEEE (2011)

Very Large Scale OWL Reasoning through Distributed Computation*

Raghava Mutharaju

Kno.e.sis Center,
Wright State University, Dayton, Ohio
raghava@knoesis.org

Abstract. Due to recent developments in reasoning algorithms of the various OWL profiles, the classification time for an ontology has come down drastically. For all of the popular reasoners, in order to process an ontology, an implicit assumption is that the ontology should fit in primary memory. The memory requirements for a reasoner are already quite high, and considering the ever increasing size of the data to be processed and the goal of making reasoning Web scale, this assumption becomes overly restrictive. In our work, we study several distributed classification approaches for the description logic EL+ (a fragment of OWL 2 EL profile). We present the lessons learned from each approach, our current results, and plans for future work.

1 Introduction

Over the years, the efficiency of classification algorithms for the description logic $\mathcal{EL}^+$has constantly improved [3,10,11], so much so that, ELK reasoner [11] can classify SNOMED CT[1], one of the largest biomedical ontologies in 5 seconds. But the improvement has been only in runtime and not space. In a recent study on the performance of reasoners [8], it was noted that, in tableau-based reasoners, memory exhaustion is a known problem. So, in this scenario, performing in-memory computations on a single machine would be problematic for ontologies larger than SNOMED CT.

The amount of available data is always on the rise. We would not be off the mark in saying that there would be ontologies bigger than SNOMED CT very soon. In fact, there is a biomedical ontology named LinkBase, which is thrice the size of SNOMED CT [26,17]. There could be even more bigger ontologies, especially, ontologies with large ABoxes. Even if we consider that the RAM prices are cheap and that might solve the issue, in order to really perform OWL reasoning at Web scale, the current infrastructure that the reasoners are based on, is not sufficient. In this scenario, there is a good possibility of falling short on both memory and computation power. This is where our work on distributed OWL reasoning algorithms is expected to bridge the gap.

* Supervisor: Pascal Hitzler.
[1] Can be obtained from http://ihtsdo.org

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 407–414, 2012.

Normal Form	Completion Rule
$A_1 \sqcap \cdots \sqcap A_n \sqsubseteq B$	**R1** If $A_1, \ldots, A_n \in S(X)$, $A_1 \sqcap \cdots \sqcap A_n \sqsubseteq B \in \mathcal{O}$, and $B \notin S(X)$ then $S(X) := S(X) \cup \{B\}$
$A \sqsubseteq \exists r.B$	**R2** If $A \in S(X)$, $A \sqsubseteq \exists r.B \in \mathcal{O}$, and $(X, B) \notin R(r)$ then $R(r) := R(r) \cup \{(X, B)\}$
$\exists r.A \sqsubseteq B$	**R3** If $(X, Y) \in R(r)$, $A \in S(Y)$, $\exists r.A \sqsubseteq B \in \mathcal{O}$, and $B \notin S(x)$ then $S(X) := S(X) \cup \{B\}$
$r \sqsubseteq s$	**R4** If $(X, Y) \in R(r)$, $r \sqsubseteq s \in \mathcal{O}$, and $(X, Y) \notin R(s)$ then $R(s) := R(s) \cup \{(X, Y)\}$
$r \circ s \sqsubseteq t$	**R5** If $(X, Y) \in R(r)$, $(Y, Z) \in R(s)$, $r \circ s \sqsubseteq t \in \mathcal{O}s$, $(x, Z) \notin R(t)$ then $R(t) := R(t) \cup \{(X, Z)\}$

Fig. 1. Normal forms and Completion rules in CEL

The remainder of the paper is organized as follows. Section 2 contains some preliminaries. In Section 3, we mention the previous work and how our work differs from it. In Section 4, we present our approaches that we have taken towards tackling this problem and in Section 5, we present our preliminary results followed by our planned work for the future.

2 Preliminaries

Concepts in description logic $\mathcal{EL}^+$are formed according to the grammer

$$C ::= A \mid \top \mid C \sqcap D \mid \exists r.C,$$

where A ranges over concept names, r over role names, and C, D over (possibly complex) concepts. Ontology in $\mathcal{EL}^+$is a finite set of *general concept inclusions* (*GCIs*) $C \sqsubseteq D$ and *role inclusions* (*RIs*) $r_1 \circ \cdots \circ r_n \sqsubseteq r$, where C, D are concepts, n is a positive integer and $r, r_1, \ldots, r_n$ are role names. For the semantics of $\mathcal{EL}^+$please refer [2].

Classification of an ontology is the computation of the complete subsumption hierarchy between all concept names occurring in the ontology. Classification is one of the standard reasoning tasks. Among others, CEL algorithm [4] performs classification of an $\mathcal{EL}^+$ontology. It uses the completion rules in Figure 1. It requires the ontology to be in normal form, where all the axioms should be in one of the forms shown in the left part of Figure 1.

3 Related Work

In order to make reasoning Web scale, algorithms should be scalable. To that extent, various parallel and distributed approaches for classification of OWL fragments and closure of RDFS have been explored. Harmelen et al. use MapReduce

and peer-to-peer network for large scale RDFS reasoning [18,24]. They extend their work to OWL Horst fragment in [7]. Many of their optimization techniques from their work on RDFS reasoning could not be carried over to OWL Horst due to the increased complexity of rules in OWL Horst. As the expressivity increases, the rules as well as the pre-conditions in the rules would be increasingly complex. An embarrassingly parallel algorithm is used in [27] for computing the RDFS closure. In [9], distributed hash tables were used for the computation of RDFS closure. Soma et al. [23] investigate two partitioning approaches for parallel inferencing in OWL Horst. In [25], backward chaining is used to scale up to a billion triples in the OWL Horst fragment. Distributed reasoning of fuzzy OWL Horst has also been investigated in [15].

Stuckenschmidt et al. have used resolution techniques in distributed settings to achieve scalability of various OWL fragments such as $\mathcal{ALC}$ [20] and $\mathcal{ALCHIQ}$ [21]. There have been attempts at achieving distributed reasoning on $\mathcal{EL}^+$profile in [16] and [22], but they do not provide any experimental results. Distribution of OWL EL ontologies over a peer-to-peer network and algorithms based on distributed hash table have been attempted in [5], but they do not provide any evaluation results.

There have also been some successful attempts at making use of the multiple cores on a single machine in order to speed up the classification of ontologies. Haarslev et al. have worked on parallel tbox classification [1] and parallel tableau based description logic reasoner for $\mathcal{ALC}$ [28]. In [13], the authors parallelized the non-deterministic choices inherent in tableau algorithms. Parallelization of tableau algorithm, for $\mathcal{SHIQ}$ has also been attempted in [14], but they haven't provided any evaluation results. In [11], the authors use multi-threading and consequence-based procedure to achieve highly optimized classification runtime. The authors of [19] extend the approach in [11] to parallel ABox reasoning. They were able to compute all ABox entailments for an ontology having 1 million individuals in 3 minutes. But, for this, they require an unreasonably high memory of 60GB on an 8 core processor. With concurrent approaches, it would be possible to improve the efficiency of the classification algorithm, but it would not be possible to achieve scalability. For Web scale reasoning and for very large ontologies, these approaches would suffer from the same memory constraints that were highlighted in Section 1.

Compared to the above approaches, the authors of [6] take a different route. They focus on using secondary memory for classification of $\mathcal{ELH}$ ontologies and were able to classify SNOMED CT in 20 minutes and the RAM used for computations is only 32MB. But this approach lacks the parallelism demonstrated in other approaches. Please note that many of the fragments mentioned here are different from the one that we are interested in, which is $\mathcal{EL}^+$. But this section highlights some of the existing scalable reasoning approaches. For reasonably expressive OWL profiles, we wish to explore the distribution of axioms of the ontology across the cluster and perform parallel computations. We explain our approach further in the next section.

4 Research Problem and Approaches

4.1 Research Problem

Our research problem can be broken down into the following two questions

1. What are the approaches for distributed reasoning of OWL reasoning algorithms; specifically, for profiles $\mathcal{EL}^+$and higher?
2. Demonstrate the need and the validity of the approach for distributed reasoning on a real world use case.

4.2 Research Plan

Our research plan for the above two questions is as follows

Step 1. Start with a relatively less expressive description logic such as $\mathcal{EL}^+$. Explore distributed reasoning approaches for this profile.

Step 2. Choose the distributed reasoning approach which is most appropriate and extend it to more expressive profiles such as $\mathcal{EL}^{++}$ [2] and $\mathcal{SROELV}_n(\sqcap, \times)$ [12]. Note that this step might not be a straightforward extension of step 1. It might require additional optimizations and further research.

Step 3. There is an ongoing work in our research center where the Semantic Web Journal website[2] is being upgraded to Drupal 7. The purpose of the upgrade is to have access to the Semantic Web extensions of Drupal 7. If not already present, we plan on developing an OWL reasoner module for Drupal and integrate the distributed reasoning work into it. The Semantic Web Journal website would be backed by an ontology and website content would be annotated appropriately. The website has a constant flow of submissions and by having a reasoner support, we plan on providing semantic search, semantic browsing and semantic content creation. Apart from the journal website content, the reasoner would also access appropriate datasets from Linked Open Data (LOD) cloud. The number of submissions for the journal website as well as the size of LOD cloud keep increasing. So we believe that this would be a very good application to demonstrate the need of having a distributed reasoner.

4.3 Approaches

All the approaches presented are for description logic $\mathcal{EL}^+$. Approaches can be categorised into distributed memory and shared memory.

[2] http://www.semantic-web-journal.net

4.3.1 Distributed Memory

MapReduce. Our first attempt was to use the popular distributed framework, MapReduce, for computing classification of $\mathcal{EL}^+$ontologies [16]. We revised the CEL algorithm [4] to suit the key-value format of the data required for MapReduce. In the Map phase, preconditions of the rules are checked and in the Reduce phase, conclusion of the rules are computed. Pros and cons of this approach are given below.

Pros

- Parallelization can be achieved easily.
- Fault tolerance is handled by the framework.

Cons

- In each iteration, duplicates are generated. This makes termination detection hard.
- MapReduce is not suitable if there are dependencies between the data chunks. In CEL completion rules, some of the rules are interdependent.
- It is difficult to filter data in subsequent iterations. For example, ideally, in the next iteration, the algorithm needs to run only on the newly generated data (compared to last iteration).

Distributed Queue. In MapReduce, nodes in the cluster cannot talk to each other. Since there are dependencies among the data chunks, there would be a need for the nodes to talk to each other. Due to this, we replaced map and reduce methods with our custom methods which can talk to other nodes, when required. We also replaced HDFS with a distributed key-value data store. CEL algorithm implementation makes use of a queue mechanism [4] to trigger rule execution. In distributed queue approach, the idea is to take this queue implementation and spread the load across the cluster. Axioms are distributed across the cluster and the queue implementation runs on each node of the cluster. So each node acts as a stand-alone reasoner, which talks to other nodes when required. This approach was not as efficient as we expected it to be due to the following reasons.

- There was a lot of cross communication among the nodes.
- Large ontologies like SNOMED CT generate many R(r)s which makes rule R3 in [4] very slow. This rule slows down the entire operation across the cluster.

Distributed Completion Rules. Instead of distributing the axioms and the queues randomly, we distributed the axioms based on their type. Based on the normal form type [4], each axiom in an ontology can be placed under one of the five types. Now, each node is dedicated to only one type of normal form and runs an appropriate rule on the axioms. Compared to the distributed queue approach, this approach has the advantage of isolating the slowest rule and not letting it affect the processing of other rules. Furthermore, we have split rule R3 into two rules, R3-1 and R3-2 as mentioned in [16]. These two rules run in parallel on separate nodes of the cluster. In order to reduce the

cross communication, we use fixpoint iteration instead of the queue algorithm to process the completion rules. This makes termination detection harder, because, we need to be able to detect that there is no new output across the cluster. This approach was efficient compared to our previous approaches. Some preliminary results are given in Section 5.

4.3.2 Shared Memory

Multi-threaded graph. Apart from the distributed approaches mentioned before, we have also tried shared memory approach. The idea here is to represent all the axioms as a graph and perform parallel traversals[3]. Concepts are represented as nodes and the relationship between concepts as edge. Unlabelled edges represent subclass relation and labelled edges represent role name. This work was done on Cray XMT[4], a massively parallel supercomputer with shared memory architecture.

After representing axioms as graphs, classification would be reduced to the problem of computing transitive closure in the graph with respect to the subclass relation. Cray XMT compiler generates parallelizable version of the code based on the hints that the programmer places in the code. Although Cray XMT provides huge computing power, if there are data dependencies in the code, it is difficult to parallelize that part of the code. We were unable to parallelize the compute intensive parts of the code due to these dependencies. Apart from this, issues like synchronization, deadlocks, hot spots need to be handled by the programmer. Overall, Cray XMT has a steep learning curve and resolving data dependencies is not straightforward. Due to this, the vast computing power of the supercomputer could not be utilized properly.

5 Results and Future Work

Except distributed completion rule approach, none of the other approaches work well on a large ontology like SNOMED CT. We were able to classify SNOMED CT in approximately 50 minutes using a 5 node cluster with the distributed completion rule approach. These are just preliminary results and they can be improved in a variety of ways like making more nodes work on the slowest rule, improving the termination algorithm etc. After further evaluation and optimizations, we plan to publish our results (with complete details) along with rest of the approaches. Then, we would be moving on to Steps 2 and 3 mentioned in section 4.2.

Acknowledgements. This work was supported by the National Science Foundation under award 1017225 "III: Small: TROn - Tractable Reasoning with Ontologies." Any opinions, ndings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reect the views of the National Science Foundation.

[3] Internship work at Clark & Parsia LLC.

[4] http://www.cray.com/products/XMT.aspx

References

1. Aslani, M., Haarslev, V.: Concurrent classification of owl ontologies - an empirical evaluation. In: Proceedings of the 2012 International Workshop on Description Logics, DL 2012, Rome, Italy, June 7-10. CEUR Workshop Proceedings, vol. 846, CEUR-WS.org (2012)
2. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), Edinburgh, UK. Morgan-Kaufmann Publishers (2005)
3. Baader, F., Lutz, C., Suntisrivaraporn, B.: CEL — A Polynomial-Time Reasoner for Life Science Ontologies. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 287–291. Springer, Heidelberg (2006)
4. Baader, F., Lutz, C., Suntisrivaraporn, B.: Efficient reasoning in $\mathcal{EL}^+$. In: Proceedings of the 2006 International Workshop on Description Logics (DL 2006). CEUR Workshop Proceedings, vol. 189 (2006)
5. De Leon Battista, A., Dumontier, M.: A platform for reasoning with owl-el knowledge bases in a peer-to-peer environment. In: Proceedings of the 5th International Workshop on OWL: Experiences and Directions (OWLED 2009), Chantilly, VA, United States, October 23-24. CEUR Workshop Proceedings, vol. 529. CEUR-WS.org (2009)
6. Delaitre, V., Kazakov, Y.: Classifying elh ontologies in sql databases. In: Proceedings of the 5th International Workshop on OWL: Experiences and Directions (OWLED 2009), Chantilly, VA, United States, October 23-24 (2009)
7. Urbani, J., Kotoulas, S., Maassen, J., van Harmelen, F., Bal, H.: OWL Reasoning with WebPIE: Calculating the Closure of 100 Billion Triples. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part I. LNCS, vol. 6088, pp. 213–227. Springer, Heidelberg (2010)
8. Dentler, K., et al.: Comparison of reasoners for large ontologies in the owl 2 el profile. Semantic Web Journal 2(2), 71–87 (2011)
9. Kaoudi, Z., Miliaraki, I., Koubarakis, M.: RDFS Reasoning and Query Answering on Top of DHTs. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 499–516. Springer, Heidelberg (2008)
10. Kazakov, Y.: Consequence-driven reasoning for horn SHIQ ontologies. In: Proceedings of the 21st International Conference on Artificial Intelligence (IJCAI 2009), July 11-17, pp. 2040–2045 (2009)
11. Kazakov, Y., Krötzsch, M., Simančík, F.: Concurrent Classification of $\mathcal{EL}$ Ontologies. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 305–320. Springer, Heidelberg (2011)
12. Krötzsch, M., Maier, F., Krisnadhi, A., Hitzler, P.: A better uncle for owl: nominal schemas for integrating rules and ontologies. In: Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, pp. 645–654. ACM (2011)
13. Liebig, T., Müller, F.: Parallelizing Tableaux-Based Description Logic Reasoning. In: Meersman, R., Tari, Z. (eds.) OTM-WS 2007, Part II. LNCS, vol. 4806, pp. 1135–1144. Springer, Heidelberg (2007)
14. Liebig, T., Steigmiller, A., Noppens, O.: Scalability via parallelization of OWL reasoning. In: Proceedings of the 4th International Workshop on New Forms of Reasoning for the Semantic Web: Scalable and Dynamic (NeFoRS 2010) (2010)

15. Liu, C., Qi, G., Wang, H., Yu, Y.: Large Scale Fuzzy *pD** Reasoning Using MapReduce. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 405–420. Springer, Heidelberg (2011)
16. Mutharaju, R., Maier, F., Hitzler, P.: A mapreduce algorithm for el+. In: Proceedings of the 23rd International Workshop on Description Logics (DL 2010), Waterloo, Ontario, Canada, May 4-7 (2010)
17. Ongenae, F., De Backere, F., Steurbaut, K., Colpaert, K., Kerckhove, W., Decruyenaere, J., De Turck, F.: Appendix b: overview of the existing medical and natural language ontologies which can be used to support the translation process, http://www.biomedcentral.com/content/supplementary/1472-6947-10-3-s2.pdf
18. Oren, E., Kotoulas, S., Anadiotis, G., Siebes, R., ten Teije, A., van Harmelen, F.: Marvin: Distributed reasoning over large-scale Semantic Web data. Web Semantics: Science, Services and Agents on the World Wide Web 7(4), 305–316 (2009)
19. Ren, Y., Pan, J.Z., Lee, K.: Optimising parallel abox reasoning of el ontologies. In: Proceedings of the 2012 International Workshop on Description Logics, DL 2012, Rome, Italy, June 7-10. CEUR Workshop Proceedings, vol. 846. CEUR-WS.org (2012)
20. Schlicht, A., Stuckenschmidt, H.: Distributed resolution for alc. In: Proceedings of the 21st International Workshop on Description Logics (DL 2008), Dresden, Germany, May 13-16 (2008)
21. Schlicht, A., Stuckenschmidt, H.: Distributed Resolution for Expressive Ontology Networks. In: Polleres, A., Swift, T. (eds.) RR 2009. LNCS, vol. 5837, pp. 87–101. Springer, Heidelberg (2009)
22. Schlicht, A., Stuckenschmidt, H.: MapResolve. In: Rudolph, S., Gutierrez, C. (eds.) RR 2011. LNCS, vol. 6902, pp. 294–299. Springer, Heidelberg (2011)
23. Soma, R., Prasanna, V.K.: Parallel inferencing for OWL knowledge bases. In: 2008 International Conference on Parallel Processing, ICPP 2008, Portland, Oregon, USA, September 8-12 (2008)
24. Urbani, J., Kotoulas, S., Oren, E., van Harmelen, F.: Scalable Distributed Reasoning Using MapReduce. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 634–649. Springer, Heidelberg (2009)
25. Urbani, J., van Harmelen, F., Schlobach, S., Bal, H.: QueryPIE: Backward Reasoning for OWL Horst over Very Large Knowledge Bases. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 730–745. Springer, Heidelberg (2011)
26. van Gurp, M., et al.: Linkbase, a philosophically-inspired ontology for nlp/nlu applications. In: KR-MED 2006, Formal Biomedical Knowledge Representation, Proceedings of the Second International Workshop on Formal Biomedical Knowledge Representation, Baltimore, Maryland, USA, November 8. CEUR Workshop Proceedings (2006)
27. Weaver, J., Hendler, J.A.: Parallel Materialization of the Finite RDFS Closure for Hundreds of Millions of Triples. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 682–697. Springer, Heidelberg (2009)
28. Wu, K., Haarslev, V.: A parallel reasoner for the description logic alc. In: Proceedings of the 2012 International Workshop on Description Logics, DL 2012, Rome, Italy, June 7-10 (2012)

Replication for Linked Data

Laurens Rietveld*

Department of Computer Science,
VU University Amsterdam,
The Netherlands
laurens.rietveld@vu.nl

Abstract. With the Semantic Web scaling up, and more triple-stores with *update* facilities being available, the need for higher levels of simultaneous triple-stores with identical information becomes more and more urgent. However, where such Data Replication approaches are common in the database community, there is no comprehensive approach for data replication for the Semantic Web. In this research proposal, we will discuss the problem space and scenarios of data replication in the Semantic Web, and explain how we plan on dealing with this issue.

Keywords: synchronization, replication, decentralized triple-stores, read-write Web.

1 Problem Description

Up until recently, Semantic Web applications often made use of *read-only* triple-stores. These application are now taking up on using SPARQL-1.1 new 'update' facility, allowing users to *write* to triple-stores.

However, if triple-store contents change faster than they currently do, replication of Linked Data becomes a real problem, *i.e.* the challenge of keeping the information consistent between different data hubs. The role of these stores changes from that of a static content delivery system to a read and write content deposit. "Personal data lockers" [19] are an example of this scenario. They allow users to push information for it to be pulled by a variety of consumers. This high-frequency, dynamic information exchange between triple-stores requires efficient replication processes optimized for dealing with at least moderate volumes of data.

Although data replication is a very well-studied issue for databases and in file-synchronization for large-scale programming efforts, there is little work done with a particular focus on Semantic Web infrastructure. In this light, our hypothesis is:

> An efficient and comprehensive Linked Data replication approach requires more than the existing data replication techniques.

* This work was supported by the Dutch national program COMMIT.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 415–423, 2012.

2 Relevance

This section explains why Data Replication is relevant for Linked Data, by describing three use cases.

2.1 Triple Store Mirroring

Semantic Web applications (in our case Hubble[1], a Clinical Decision Support prototype using Linked Data) often rely on external triple-stores from the linked open data cloud. When one triple-store is slow or down, this has an effect on the responsiveness of the application. In scenarios where clinicians can walk around in the hospital with Hubble on their tablet, unreliable connections should not hinder application functionality. How can we ensure that most of the application still functions without internet connection? Linked Data Replication allows us to mirror an external triple-store either locally or to another server. This way we avoid direct dependencies on external triple stores: the application uses the mirrored triple-store, and the mirrored triple-store is a full replication of the master. In some cases, *partial* data replication is sufficient, e.g. when the application only relies on a subset of the data provided by the triple-store. Such applications where connectivity is unreliable is becoming more common in the Semantic Web domain, as more and more Semantic Web applications are appearing on hardware such as smart phones and tablets.

2.2 Annotations on Census Data

This use-case involves Dutch historical census data available as Linked Data. Annotation of original sources is one of the core activities of historical researchers. However, they are typically only interested in a subset of the dataset. One can consider a master triple-store containing all the data, and several subsets of the triple-store used by different researchers. Any annotation made on the subset, should be propagated to the master. Vice versa, any change made to the master, should be propagated to all the subsets. This scenario involves a combination of the problems of concurrent editing, dealing with version conflicts, and partial replication.

2.3 SemanticXO and Backups

Where the previous use-cases made use of a central 'master' triple-store, this is not always the case. The following use-case is an example of decentralized partial replication for Linked Data. The XO laptop is part of the One Laptop per Child (OLPC) project which aim is to create educational opportunities for the worlds poorest children by providing each child with a "rugged, low-cost, low-power, connected laptop". *Semantic*XO is a project which aims at providing an infrastructure to integrate the programs running on the XO into the Web

[1] https://github.com/Data2Semantics/Hubble

of Data. These programs can then publish and consume content to and from the WoD using the XO as the data provider [9]. The SemanticXO's all contain their own local triple-store. Due to unreliable internet connections, the laptops are not always connected to a central server or to the internet. Therefore communication between laptops in a mesh network has added value: exchanging information without the need to be connected to a network router. Currently, a-synchronous decentralized transfer of data is not possible. This makes tasks such as backup difficult. By using the SemanticXO triple-stores, the graphs in each triple-store can be replicated to other laptops. This way, the application data of each SemanticXO is backed-up on other XO laptops or XO servers. Because the SemanticXO's operate in environments with numerous constraints, the data replication functionality needs to adapt to these constraints [15] (e.g. by deciding which graphs in the triple-store to replicate, and in which order).

3 Related Work

3.1 Database Replication

Database replication is often used to improve performance and/or to improve availability [5]. The majority of database replication techniques are based on the state machine approach [18]. This approach ensures that replicated databases which share the same initial state and execute the same requests in the same order, will do the same thing and produce the same output. Inconsistent networks (i.e. unreliable networks, or networks with replicas which are not always online) often require a 2-phase or 3-phase commit protocol for every request, to maintain a consistent view. These protocols impose a substantial communication cost on each database transaction. Research into group communication protocols [2] reduces this overhead by avoiding the need to use these protocols on a *per action* basis while still maintaining a global persistent order. The state machine approach is usable for Linked Data, but requires server access; something which is not always the case when replicating remote triple stores.

Research into partial database replication [1,10,21] shows that full replication approaches are not directly applicable to partial replication scenarios. One of the problems in partial replication is that insert/update queries might rely on data which is missing on a partial replica. An approach to deal with this issue is described in [10], where the transaction logs are send to every replica, regardless of the replica holding a copy of the modified data. The replica only updates data items for which it holds a copy. If data items are referenced for which is does not hold a copy, the replica requests this information from the original server. As a downside, the replica might often receive transactions which it will not execute, thus creating unneeded overhead of network traffic. Additionally, because of the connectivity in graph structures, the approach of requesting missing information from the original server is not trivial task. In a Semantic Web scenario, an insert/select query executed on a partial replica has no way of knowing whether

an empty results from the *where* clause is caused by missing information on the partial replica, or whether this information should be absent anyway (i.e. is also missing on the original triple-store).

3.2 Ontology Differences

Related work on ontology differences is often inspired by the classical Version Control Systems. In [11], the causes, problems, and an approach for dealing with ontology changes are described, to achieve maintaining of interoperability while ontologies change. Here, the approach for dealing with different ontology versions is by comparing ontological classes, and displaying these side-by-side in RDF/XML.

[4] contains a description on how to formalize the differences betweevoidn graphs. Such differences can then be used for the updating and synchronization of graphs. A distinction is made between *weak* and *strong* patches, where a weak patch is only applicable to the same graph it was computed from, and a strong patch specifies the changes in a more context dependent manner. A weak patch is similar to the database replication methodology described above. A strong patch provides a way to deal with propagating these changes to partial replica.

Other related work on ontology differences is from [6,13] which focus on representing changes made to ontologies. Additionally, work from [8] resulted in an implementation (Protégé plugin) where the semantic differences between ontologies is calculated.

What most of these approaches have in common is the need to calculate the entailment and compare the complete graphs. For Linked Data replication this is often too heavy to perform, especially if close to any-time behaviour is desired. Research on incremental and stream reasoning ([3,7]) however show promising results on the time it takes to calculate the entailment.

Another common aspect of these approaches is their unsuitability for partial data replication, as in such a situation both graphs (the full master, and the partial slave) will always be different.

3.3 Linked Data Replication

One example of work on Linked Data Replication is RDFSync[22]. Here, full data replication between triple stores is achieved by decomposing the graphs into smaller *Minimum Self-Contained Graphs* (MSGs). By comparing the hashes of the MSGs of both triple store, the algorithm selects the MSGs it needs to transfer. This way, only the difference (including a certain amount of overhead) between triple stores is transferred. This approach however does not cover the complete problem space of Linked Data replication. RDFSync does not take into account partial data replication, and it requires installation on both servers; something which is not always possible.

An example of partial data replication is [16], where partial data replication in a master/master network is applied in the domain of mobile devices. Relatively heavy operations such as conflict resolution and merging, is done on the server

(i.e. triple store), which lead to low hardware requirements for the mobile devices. This approach requires a high level of server access on the triple store, something which is not always possible. This work is continued in [23], where the partial replication is made context-dependent (e.g. by user location or language).

Work on p2p Semantic Wiki's focusses mainly on concurrent editing and resolving conflicts in a full replication scenario. Some approaches (e.g. giki) use the GIT versioning system as underlying tool to deal with concurrent editing. Another approach is done by [20], where collaborative editing techniques from regular (non-Semantic-Web) p2p wikis (e.g. WOOT [12]) are extended to deal with the RDF model. These full data replication approaches deal with master/master networks, and all require a high level of server access to perform.

Finally, strongly related to Linked Data Replication is sparqlPuSH [14], which provides a mechanism to get notifications when the content of a triple-store is updated. Although this does require server access (installation) on the triple store server, it might be useful in the context of partial data replication, as the update mechanism supports notifications on subsections of the content.

4 Problem Space

4.1 Dimensions

We consider the problem space of data replication for RDF data to contain the following six dimensions: network structure, partiality, size, difftype, access level and time granularity.

Network Structure (Master-Master vs. Master-Slave): A network of master-master nodes contains nodes which can all perform updates. The changes each node makes are propagated to the other nodes. A master-master network introduces problems such as concurrent editing. How can such a network deal with conflicts when the same information is changed at the same time on two triple stores. The alternative to a master-master network is a network of master-slave nodes, where only the master has permission to update a graph, and the slaves have read-only rights. Data is then replicated from the master to the slave.

Partiality (Full vs partial data replication): Partial replication increases the data replication task considerably. How to detect changes related to the subset being replicated is one of the challenges, and how to define and support the views of data that should be replicated. Can we use rankings in the data to select the 'important' part of the graph to replicate, or use application/user profiles to detect what the information needs of the applications or users are.

Network Size : The more nodes there are in the network, the more urgent problems such as complexity and performance become. This is especially the case for master-master networks.

DiffType : Where the dimensions above are of a technical or infrastructural nature, and contain (almost) binary classes, this dimension is more targeted towards logics and is more scalular than binary. On one end of the scale we have the structural difftype, where the other values on the scale (increasing in complexity) makes use of semantics. The structural difftype essentially compares serializations of two triple stores. This does not account for the same knowledge represented differently in both stores.

Access level (No access vs. Black Box vs. White Box): The possible approaches for data replication depend on the abstraction level of the triple-stores. Some scenarios might require data replication to be implemented using a black box approach: the replication framework should work on all kinds of triple stores, and have no access to the lower level functionality of those stores. Other scenarios might require data replication where a lower level of functionality and triple-store access is required. This often results in different implementations for different triple-store vendors, as the architecture of the stores differ. Alternatively, there are situations where one might want to replicate a triple store without server access, and with only SPARQL access. This decreases the possible solutions considerably, as there is no way to install for instance a custom middle-layer (e.g. used to track changes to the triple store) on the server.

Time Granularity : How often does a triple-store change? Can changes occur any minute, or is it only updated once a year? The requirements and available solutions differ greatly between both.

4.2 Methods

There are three methods for Data Replication. These methods differ in applicability for each of the dimensions above.

1. Copying the complete graph. This is relatively inefficient, as often just a part of the graph changed.
2. Propagating the update queries, or bulks of update queries. This approach is difficult for partial data replication, as update queries on the full triple stores have a different context than the same queries on the partial triple store. This can result in different data being inserted in both triple-stores.
3. Propagating the actual difference between triple stores, either after a change has been made or at larger intervals (e.g. depending on the update frequency/time granularity of the triple-store). This requires knowing what has changed, and a formalization of this difference.

4.3 Replication for Linked Data vs. Database Replication

The replication scenarios of Linked Data and database replication differ greatly. Database replication scenarios often involve a closed network with large control over the different database servers. Linked Data however is an open network,

with one public query protocol standard, where there is often no control over external triple-stores.

These differences makes Linked Data replication partially a conceptually different problem than database replication. Applying the state-machine approach to Linked Data requires a certain level of server access, something which is often not feasible. Additionally (as explained in section 3.1), the graph structure and inference functionality of Linked Data presents issues in *partial* replication which are not covered by current database research.

5 Research Questions

The main question of this research is **How can we achieve Linked Data replication for all possible dimensions?**.

The different dimensions shown in section 4 present a large problem space with different questions for each of them. In this doctoral research, we chose to focus on the following questions: **Can we distinguish between general data replication scenarios for Linked Data, and how do these relate to the different dimensions?** This provides a specific set of requirements for the different replication scenarios, and a roadmap with which to guide this research.

How to decide which part of the data to replicate? For partial data replication, the selection of what to sync might not be obvious. Therefore, a selection of the graph needs to be made which needs to be replicated, for example using query logs, user profiles, or by rankings in the dataset.

How to efficiently use existing semantic diff algorithms in a data replication scenario? Existing research on semantic differences mostly have an analytical perspective, which might not fit the data replication requirements. Vice versa, in data replication scenarios we might make assumptions on datasets, which makes the semantic diff task easier. Something which is often not possible from an analytical point of view.

This introduces another question, namely: **How to calculate the semantic difference between a triple store and its partial replica?** It is not a trivial procedure to calculate the semantic difference between stores when on store is a subset of the other. After all, we are only interested in the difference of the *subset* of the original triple store, and the partial replica.

How to efficiently detect changes? This differs depending on the level of server access. No server access to a server means no middle-layer on the server to detect changes. What is the best way to do such change detection using for instance SPARQL?

What is the best 'unit of change'? E.g. synchronizing the update query, batches of update queries, the changed triples, a subset of the graph, or the complete graph. Which scenarios require what kind of change set?

6 Approach

For the actual synchronization of the changes between the triple store, there are several existing tools and platforms to use. In previous work we studied a

basic infrastructure for synchronization of basic RDF triples. We applied existing tools such as rsync, MySQL and GIT in the domain of the Semantic Web, and evaluated their performance using the standard SP^2 Semantic Web query testing environment. Besides these approaches there are other (e.g. Microsoft Sync Framework or the OpenSync) tools and platforms we can use for the actual distribution of data.

We will carry out our research in three phases. The first phase consists of making an overview of the general Linked Data replication scenarios, and their dimensions and requirements.

In the second phase, we will (starting with the scenario estimated as least challenging) use current database and Linked Data techniques to develop a method for data replication. If these techniques are insufficient in solving the problem of data replication, then the research will aim to develop techniques which do support data replication for this scenario.

In the third phase we will evaluate the Linked Data replication method created in phase 2. For experiment validity, all the servers are implemented using a virtual machine (VirtualBox) with the same hardware specifications. We will use a dataset generated by SP^2Bench[17], a data generator for creating arbitrarily large DBLP[2]-like datasets. We measure the performance by the bandwidth usage in the network of nodes, and the replication latency (i.e. the time it takes for both triple-store to be consistent).

7 Conclusion

We showed the importance and different scenarios of Linked Data replication. There is no other research with a comprehensive focus on data replication for Linked Data. However, as shown in section 3, there is related work on which we can build this research. We believe our previous work on synchronization infrastructures for Linked Data, and the related work, provides a solid base to build this research on.

References

1. Alonso, G.: Partial database replication and group communication primitives (Extended Abstract). In: Advances in Distributed Systems, pp. 1–6 (1997)
2. Amir, Y., Tutu, C.: From total order to database replication. In: Distributed Computing Systems, pp. 494–503. IEEE Comput. Soc. (2002)
3. Barbieri, D.F., Braga, D., Ceri, S., Della Valle, E., Grossniklaus, M.: Incremental Reasoning on Streams and Rich Background Knowledge. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part I. LNCS, vol. 6088, pp. 1–15. Springer, Heidelberg (2010)
4. Berners-lee, T., Connolly, D.: D delta: an ontology for the distribution of differences between rdf graphs. In: WWW (2004), http://www.w3.org/DesignIssues/Diff

[2] http://dblp.uni-trier.de/

5. Cecchet, E., Candea, G.: Middleware-based database replication: the gaps between theory and practice. In: ACM SIGMOD: Management of Data, pp. 739–752 (2008)
6. Franconi, E., Meyer, T.: Semantic diff as the basis for knowledge base versioning. In: Proc. of the 13th International Workshop on Non-Monotonic Reasoning (2010)
7. Goncalves, R., Parsia, B.: Analysing the evolution of the NCI Thesaurus. In: 24th IEEE International Symposium on Computer-Based Medical Systems (2011)
8. Groza, T.: Semantic Versioning Manager: Integrating SemVersion in Protégé. In: Proceedings of the 9th International Protege Conference, pp. 1–3 (2006)
9. Guéret, C., Schlobach, S.: SemanticXO: Connecting the XO with the World's Largest Information Network. In: Yonazi, J.J., Sedoyeka, E., Ariwa, E., El-Qawasmeh, E. (eds.) ICeND 2011. CCIS, vol. 171, pp. 261–275. Springer, Heidelberg (2011)
10. Holliday, J., et al.: Partial database replication using epidemic communication. In: International Conference on Distributed Computing Systems, pp. 485–493 (2002)
11. Klein, M.: Ontology versioning on the Semantic Web. Stanford University, pp. 75–91 (2001)
12. Oster, G., et al.: Data consistency for P2P collaborative editing. In: 20th Anniversary Conference on Computer Supported Cooperative Work, pp. 259–268 (2006)
13. Palma, R., et al.: Change Representation For OWL 2 Ontologies. In: 6th International Workshop on OWL: Experiences and Directions, pp. 1–10 (2009)
14. Passant, A.: sparqlPuSH: Proactive notification of data updates in RDF stores using PubSubHubbub. In: Scripting for the Semantic Web, pp. 1–10 (2010)
15. Rietveld, L., Schlobach, S.: Semantic Web in a Constrained Environment. In: Downscaling the Semantic Web Workshop, ESWC (2012)
16. Schandl, B.: Replication and Versioning of Partial RDF Graphs. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part I. LNCS, vol. 6088, pp. 31–45. Springer, Heidelberg (2010)
17. Schmidt, M., Hornung, T., Lausen, G., Pinkel, C.: SP2Bench: A SPARQL Performance Benchmark. In: Data Engineering 2009, pp. 222–233 (2009)
18. Schneider, F.B.: Implementing Fault-Tolerant Approach: A Tutorial Services Using the State Machine. ACM Computing Surveys (CSUR) 22(4), 299–319 (1990)
19. Siegel, D.: Pull: The Power of the Semantic Web to Transform Your Business. Portfolio (2009)
20. Skaf-Molli, H., Rahhal, C., Molli, P.: Peer-to-Peer Semantic Wikis. In: Bhowmick, S.S., Küng, J., Wagner, R. (eds.) DEXA 2009. LNCS, vol. 5690, pp. 196–213. Springer, Heidelberg (2009)
21. Sousa, A., Pedone, F.: Partial replication in the database state machine. In: International Symposium on Network Computing and Applications, pp. 298–309 (2001)
22. Tummarello, G., Morbidoni, C., Bachmann-Gmür, R., Erling, O.: RDFSync: Efficient Remote Synchronization of RDF Models. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 537–551. Springer, Heidelberg (2007)
23. Zander, S., Schandl, B.: Context-driven RDF data replication on mobile devices. In: 6th International Conference on Semantic Systems, vol. 3, pp. 131–155 (2011)

Scalable and Domain-Independent Entity Coreference: Establishing High Quality Data Linkages across Heterogeneous Data Sources*

Dezhao Song

Department of Computer Science and Engineering, Lehigh University
19 Memorial Drive West, Bethlehem, PA 18015, USA
des308@cse.lehigh.edu

Abstract. Due to the decentralized nature of the Semantic Web, the same real world entity may be described in various data sources and assigned syntactically distinct identifiers. In order to facilitate data utilization in the Semantic Web, without compromising the freedom of people to publish their data, one critical problem is to appropriately interlink such heterogeneous data. This interlinking process can also be referred to as *Entity Coreference*, i.e., finding which identifiers refer to the same real world entity. This proposal will investigate algorithms to solve this entity coreference problem in the Semantic Web in several aspects. The essence of entity coreference is to compute the similarity of instance pairs. Given the diversity of domains of existing datasets, it is important that an entity coreference algorithm be able to achieve good precision and recall across domains represented in various ways. Furthermore, in order to scale to large datasets, an algorithm should be able to intelligently select what information to utilize for comparison and determine whether to compare a pair of instances to reduce the overall complexity. Finally, appropriate evaluation strategies need to be chosen to verify the effectiveness of the algorithms.

Keywords: Entity Coreference, Linked Data, Domain-Independence, Scalability, Candidate Selection, Pruning.

1 Introduction, Challenges and Expected Contributions

Linked Data [3], which encourages the sharing of data and publishing of links to other datasets, has reached an impressive size: 295 datasets with about 31 billion triples and 500 million links across these datasets[1]. Since the same real world entity (e.g., people, locations, etc.) may be described by more than one data source with syntactically distinct identifiers, the biggest benefit of Linked Data is to enable people to walk from one dataset to others by following the linkages in order to obtain a relatively comprehensive view of the entities of interest.

To really facilitate the utilization of this large-scale and decentralized Linked Data, one critical problem is how to appropriately interlink such heterogeneous

* Advisor: Professor Jeff Heflin.

[1] http://www4.wiwiss.fu-berlin.de/lodcloud/state

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 424–432, 2012.

data with automated approaches. This interlinking problem has been well studied in Databases (*Record Linkage*) and Natural Language Processing (*Entity Coreference*) to find out which identifiers refer to the same real world entity. In this paper, we use the term *Entity Coreference* to refer to the process of finding ontology instances that describe the same real world entity in the Semantic Web.

Challenges. First of all, in order to detect coreferent instances precisely and comprehensively, it is important to locate and utilize the relevant information (the context) of the instances appropriately. Various situations can mislead the entity coreference results, such as name variations, the use of abbreviations, misspellings, etc. Also, the collected data may come from heterogeneous data sources and may be incomplete. To ensure the quality of the generated links, an entity coreference algorithm needs to address such challenges appropriately.

Making this context selection and utilization process domain-independent is equally important. A domain refers to the category (e.g., People, Geographic, etc.) and the usage (e.g., academic people, politics, etc.) of the data. In the past, domain-specific techniques have successfully helped to achieve good entity coreference results, e.g., relying on matching person names to identify coreferent person instances. However, when considering various domains, humans may lack the knowledge or time to specify what information to utilize and thus coreference tools are less likely to be available for all domains end users deal with.

Furthermore, scalability needs to be taken into account when designing entity coreference algorithms. Considering the scale of Linked Data, approaches that perform a brute-force comparison on every pair of instances [1, 16] are less likely to succeed. As a key part of this proposal, we will explore novel approaches to scaling entity coreference on large datasets: Candidate selection (*CS*) and context pruning (*CP*), i.e., doing fewer comparisons vs. doing faster comparisons. *CS* selects instance pairs that are likely to be coreferent in a lightweight manner and we only apply the expensive entity coreference algorithms on selected candidates. The key point of *CP* techniques is to compare an appropriately selected portion of the context to speed up the comparison for a single pair of instances.

Contributions. We propose to develop scalable and domain-independent algorithms for precisely and comprehensively detecting coreferent ontology instances from heterogeneous data sources with the following contributions:

- Developing mechanisms for automatically collecting and weighting context information of ontology instances in a domain-independent manner;
- Developing algorithms for detecting coreferent instances based upon the collected context, achieving precision and recall comparable to that of the state-of-the-art across various domains (e.g., >90% precision and recall);
- Devising techniques to link datasets without discriminative labels by exploring how to appropriately combine individually non-discriminating predicates;
- Developing effective pruning algorithms on the context of ontology instances in order to speed up the computation for a single pair of instances;
- Devising lightweight and domain-independent candidate selection algorithms targeting BTC-scale datasets (billions of triples and 400 million instances

[9]). Furthermore, the coreference results should not be affected much by applying such pruning techniques (e.g., 1-2% lower F1-score).

2 Related Work

Entity Coreference. The system by Aswani et al. [1] needs to interact with search engines to retrieve context information and thus may not scale to large datasets. RiMOM [23] and Silk [20] rely on human provided matching rules and thus costly to customize to new domains. RiMOM matches instances by comparing their property value pairs with Edit distance or the Vector Space model; to the best of our knowledge, it requires domain configuration to assign property weights. Silk is a general framework for users to specify rules for matching instances, but it may be difficult for users to specify such rules for all domains. Compared to these systems, we try to reduce the need of human input in developing entity coreference systems.

Hu et al. [7] build a kernel by adopting the formal semantics of the Semantic Web that is then extended iteratively in terms of discriminative property-value pairs in the descriptions of URIs. Algorithms that combine formal semantics of the Semantic Web and string matching techniques also include Zhishi.me [11], LN2R [15], CODI [12] and ASMOV [8]. These systems can be applied to datasets in different domains without human provided matching rules, such as People, Location, Organization and Restaurant. One disadvantage of reasoning based approaches is that they highly depend on the correct expressions of the ontologies. For example, as reported by the developers of the ASMOV system, in some dataset, the *surname* property was declared to be functional, yet if a person takes a spouses name, they will have different surnames for data collected at different times. According to our current experiments, our proposed algorithm is able to outperform several of these state-of-the-art systems on some benchmark datasets; however, further experiments are needed for a more comprehensive comparison on more diverse datasets.

Candidate Selection. Candidate selection selects instance pairs that are likely to be coreferent to reduce the overall complexity. ASN [26] relies on human input for identifying a candidate selection key; but sufficient domain expertise may not be available for various domains. Supervised [10] or partially-supervised [4] approaches have been explored to learn the candidate selection key; however, obtaining a sufficiently-sized groundtruth data is impractical for large datasets. Compared to these systems, our proposed candidate selection algorithm is unsupervised and is able to automatically learn the candidate selection key.

Indexing techniques have also been well-adopted for candidate selection [5]. PPJoin+ [25] adopts a positional filtering principle that exploits the ordering of tokens in a record. EdJoin [24] employs filtering methods that explore the locations and contents of mismatching n-grams. BiTrieJoin [21] is a trie-based method to support efficient edit similarity joins with sub-trie pruning. FastJoin [22] adopts fuzzy matching techniques that consider both token and character level similarity. Similar algorithms also include AllPairs [2] and IndexChunk

[14]. Although our proposed candidate selection algorithm also adopts indexing techniques, a secondary filtering on the looked-up candidates from the index significantly reduces the size of the final candidate set.

3 Research Accomplished

In this section, we present the core idea of each accomplished work.

Exhaustive Pairwise Entity Coreference based on Weighted Neighborhood Graph (EPWNG). *EPWNG* detects coreferent ontology instances by computing the similarity for every instance pair between datasets based on a set of paths (the context, Fig. 1) [16, 18]. $path = (x, P_1, N_1, ..., P_n, N_n)$, where x is an ontology instance; N_i and P_i are any expanded RDF node and predicate in the path. Each node N_i has a weight W_i computed based on the discriminability of its associated predicate P_i and the path weight is the multiplication of all its node weights. *EPWNG* compares the comparable paths in the context of two instances x and y. For each path m of x, we find the path n from y that is comparable and has the highest string similarity to m. We call the similarity between m and n the path score; the average path weight of m and n is treated as the weight of this score. This process is repeated for every path of x and the weighted average on such (path score, path weight) pairs is computed as the final similarity score for x and y. Here, two paths are comparable if their predicates at corresponding positions are comparable, i.e., having the same semantics. E.g., predicate *CiteSeer:name* is comparable to *DBLP:name*. Although the mapping axioms of predicate comparability were manually created in our experiments, they can also be automatically derived from ontology alignment systems [13].

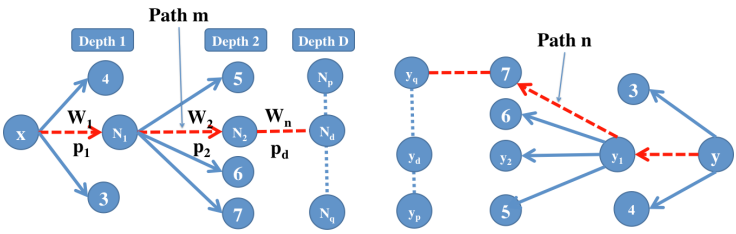


Fig. 1. Weighted Neighborhood Graph (G)

If we assume that multiple heterogeneous sources contain n instances in total, and that the context graphs have branching factor b and depth d, then the time complexity of *EPWNG* is $O(n^2 b^{2d})$, making it prohibitively expensive for dealing with large contexts and datasets.

Context Pruning. Given the complexity of *EPWNG*, one question is: *Can we speed up the comparison on a single instance pair by only considering the context that could potentially make a significant contribution to their final similarity score, i.e., reducing the impact of the branching factor b*? Therefore, we propose a sampling based context pruning technique [19]. Instead of actually computing

the string similarity between the last nodes of all pairs of comparable paths of two instances, we estimate how similar a node could be to another (the potential contribution) with a small sample from the entire dataset. When comparing two instances, before computing the end node similarity, we estimate if the potential contribution of the rest of the context would enable the similarity of two instances to go above a threshold. If so, we continue processing the remaining context; otherwise, we simply stop to save computational cost. Given that performing an estimation takes time itself, we further design a utility function to judge if it is worth estimating, which additionally provides 10% runtime savings.

Candidate Selection (CS). We further propose a candidate selection technique to reduce the impact of number of instances [17]. Ideally, a candidate selection algorithm should be able to automatically determine what information to utilize to select candidates, cover as many true matches as possible, and also select fewest pairs to help to scale the entire entity coreference process. Our proposed algorithm selects candidate instance pairs by computing a similarity on their character bigrams extracted from discriminating literal values that are chosen using domain-independent unsupervised learning. With unsupervised learning, we learn a set of datatype properties as the candidate selection key that both discriminates and covers the instances well. We then utilize the object values of such predicates for candidate selection. Instances are indexed on these object values to enable efficient look-up for similar instances. This algorithm has been shown to possess the properties discussed above on datasets in several domains (People, Publications, Hotel and Restaurant) with up to 1M instances.

4 Evaluation and Preliminary Results

Metrics. The standard metrics for evaluating entity coreference algorithms include: *Precision*: the number of correctly detected pairs divided by the total number of detected pairs; *Recall*: the number of correctly detected pairs divided by the number of coreferent pairs according to the groundtruth; and their *F1-score* calculated as $2*\frac{Precision*Recall}{Precision+Recall}$. Since it could be difficult to obtain perfect groundtruth for large datasets, sampled precision (sP) and relative recall ($relR$) could be adopted. $relR$ is calculated as $\frac{|correctly\ detected\ pairs\ from\ one\ system|}{|correctly\ detected\ pairs\ from\ all\ systems|}$; to measure sP, we can manually check the correctness of a subset of the detected links. The idea of *wisdom of the crowd* can be adopted for assessing precision while having perfect groundtruth to measure recall could still be difficult.

For candidate selection, Reduction Ratio (RR)=$1-\frac{|candidate\ set|}{N*M}$, Pairwise Completeness (PC)=$\frac{|true\ matches\ in\ candidate\ set|}{|true\ matches|}$, and their F1-score ($F_{cs}$) [10, 26] are three commonly used metrics. N and M are the size of two instance sets that are matched to one another. PC evaluates how many true positives are returned by an algorithm, RR is the degree to which it reduces the number of comparisons needed, and F_{cs} gives a comprehensive view of how well a system performs. Finally, runtime is an important metric for evaluating both types of systems.

Since the size of groundtruth and $N*M$ in RR may not be at the same order of magnitude, the calculated numbers of RR, PC and F_{cs} might not indicate

the actual differences of two systems appropriately. Particularly, when applied to large datasets, a large change in the size of the candidate set may only be reflected by a small change in RR due to its large denominator. Thus, in addition to evaluating candidate selection results with RR, PC and F_{cs}, we could apply an actual entity coreference algorithm to the selected candidates to measure the precision and recall of the final coreference results and the overall runtime.

Datasets. The Ontology Alignment Evaluation Initiative (OAEI) provides benchmark datasets for evaluating entity coreference systems. DBpedia, New York Times, Freebase, RKB and SWAT[2] are all suitable datasets as well. Finally, the entire LOD should be perfect for testing entity coreference algorithms.

Preliminary Results. In Table 1, *EPWNG* outperforms a few other coreference algorithms on three datasets (<2K instances) from OAEI2010 (left); compared to state-of-the-art candidate selection systems on 100K instances (right), *CS* enables the entire coreference process to run the fastest with the best coreference F1-scores. To further demonstrate and improve the domain-independence of *EPWNG* and *CS*, we will apply them to other diverse datasets from OAEI, including Location, Organization and Medicine.

Table 1. Evaluating Against State-of-the-Art Systems

Dataset	System	$P(\%)$	$R(\%)$	$F1(\%)$
Person1	EPWNG [18]	**100**	**100**	**100**
	RiMOM [23]	**100**	**100**	**100**
	ObjectCoref [7]	**100**	99.8	99.9
	LN2R [15]	**100**	**100**	**100**
	CODI [12]	87	96	91
Person2	EPWNG [18]	98.52	**99.75**	**99.13**
	RiMOM [23]	95.2	99	97.1
	ObjectCoref [7]	**100**	90	94.7
	LN2R [15]	99.4	88.25	93
	CODI [12]	83	22	36
Restaurant	EPWNG [18]	74.58	98.88	**85.02**
	RiMOM [23]	**86**	76.8	81.1
	LN2R [15]	75.67	75	75.3
	CODI [12]	71	72	72

Dataset	System	F_{cs}	$Coref\ F1\ (\%)$	$Time$ (s)
RKB Person	CS [17]	**99.68**	**93.63**	**12.25**
	AllPairs [2]	99.36	92.52	83.76
	PPJoin+ [25]	99.36	92.52	82.96
	EdJoin [24]	99.59	92.84	63.31
SWAT Person	CS [17]	99.32	94.90	**12.63**
	AllPairs [2]	99.52	**94.99**	108.34
	PPJoin+ [25]	99.52	**94.99**	106.72
	EdJoin [24]	**99.59**	94.94	102.77
RKB Pub	CS [17]	**99.99**	**99.74**	**15.05**
	AllPairs [2]	99.02	99.27	340.14
	PPJoin+ [25]	99.02	99.27	342.21
	EdJoin [24]	97.97	98.90	1330.20

5 Proposed Research

On-the-Fly Candidate Selection. Instead of pre-selecting candidate pairs, we are exploring candidate selection techniques at runtime. Consider that during the entity coreference process, an instance is compared to many other instances; the results of these prior comparisons could be useful in determining whether two instances might be coreferent. At any point in time, each instance should then have a *Matching History*, i.e., a set of other instances that it is somewhat similar to. One hypothesis is that two coreferent instances should share a sufficient amount of common instances in their histories. Therefore, the intuition of this on-the-fly candidate selection idea is that before actually computing the similarity

[2] http://swat.cse.lehigh.edu/resources/data

for an instance pair with expensive techniques, it might be worthwhile to spend a little effort to examine if their histories are similar enough for filtering purposes. Furthermore, as we process more instances, more true matches should be covered, thus it might make sense to gradually increase the threshold on such similarity of instances' matching histories to better balance F1-score and runtime.

Towards Linking the Entire Linked Open Data (LOD). With the goal of being able to handle the entire LOD, we will explore the following problems.

First, when handling the entire LOD, automated methods are needed to determine predicate comparability. As an alternative to complex ontology alignment systems, one idea is to determine predicate comparability based upon their value space, such that predicates with similar value spaces are comparable. Take the *fullname* predicate as an example. Rather than treating the names on their whole as values, tokens or n-grams can be extracted to form the value space.

Furthermore, given that LOD covers datasets from various domains, one might imagine how would the coreference results of one type of instances impact the others. For example, academic publications and researchers are generally correlated in academic datasets. Suppose we start from publications (since titles are generally very discriminating), could we then be able to achieve higher recall on matching person data by being able to provide better hints for person instance pairs with non-discriminative names (due to abbreviation, misspelling, etc.) but sharing coreferent publication instances (represented with syntactically distinct URIs) in their context? One step further, could we come up with approaches to automatically prioritize the domains to process, i.e., determining which domains should be processed first so that the other domains could benefit most? For scalability reasons, we could start with the existing linkages in the most influential domain instead of detecting everything from scratch. Since the existing links in LOD are of questionable quality [6], a lightweight verification step might be needed to firstly check the correctness of such links.

Last but not least, in prior work [16–19], the data we try to integrate generally contains some discriminative labels, e.g., names for people, hotel and restaurant and titles for publications. The question is what if we try to address domains that lack such discriminating labels? Maybe all predicates would then have relatively the same weight and thus *EPWNG* erroneously treats every piece of information the same? Or maybe all datatype properties will be selected for candidate selection and therefore no reduction will be achieved by having to deal with every single triple? One preliminary idea to handling non-discriminative data is to combine values from multiple properties, expecting the combined values could be more discriminating than that of any individual property.

References

1. 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.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 329–342. Springer, Heidelberg (2006)

2. Bayardo, R.J., Ma, Y., Srikant, R.: Scaling up all pairs similarity search. In: Proceedings of the 16th International Conference on World Wide Web (WWW), pp. 131–140 (2007)
3. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - the story so far. Int. J. Semantic Web Inf. Syst. 5(3), 1–22 (2009)
4. Cao, Y., Chen, Z., Zhu, J., Yue, P., Lin, C.Y., Yu, Y.: Leveraging unlabeled data to scale blocking for record linkage. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI), pp. 2211–2217 (2011)
5. Christen, P.: A survey of indexing techniques for scalable record linkage and deduplication. IEEE Transactions on Knowledge and Data Engineering, TKDE (2011)
6. Halpin, H., Hayes, P.J., McCusker, J.P., McGuinness, D.L., Thompson, H.S.: When owl:sameAs Isn't the Same: An Analysis of Identity in Linked Data. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 305–320. Springer, Heidelberg (2010)
7. Hu, W., Chen, J., Qu, Y.: A self-training approach for resolving object coreference on the semantic web. In: Proceedings of the 20th International Conference on World Wide Web (WWW), pp. 87–96 (2011)
8. Jean-Mary, Y.R., Shironoshita, E.P., Kabuka, M.R.: Ontology matching with semantic verification. Journal of Web Semantics 7(3), 235–251 (2009)
9. Khatchadourian, S., Consens, M.P.: Understanding billions of triples with usage summaries. In: Semantic Web Challenge (2011)
10. Michelson, M., Knoblock, C.A.: Creating relational data from unstructured and ungrammatical data sources. J. Artif. Intell. Res. 31, 543–590 (2008)
11. Niu, X., Sun, X., Wang, H., Rong, S., Qi, G., Yu, Y.: Zhishi.me - Weaving Chinese Linking Open Data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part II. LNCS, vol. 7032, pp. 205–220. Springer, Heidelberg (2011)
12. Noessner, J., Niepert, M., Meilicke, C., Stuckenschmidt, H.: Leveraging Terminological Structure for Object Reconciliation. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part II. LNCS, vol. 6089, pp. 334–348. Springer, Heidelberg (2010)
13. Pavel, S., Euzenat, J.: Ontology matching: State of the art and future challenges. IEEE Transactions on Knowledge and Data Engineering, TKDE (2011)
14. Qin, J., Wang, W., Lu, Y., Xiao, C., Lin, X.: Efficient exact edit similarity query processing with the asymmetric signature scheme. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 1033–1044 (2011)
15. Saïs, F., Pernelle, N., Rousset, M.-C.: Combining a Logical and a Numerical Method for Data Reconciliation. In: Spaccapietra, S. (ed.) Journal on Data Semantics XII. LNCS, vol. 5480, pp. 66–94. Springer, Heidelberg (2009)
16. Song, D., Heflin, J.: Domain-independent entity coreference in RDF graphs. In: Proceedings of the 19th ACM Conference on Information and Knowledge Management (CIKM), pp. 1821–1824 (2010)
17. Song, D., Heflin, J.: Automatically Generating Data Linkages Using a Domain-Independent Candidate Selection Approach. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 649–664. Springer, Heidelberg (2011)
18. Song, D., Heflin, J.: Domain-independent entity coreference for linking ontology instances. ACM Journal of Data and Information Quality, ACM JDIQ (2012)

19. Song, D., Heflin, J.: A pruning based approach for scalable entity coreference. In: Proceedings of the Twenty-Fourth International Florida Artificial Intelligence Research Society Conference (FLAIRS), pp. 98–103 (2012)
20. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and Maintaining Links on the Web of Data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 650–665. Springer, Heidelberg (2009)
21. Wang, J., Li, G., Feng, J.: Trie-join: Efficient trie-based string similarity joins with edit-distance constraints. PVLDB 3(1), 1219–1230 (2010)
22. Wang, J., Li, G., Feng, J.: Fast-join: An efficient method for fuzzy token matching based string similarity join. In: Proceedings of the 27th International Conference on Data Engineering (ICDE), pp. 458–469 (2011)
23. Wang, Z., Zhang, X., Hou, L., Zhao, Y., Li, J., Qi, Y., Tang, J.: RiMOM results for OAEI 2010. In: Proceedings of the 5th International Workshop on Ontology Matching (2010)
24. Xiao, C., Wang, W., Lin, X.: Ed-join: an efficient algorithm for similarity joins with edit distance constraints. Proc. VLDB Endow. 1(1), 933–944 (2008)
25. Xiao, C., Wang, W., Lin, X., Yu, J.X., Wang, G.: Efficient similarity joins for near-duplicate detection. ACM Trans. Database Syst. 36(3), 15 (2011)
26. Yan, S., Lee, D., Kan, M.Y., Giles, C.L.: Adaptive sorted neighborhood methods for efficient record linkage. In: ACM/IEEE Joint Conference on Digital Libraries (JCDL), pp. 185–194 (2007)

Distributed Reasoning on Semantic Data Streams*

Rehab Albeladi

School of Electronics and Computer Science, University of Southampton,
Southampton, SO17 1BJ, United Kingdom
{raab1g09,km,nmg}@ecs.soton.ac.uk

Abstract. Data streams are being continually generated in diverse application domains such as traffic monitoring, smart buildings, and so on. Stream Reasoning is the area that aims to combine reasoning techniques with data streams. In this paper, we present our approach to enable rule-based reasoning on semantic data streams using data flow networks in a distributed manner.

Keywords: Stream reasoning, Rete networks, XMPP.

1 Introduction

Developments in the area of the Internet and Web are constantly evolving. On one hand, the growing usage of sensors and embedded devices gives rise to a vision of the future Internet called "The Internet of Things" which aims to interconnect all these devices in a global network. By providing Internet connectivity to 'smart' embedded devices, computers will be able to automatically identify, monitor, react to, and perform actions on everyday objects. On the other hand, as the current Web is becoming the largest media of information, many researchers are working on "The Semantic Web", a vision of the future Web that aims to enable computers to understand the meanings of Web. Data in the Semantic Web has to be given well-defined meanings to be machine processable. A number of formats have been standardized such as RDF, RDFS, and OWL. The aim of these formats is to structure and give semantics to the Web data, which will then enable automatic reasoning and processing of this data.

Despite the fact that the Internet of Things focuses on the infrastructure issues and the Semantic Web focuses more on knowledge representation - as they basically work in different layers - the two visions aim to interlink the virtual and physical worlds. The Internet of Things identifies real world objects using unique RFID tags, while the Semantic Web uses URIs to uniquely identify real world objects. However, both visions can complement each other.

While event processing engines or Data Stream Management Systems [1][2] can be used to process data streams generated by the IoT devices, the heterogeneity of the data sources and formats makes interoperability a real challenge. Furthermore, stream processing engines cannot perform complex reasoning tasks due to the lack of

* Advisors: Kirk Martinez and Nicholas Gibbins.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 433–436, 2012.

semantics [3]. Adopting Semantic Web formats provides standardization and enables automatic reasoning over the IoT streaming data.

On the other hand, while semantic reasoners work efficiently on typical static knowledge, the challenge of reasoning upon rapidly changing information and data streams has received far less attention than reasoning upon static data. The combination of reasoning techniques with data streams gives rise to "Stream Reasoning", which is a little explored, but high impact research area [4]. This area aims to provide the abstractions, foundations, methods, and tools required to integrate data streams and reasoning systems.

In this research, we aim to approach the following research questions:

- How to enable rule-based reasoning over RDF data streams as data flow networks efficiently (using minimal resources) and effectively (providing timely results with high precision and recall)?
- How to distribute the reasoning network in order to maintain the scalability and improve the efficiency (e.g. by pushing filters near to data sources)?

We also aim to provide a proof-of-concept implementation, which will be evaluated for reasoning on real-time RDF streams. We expect the system to perform faster than a traditional triple store, as there is no indexing. The tradeoff between completeness of the results and processing time is expected to be controllable by varying window sizes.

2 Related Work

In the Semantic Web, data is represented in RDF, and queries can be performed using SPARQL. However, to express streaming data, RDF needs to be extended to represent time, which is an important concept in data streams. SPARQL also cannot support queries on streaming data as it lacks crucial operators found in the data stream management systems, such as the window operators. Research in the stream reasoning area mainly focuses on extending SPARQL to process RDF streams. The first attempt to extend SPARQL was presented by Bolles et al. [5]. They introduced Streaming SPARQL as a SPARQL extension to cope with window queries over RDF streams. Continuous SPARQL [6] is a SPARQL extension that follows a CQL-like [2] approach. EP-SPARQL [7] is another SPARQL extension proposed as a new language for event processing and stream reasoning.

We focus more on the infrastructure of the reasoning process. Using Rete networks [8], our approach enables reasoning in a continuous manner using low-level operators that work directly on RDF streams. Rete networks are also used in [9] to enable schema-enhanced pattern detection on RDF data streams. However, they present a fixed approach that can only operate over RDF Schema, while we aim to provide generic rule-based reasoning and query answering.

We also aim to provide a scalable distributed reasoning. Hoeksema and Kotoulas [10] present a parallel approach for stream reasoning using Yahoo S4 framework. They introduce a number of RDFS specialized reasoning Processing Elements to distribute triples over multiple streams. Continuous query answering is also supported by a number of components that can be combined to translate a subset of C-SPARQL into a parallel execution plan.

3 Proposed Approach

Continuous Reasoning: To enable the rule-based reasoning process, we use the Rete algorithm [8], in which reasoning is implemented natively over streams as data flow networks. The Rete algorithm – originally designed to solve the many pattern/ many objects problem - can process large data sets efficiently because it avoids iterating over both data elements and production rules. To avoid iteration over data elements, the Rete algorithm stores with each condition (or pattern), a list of the data elements that it matches. These lists are updated when the working memory changes. To avoid iteration over rules, rules are translated into Rete networks of nodes. The nodes represent different operators that can be shared between rules and the data flows between these nodes. The tree-like network divides the matching process into multiple steps that perform different checks, so if a data element does not match the first node, it is simply discarded and does not complete its way through the network.

A prototype RDFS reasoner for RDF data streams has been fully implemented, combining features from both reasoning techniques and stream processing techniques; it performs the inference task as a rule engine using a Rete network, while the implemented Rete network performs some DSMS operations, such as converting streams into relations by using the sliding window technique. The system is fed by RDF streams, which are matched against the RDFS entailment rules, so producing new sets of data in a continuous manner. In our initial evaluation, we have been able to demonstrate the tradeoff between completeness and execution time by varying window sizes.

Distribution: For efficient processing of large volume data, scalability is a major concern. Distributed processing of data streams enables more scalable systems as they can scale in two dimensions: the hardware performance of each computing node and the number of nodes [11]. A distributed stream reasoner should be also more fault-tolerant by avoiding a single point of failure and enabling the migration of operators between the affected nodes.

We propose distributing the Rete network operators across multiple machines and use the eXtensible Messaging and Presence Protocol (XMPP) [12] for nodes' communication. We have chosen XMPP for its push-based distribution style, which satisfies the real-time requirement of streaming applications with minimal latency, and for its ease of integration with Web technologies.

We have built a prototype system that can process RDF data streams using distributed Rete networks, in which nodes are distributed in multiple machines and can communicate with each other using XMPP in a publish/subscribe pattern, and are now working on combining this system with our previous reasoner to perform continuous reasoning on streaming RDF data in a distributed manner.

4 Conclusions and Future Work

We have proposed a stream reasoning framework using distributed Rete networks. Our prototype system supports reasoning over RDFS entailment rules using a

pre-designed Rete network. The prototype can be similarly extended to support OWL 2 RL reasoning. However, to enable generic query answering, we need to define an algorithm that can dynamically translate queries into rules and then to Rete networks.

On the distribution side, we only investigated the communication aspect of the challenge. Another issue to be addressed is the load balancing. An efficient method to dynamically adjust the allocation of processing among the available nodes is required.

Finally, a set of experiments to evaluate the system needs to be planned. RDF streams with different arrival rates will be fed into the system, and the results will be evaluated using precision and recall metrics to determine the effectiveness of the system, while measuring processing time and memory consumption.

Acknowledgments. This research is funded by Taibah University, Medina, Saudi Arabia.

References

1. Abadi, D., Carney, D., Çetintemel, U., Cherniack, M., Convey, C., Lee, S., Stonebraker, M., Tatbul, N., Zdonik, S.: Aurora: A New Model and Architecture for Data Stream Management. The VLDB Journal 12(2), 120–139 (2003)
2. Arasu, A., Babcock, B., Babu, S., Datar, M., Ito, K., Motwani, R., Nishizawa, I., Srivastava, U., Thomas, D., Varma, R., Widom, J.: STREAM: The Stanford Stream Data Manager. IEEE Data Engineering Bulletin 26 (2003)
3. Della Valle, E., Ceri, S., van Harmelen, F., Fensel, D.: It's a Streaming World! Reasoning upon Rapidly Changing Information. IEEE Intelligent Systems 24(6), 83–89 (2009)
4. Della Valle, E., Ceri, S., Barbieri, D.F., Braga, D., Campi, A.: A First Step Towards Stream Reasoning. In: Domingue, J., Fensel, D., Traverso, P. (eds.) FIS 2008. LNCS, vol. 5468, pp. 72–81. Springer, Heidelberg (2009)
5. Bolles, A., Grawunder, M., Jacobi, J.: Streaming SPARQL - Extending SPARQL to Process Data Streams. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 448–462. Springer, Heidelberg (2008)
6. Barbieri, D., Braga, D., Ceri, S., Grossniklaus, M.: An execution environment for C-SPARQL queries. In: 13th International Conference on Extending Database Technology. ACM, Lausanne (2010)
7. Anicic, D., Fodor, P., Rudolph, S., Stojanovic, N.: EP-SPARQL: A unified language for event processing and stream reasoning. In: Proceedings of the 20th International Conference on World Wide Web. ACM, Hyderabad (2011)
8. Forgy, C.: Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. Artificial Intelligence 19, 17–37 (1982)
9. Komazec, S., Cerri, D.: Towards efficient schema-enhanced pattern matching over RDF data streams. In: Proceedings of the First International Workshop on Ordering and Reasoning (2011)
10. Hoeksema, J., Kotoulas, S.: High-performance distributed stream reasoning using S4. In: Proceedings of the First International Workshop on Ordering and Reasoning (2011)
11. Urbani, J., Kotoulas, S., Oren, E., van Harmelen, F.: Scalable Distributed Reasoning Using MapReduce. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 634–649. Springer, Heidelberg (2009)
12. Saint-Andre, P.: Extensible messaging and presence protocol (xmpp): Core, RFC. The Internet Society (2004)

Reusing XML Schemas' Information as a Foundation for Designing Domain Ontologies

Thomas Bosch

GESIS - Leibniz Institute for the Social Sciences, Mannheim, Germany
thomas.bosch@gesis.org

Abstract. Designing domain ontologies from scratch is a time-consuming endeavor requiring a lot of close collaboration with domain experts. However, domain descriptions such as XML Schemas are often available in early stages of the ontology development process. For my dissertation, I propose a method to convert XML Schemas to OWL ontologies in an automatic way. The approach addresses the transformation of any XML Schema documents by using the XML Schema metamodel, which is completely represented by the XML Schema Metamodel Ontology. Automatically, all Schema declarations and definitions are converted to class axioms, which are intended to be enriched with additional domain-specific semantic information in form of domain ontologies.

Keywords: Semantic Web, Ontology Design, OWL, XML, XSD, XSLT.

1 Problem and Research Question

XML represents a large set of information within the context of various domains and has reached wide acceptance as standard data exchange format. Traditionally, ontology engineers work in close collaboration with domain experts to design domain ontologies manually, which requires a lot of time and manpower. Domain ontologies as well as XSDs describe domain data models. In many cases, XSDs are already existent and can therefore be reused in the process designing domain ontologies from scratch. As a consequence, saved time and effort could be used more effectively to enrich data models with supplementary domain-specific semantic information, not or not satisfyingly covered by the underlying XSDs. The main research question, how the time-consuming process designing domain ontologies based on already available XSDs could be accelerated, results from the stated problem.

2 Proposed Approach

Concept. Bosch and Mathiak have developed the concept of the generic multilevel approach to design domain ontologies based on already available XSDs [1]. XSDs determine the terminology, the vocabulary and the syntactic structure of XML document instances. XSDs are instances of the XSD metamodel. The components of the

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 437–440, 2012.

XSD abstract data model, also called element information items (EIIs) in the XML representation, are mapped to classes, universal restrictions on datatype and object properties of a generic ontology, the XML Schema Metamodel Ontology (XSDMO). The idea of the developed approach is to convert any XSDs automatically to classes, hasValue restrictions on XSDMO's datatype properties and universal restrictions on XSDMO's object properties using XSLT (Bosch and Mathiak explain implementation details in [2]). Any XSD can be transformed into a generated ontology, since each component of the XSD abstract data model is covered by this approach. On the instance level, XML document instances are translated into an RDF representation of the generated ontologies by means of a Java program as XSLT is less powerful for this purpose. After these two transformation processes, which take only seconds, all the information located in the XSDs is re-used in generated ontologies and their RDF representations can now be published in the LOD cloud and be linked to other RDF datasets. Generated ontologies are not directly as useful as manually created domain ontologies, as XSD and OWL follow different modeling goals, since generated ontologies' structures are rather complex, and as generated ontologies are not conform to the highest quality requirements of domain ontologies. Therefore, the generated ontologies' class axioms are intended to be further supplemented with additional domain-specific semantic information, not specified in underlying XSDs, in form of domain ontologies. These domain ontologies can be derived automatically out of the generated ontologies using SWLR rules on the schema as well as on the instance level. Consequentially, all XML data conforming to XSDs can be imported automatically as domain ontologies' instances. The effort and the time, however, delivering high quality domain ontologies subsequently is much less than creating domain ontologies completely manual and could be used more effectively to expand the XSDs' domain knowledge.

Related Work. The XSDMO corresponds to the general database ontology designed by Kupfer et al. [3]. They have defined a schema-to-ontology mapping: database ontologies are generated automatically from database schemas. Semantic domain-specific information is added supplementary to database ontologies in form of domain ontologies.

Motivation and Use Case. Bosch et al. delineate the DDI ontology [4], whose derivation serves as complete, intuitive, and representative use case to motivate the approach's application. The Data Documentation Initiative (DDI) is an acknowledged international standard for the documentation and management of data from the social, behavioral, and economic sciences. Excerpts of the DDI ontology are derived out of the underlying XSDs describing the statistics domain. DDI XML documents include XML elements 'Question' containing 'QuestionText' elements, which may comprise plain text such as 'How old are you?'. The element 'Question' is an instance of the XSD EII 'element' whose 'name' and 'type' attributes have the values 'Question' and 'QuestionType'. The complex type 'QuestionType' includes the EII 'complexContent' containing the EII 'extension' which comprises a sequence. This sequence contains a reference to the global element 'QuestionText', the type of the XML element 'QuestionText'. 'QuestionText' includes the text 'How old are you?' which if of the XSD's primitive datatype string.

XSD's EIIs are converted to generated ontology's classes which are defined as sub-classes of XSDMO' super-classes: <EII> ⊑ <meta-EII>. The global element 'QuestionText' (<xs:element name="QuestionText"/>), for example, is translated into the class 'QuestionText-Element…' which is specified as sub-class of the super-class 'Element' (<QuestionText-Element…> ⊑ <Element>), as each particular EII 'element' is also part of the 'Element' class extension. EIIs' attributes' values are converted to XSDMO's datatype properties '<attribute>_<domain meta-EII>_String' and to hasValue restrictions on these datatype properties: <domain EII> ⊑ ∃ <attribute>_<domain meta-EII>_String.{<String>}. The value 'Question' of the 'element' EII's attribute 'name' (<xs:element name="Question"/>) is translated into the datatype property 'name_Element_String' and into the datatype property's universal restriction Question-Element… ⊑ ∃ name_Element_String.{'Question'}, since 'Question-Element…' resources must have at least one relationship along the datatype property 'name_Element_String' to the string individual 'Question'. EIIs' attributes' values referring to other EIIs are transformed into XSDMO's object properties' universal restrictions <domain EII> ⊑ ∀ <ref|substitutionGroup|refer>_<domain meta-EII>_<range meta-EII>.<range EII>. The value 'QuestionText' of the 'element' EII's attribute 'ref' (<xs:element ref="QuestionText"/>) referring to the EII 'element' with the name 'QuestionText' is translated into the object property's universal restriction QuestionText-Element-Reference… ⊑ ∀ ref_Element_Element.Question Text-Element…, as 'QuestionText-Element-Reference…' instances can only have 'ref_Element_Element' relationships to 'QuestionText-Element…' resources or have no such relations. Values of EIIs' attributes referring to type definitions are translated into universal restrictions on XSDMO's object properties <domain EII> ⊑ ∀ type|base_<domain meta-EII>_Type.<range EII>. The value 'QuestionType' of the attribute 'type' of the 'Question' EII 'element' (<xs:element name="Question" type = "QuestionType"/>) is converted to the object property's universal restriction Question-Element… ⊑ ∀ type_Element_Type.QuestionType-Type…. The part-of relationship of the EII 'sequence' (<sequence><element ref="QuestionText"/></sequence>) is translated into the object property's universal restriction Sequence… ⊑ ∀ contains_Sequence_Element.QuestionText-Element-Reference…. The strict order of the in the sequence contained EIIs is expressed by the object property's universal restriction Sequence… ⊑ ∀ sequence.QuestionText-Element-Reference…. As resources of the class 'QuestionText-Element…' may have text as content, the datatype property 'value_Element_String' is introduced and the datatype property's universal restriction QuestionText-Element… ⊑ ∀ value_Element_String.String is defined.

We want to derive that the 'Question-Element…' resource 'age' is also of the type 'Question' with the question text 'How old are you?'. The following program fragment demonstrates the antecedent and the consequent of the SWRL rule, executed by a rule engine to derive the two statements: `(?a type_Element_Type ?b) ∧ (?b contains_ComplexType_ComplexContent ?c) ∧ … ∧ (?g rdf:type QuestionText-Element…) ∧ (?g value_Element_String ?h) -> (?a rdf:type Question) ∧ (?a questionText ?h)` The two statements can be derived since the individual 'age', substituting the SWRL variable '?a', has a relationship along 'type_Element_Type' to

an individual replacing the variable ‘b’. This resource is linked to an instance ‘?c’ via ‘contains_ComplexType_ComplexContent’. Further, there’s a navigation path from the ‘?c’ individual to the ‘?g’ instance along the stated properties. As XML elements ‘QuestionText’ may contain text nodes like ‘How old are you?’, the ‘?g’ instance is assigned to the class ‘QuestionText-Element…’ ensuring that derived question texts are only strings contained in ‘QuestionText-Element…’ resources. The ‘?g’ resource must have a ‘value_Element_String’ relation to a ‘?h’ individual. As the instances ‘age’ and ‘How old are you?’ correspond to the SWRL rule’s antecedent, it can be inferred that the resource ‘age’ is a question with the question text ‘How old are you?’.

3 Results and Future Work

The approach’s concept has been finalized and the mapping of the XSD metamodel to the XSDMO has been defined and implemented. The mapping between XSDs and generated ontologies has been specified and programmatically realized. Also the generality of the approach has been verified, since the generic test cases have shown that all meta-EIIs of the XSD metamodel are covered and thus each XSD can be transformed into a generated ontology using the same transformation rules.

Currently, I’m writing a Java program translating XML documents into RDF representations of the generated ontologies. So far, the most relevant subsets of the DDI domain ontology are derived and appropriate SWRL rules are defined. To verify the hypothesis that the effort and the time delivering high quality domain ontologies using the developed approach is much less than creating domain ontologies manually, the traditional manual and the proposed semi-automatic approach will be compared by means of a user study. For an extensive evaluation of the work, it is absolutely essential to create generated ontologies and to deduce domain ontologies out of XSDs of multiple and differing domains.

References

1. Bosch, T., Mathiak, B.: Generic Multilevel Approach Designing Domain Ontologies Based on XSDs. In: Proceedings of the Workshop Ontologies Come of Age in the Semantic Web, 10th International Semantic Web Conference, pp. 1-12. CEUR Workshop Proceedings, Aachen (2011)
2. Bosch, T., Mathiak, B.: XSLT Transformation Generating OWL Ontologies Automatically Based on XSDs. In: IEEE Xplore Digital Library, 6th International Conference for Internet Technology and Secured Transactions, pp. 660–667. IEEE Xplore Digital Library (2012)
3. Kupfer, A., Eckstein, S., Störmann, B., Neumann, K., Mathiak, B.: Methods for a Synchronised Evolution of Databases and Associated Ontologies. In: Proceeding of the 2007 Conference on Databases and Information Systems IV (2007)
4. Bosch, T., Cyganiak, R., Wackerow, J., Zapilko, B.: Leveraging the DDI Model for Linked Statistical Data in the Social, Behavioural, and Economic Sciences. In: Proceedings of the International Conference on Dublin Core and Metadata Applications, pp. 46–55 (2012)

A Multi-domain Framework for Community Building Based on Data Tagging

Bojan Božić

Austrian Institute of Technology,
Donau-City-Straße 1,
1220 Vienna,
Austria

Abstract. In this paper, we present a doctoral thesis which introduces a new approach of time series enrichment with semantics. The paper shows the problem of assigning time series data to the right party of interest and why this problem could not be solved so far. We demonstrate a new way of processing semantic time series and the consequential ability of addressing users. The combination of time series processing and Semantic Web technologies leads us to a new powerful method of data processing and data generation, which offers completely new opportunities to the expert user.

1 Introduction

Nowadays time series processing is not only a very complex research field, but also a very specialized one. There are a lot of parties interested in time series data and all of them have certain "tailor-made" solutions for their specific problems. Therefore, we have developed the Time Series Semantic Language (TSSL). TSSL evolves from a conservative, general-purpose time series processing language, to a processing language for semantically enriched time series.

Our idea is to use semantically enriched time series to improve data processing in the Semantic Web, i.e. to be able to annotate data flows as sensor data with additional information, e.g. tagging postings of scientist with a specific research topic, which can be seen as time series, or the other way around, to use time series data as input for the creation of ontologies.

The first early prototype of the language has been developed at the Austrian Institute of Technology as scripts for processing environmental time series data. We extended the language implementation from a very specific usage to a general and dynamic language for many fields of application. To empower this, we added semantic functionality and implemented first prototypes.

To demonstrate our work and ideas, this paper describes the use of this dedicated language, which enables time series generation and processing enriched with semantics.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 441–444, 2012.

2 Language Specification

The language on which this research is based, is originally a classical time series processing language. This means that it is a generic language for processing time series data.

The language supports homogeneous (with fixed time grids) and inhomogeneous time series processing. Time series can have very complex data structures. It is also possible to work with time patterns, time intervals, and single slots. The complex types of aggregation can be performed with predefined, but also with user-defined functions.

Table 1. General expressions and their meanings

Expression	Meaning
`< [n].sin * 2 + 3 >`	Calculation is applied to all slots.
`A, B < A + 2 * B >`	Combination of two time series (aggregation).
`< [n] > every 2 hours`	Projection to a fixed time grid.
`< (t .. t-2).mean > every 1 hour`	Sliding mean value.
`< [n]->hot if [n].temperature > 100 otherwise [n]->cold >`	Filtering, classification.

Some common expressions are shown in table 1. The first expression calculates the sine of the value from each slot, multiplies it by 2 and adds the value of 3. Expression 2 specifies two time series, where each slot of time series A is added to the doubled value of each slot from time series B. The usage of a time grid is shown in expression 3, where only the slot default value is taken every 2 hours and copied to the output time series. Expression 4 calculates a mean value for each slot and the previous two slots, but only every 1 hour. Finally, the last expression takes the property "hot" if the temperature is higher then 100, and "cold" otherwise.

TSSL has been implemented in the Python programming language, to guarantee the ease of extensibility and interoperability with other programming languages. Therefore it is usable as a standalone library on major platforms[1].

3 Semantic Time Series Processing

The main innovative contribution of our work is semantic time series processing. It tries to fix the weaknesses of current time series processing systems, such as:

[1] Currently Java, .Net, and native.

- meta-information is often non-existent or not bound to the processing of data,
- the linkage of ontologies is missing and therefore connections of information cannot be respected automatically,
- no possibility to add domain-specific ontologies at runtime, hence domain-specific processing is hard to implement.

The semantically enriched time series processing language, introduced in this paper is able to use predefined or user-provided ontologies to assign meaning to information. It supports automatic consideration of domain-specific calculations and functions, such as mean value calculation, thresholds, etc. for certain domains. This means that it depends on the domain how certain processing steps are affected. The advantage is the lower fault probability, because complex expressions are easier to phrase. Another issue is the verification of reasonability, e.g. there needs to be a difference between the water temperature, room temperature, and outdoor temperature. For all mentioned functionality extensions there is no need to change the syntax of the language.

This principle becomes even more obvious, if we take a look at exemplary expressions for both processors. The following expressions are examples of filtering a meteorologic time series. The name of the time series is *MeteoTS*. The first expression defines that a warning should be returned if the precipitation is greater than 1000 $\mathrm{l/m^2}$ or the temperature is greater than 40 ^{o}C or the wind speed is greater than 56 knots, etc. If no semantics is supported, one needs to specify every single condition and every single kind of warning in the expression.

```
  MeteoTS < warning if precipitation > 1000 l/m²
or temperature > 40°C or wind > 56 knots ... >
```

The second expression has the same meaning as the first one. The only difference is that it is written for a time series processor that supports semantic time series processing. The name of the time series is again *MeteoTS*. Again a warning is returned if the value exceeds the allowed limit. The difference is that the value and the limit are not specified exactly, they rather depend on the used ontology. This means that we have different values and different limits depending on the targeted domain. The same expression can thus be used for a number of different processings and many different target groups.

```
MeteoTS < warning if value > allowed >
```

The example above shows only one possible use case for semantic extensions. It does not mean that the only improvement of semantics in time series processing is the flexible formulation of thresholds. There are many other use cases, like consideration of special information in different domains (e.g. data of a meteorologic time series may be interesting for many different domains like government, event management, air traffic, agriculture, tourism, etc., but every domain is interested in a different view of the data). Thus, semantics can help us to provide the right information to the right interest group.

4 Related Work

Semantic Web technologies have undergone a huge development in the last couple of years. New tools, technologies and projects are being introduced almost on a daily basis and first steps have been undertaken to combine the concepts of Semantic Web and Web 2.0 [1].

In ontology-based knowledge management, the SEKT[2] project produced very interesting results. Current issues in social ontologies [3], and a discussion on the relation between sociability and semantics [2] are important and could be of high interest.

The state-of-the-art in Semantic Web and Web Mining is developing very fast, and these two research areas are more and more combined, as results of Web Mining are improved by exploiting semantic structures in the Web, and Web Mining techniques are used for building the Semantic Web [4].

5 Conclusion

Our time series processing language for semantically enriched time series is an attempt to assign the right time series to the right person. The language itself is first of all a time series processing language, which covers classical time series processing functionality like arithmetic calculations, time patterns, slot selection, aggregation, mean value calculation, and much more.

Semantic time series processing is one of the features that distinguishes our language from others. It enables the consideration of meta-information, the integration of ontologies and the possibility to add domain-specific ontologies at run-time.

The time series processing language is already used in different domains like environment, traffic, etc., and several prototypes for the Semantic Web like components for accessing RDF stores, visualization and filtering, user context management.

References

1. Ankolekar, A., Krötzsch, M., Tran, T., Vrandečić, D.: The two cultures: Mashing up web 2.0 and the semantic web. Web Semantics: Science, Services and Agents on the World Wide Web (6), 70–75 (November 2007)
2. Mika, P.: Social networks and the semantic web: the next challange. IEEE Intell. Syst. 1(20), 82–85 (2005)
3. Mika, P., Gangemi, A.: Descriptions of social relations. In: Proceedings of the First Workshop on Friend of a Friend. Social Networking and the (Semantic) Web (2004)
4. Stumme, G., Hotho, A., Berendt, B.: Semantic web mining state of the art and future directions. Web Semantics: Science, Services and Agents on the World Wide Web (4), 124–143 (February 2006)

[2] http://www.sekt-project.com

Towards a Theoretical Foundation for the Harmonization of Linked Data

Enrico Daga*

Knowledge Media Institute, The Open University, United Kingdom
Semantic Technology Laboratory, ISTC, Consiglio Nazionale delle Ricerche, Italy

Abstract. In real world cases, building *reliable problem centric views* over Linked Data [1] is a challenging task. An ideal method should include a formal representation of the requirements of the needed dataset and a controlled process moving from the original sources to the outcome. We believe that a goal oriented approach, similar to the AI planning problem, could be successful in controlling the process of linked data fusion, as well as to formalize the relations between requirements, process and result.

Keywords: Linked Data, Data Harmonization, Planning.

1 Introduction

We intend *Linked Data Harmonization* to be a controlled data preparation process, which transforms, aggregates, filters, fix and clean information from various linked data sources into a new *harmonized dataset* to fulfill the needs of a specific problem space, expressed as a formalized data schema. There is no single command that can achieve this. In contrast several actions must be performed in order to reach the goal (SPARQL, Linking, Programming, Rules, Reasoning, etc.). This task strongly resembles the AI planning problem. A planner takes a goal, a description of object types and properties as well as possible actions, a description of the initial state of the world, and returns as output a sequence of actions that will achieve the goal, when executed. The hypothesis we wish to verify is the following: *we can define a theory for the definition of plans for the integration of linked data whose accuracy is verifiable with respect to the needs of a particular task.*

Recently, a serious evaluation of the reliability of the linked data paradigm is emerging. This includes discussions about the capabilities of the tools for exploiting linked data [2] as well as on how this information could be effectively reused for reliable data analysis task [3].

We classify the methods for linked data integration in two categories:

(1) *goal/query oriented*: the user specifies a set of requirements in a declarative way (the query language) and data is kept where it actually is, the integration

* A special thank to Angelo Oddi (Planning and Scheduling Team PST, ISTC-CNR) which introduced me to the planning and helped me in the design of the experiment.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 445–448, 2012.

being factored at query processing level (for example [4]). This approach formalizes the requirements (the query) but the reliability of its output depends on the real-time availability of remote systems and on the limitations of the capabilities of the query language (with respect to entity linking, for example).

(2) *process/data oriented*: a user customizes a (set of) tools in order to define a process to build a dataset from the sources able to answer a set of (implicit) domain requirements (an example is LDIF [5]). This approach makes feasible to combine multiple commands for dealing with sub-tasks (like the linking of equal entities), but does not provide a way to formally express requirements and goals.

Other approaches include the formalization of prototypical tasks through reasoning patterns [6], approaching the semantic web as a unique ontology and not as linked data, and the field of ontology matching [7], which focuses on the ontological level (the OWL level) instead of the data structure (the RDF level). Existing approaches however do not deal with the problem of specifying data requirements and ensuring reliability with respect to these requirements. A goal/data oriented approach as the solution we envisage here seems not to be addressed by existing methods and tools.

2 Towards a Theory for Linked Data Harmonization

2.1 Methodology

To formulate our theory we consider four tasks[1].

1. Represent a dataset and its portions. We base our model on the concepts of `Dataset`, graph `Slice` - a *pattern* for detecting a coherent subsets of triples according to some criteria and `Symbol`, representing any RDF resource. In VoID [8] the concepts of property and class partition have been introduced, while [9] used the concept of Path - all are kind of slices in our model.

2. Model properties and operators. Properties describe the features of a dataset, or of a specific slice. For example, a property may indicate the presence of a given slice in a dataset or describe relations between symbols. For example two predicates are reversible or another one may represent the amount of values for a predicate on any subject. *Operators* encode the actions that can be performed on a dataset, for example `COPY`, `FILTER`, `APPEND`. They have *parameters* and *effects*. Parameters bind the functionality to graph properties (*preconditions*), which constrain the operator to be applicable on a specific dataset state. Effects are consequences of the executed action described in term of dataset properties. Dataset model, properties and operators constitute the planning *domain*, which encodes type of objects and possible actions involving them.

3. Model requirements. This is a description of the initial state and of the goal. The *goal* is the expression of the task in terms of properties of the *goal dataset*,

[1] Follows a synthetic description of each aspect. More details and relevant online resources are available at `http://www.enridaga.net/phd/iswc2012/`

while the *initial state* includes the properties of the *source datasets* and the relations between the symbols used in both.

4. Produce harmonization plans. We intend to simulate a data fusion process with a state of the art planner and evaluate how it may support our hypothesis (and to what extend). Then, if necessary, build our own tool for generating plans to be run by state of the art linked data frameworks, such as LDIF [5].

2.2 Evaluation

We intend to evaluate our theory and the resulting methodology in the following ways: (1) analyzing the class of harmonization situations it is able to support (a qualitative evaluation of our hypothesis); (2) doing a task based evaluation (how much effort is required with/without this approach in a given scenario? How much is the cost of interoperation of our data with data consuming tools?); (3) defining a scenario and manually executing the process using SPARQL, state of the art tools and by writing an ad-hoc program. The resulting dataset will be the *gold standard* to compare with the one produced by our tool. This should evaluate the overall approach from the user point of view.

It is a theoretical problem to understand how many real-world situations our theory may cover, so we intend to discuss also unsupported scenarios.

3 Lessons Learnt from an Initial Experiment

We defined a pilot use case starting from the following exemplary task:

> *Report about the number of tenders from the EU in public infrastructures of a specific country along with the number of citizens living in the region.*

We identified 2 data sources: (1) LOTED [10] - which contains information about countries and tenders over the years (2) EUROSTAT (via ontologycentral.com), to retrieve statistics about population. A initial attempt we considered deterministic planning[2]. A subset of the theory have been encoded as PDDL[3] *domain*, and a requirements as *problem*. As test, the Fast Downward[4] planner has been used[5]. This experiment allowed us to do a first evaluation of the feasibility of the approach. We have been able to discover a valid plan. However we needed to make several compromises in the modeling phase. There is a trade off between computational efficiency (computability) and expressivity of the domain. To make the planner find a plan, we needed to have exactly the properties, operators and objects useful to solve this single problem - nothing less and nothing

[2] For an overview of deterministic planning and recent advances in the field, see [11].
[3] The Planning Domain Definition Language, which is the de facto standard for describe the features of a deterministic planner [11].
[4] http://www.fast-downward.org/
[5] PDDL files and problem solution are available at http://www.enridaga.net/phd/iswc2012.

more. In addition, we discovered several limitations of a classic deterministic planner that we started to analyze taking PDDL as reference specification: a data integration process needs to clone, create and destroy objects (datasets are appended, slices are copied); slices have several complex relations (any slice contains potentially many others, and it could be necessary to know if a needed slice can be obtained by specializing an available one); initial knowledge can be uncertain (the planner should be able to inspect available graphs on demand): all these features are not supported by PDDL. The following steps are to complete the analysis of the requirements a planner must satisfy in order to support our theory and to implement a software able to solve harmonization problems.

References

1. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. In: Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool (2011)
2. Rivero, C.R., Schultz, A., Bizer, C., Ruiz, D.: Benchmarking the performance of linked data translation systems. In: Linked Data on the Web Workshop at WWW 2012 (2012)
3. Auer, S.: Creating knowledge out of interlinked data: making the web a data washing machine. In: Proceedings of the International Conference on Web Intelligence, Mining and Semantics, WIMS 2011. ACM, New York (2011)
4. Álvarez, J.M., Labra, J.E., Calmeau, R., Marín, Á., Marín, J.L.: Query Expansion Methods and Performance Evaluation for Reusing Linking Open Data of the European Public Procurement Notices. In: Lozano, J.A., Gámez, J.A., Moreno, J.A. (eds.) CAEPIA 2011. LNCS, vol. 7023, pp. 494–503. Springer, Heidelberg (2011)
5. Schultz, A., Matteini, A., Isele, R., Bizer, C., Becker, C.: Ldif-linked data integration framework. In: 2nd International Workshop on Consuming Linked Data, Bonn, Germany (2011)
6. Van Harmelen, F., Ten Teije, A., Wache, H.: Knowledge engineering rediscovered: towards reasoning patterns for the semantic web. In: Proceedings of the Fifth International Conference on Knowledge Capture, pp. 81–88. ACM (2009)
7. 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)
8. Alexander, K., Hausenblas, M.: Describing linked datasets-on the design and usage of void, the vocabulary of interlinked datasets. In: International World Wide Web Conference on Linked Data on the Web Workshop, LDOW 2009 (2009)
9. Presutti, V., Aroyo, L., Adamou, A., Schopman, B., Gangemi, A., Schreiber, G.: Extracting core knowledge from linked data. In: The 2nd Int. Workshop on Consuming Linked Data (COLD 2011) at ISWC 2011 (2011)
10. Valle, F., dAquin, M., Di Noia, T., Motta, E.: Loted: Exploiting linked data in analyzing european procurement notices. In: Proceedings of the 1st EKAW Workshop on Knowledge Injection into and Extraction from Linked Data (2010)
11. Gerevini, A.E., Haslum, P., Long, D., Saetti, A., Dimopoulos, Y.: Deterministic planning in the fifth international planning competition: Pddl3 and experimental evaluation of the planners. Artificial Intelligence 173(5-6), 619–668 (2009)

Knowledge Pattern Extraction and Their Usage in Exploratory Search*

Andrea Giovanni Nuzzolese[1,2]

[1] STLab-ISTC Consiglio Nazionale delle Ricerche, Rome, Italy
[2] Dipartimento di Scienze dell'Informazione, Università di Bologna, Italy

Abstract. Knowledge interaction in Web context is a challenging problem. For instance, it requires to deal with complex structures able to filter knowledge by drawing a meaningful context boundary around data. We assume that these complex structures can be formalized as Knowledge Patterns (KPs), aka frames. This Ph.D. work is aimed at developing methods for extracting KPs from the Web and at applying KPs to exploratory search tasks. We want to extract KPs by analyzing the structure of Web links from rich resources, such as Wikipedia.

1 Problem Statement and Related Work

In the vision of the Semantic Web agents are supposed to interact with Web knowledge in order to help humans in solving knowledge-intensive tasks. Though Linked Data is a breakthrough in Semantic Web it is still hard to build contextualized views over data, which would allow to select relevant knowledge for a specific purpose, i.e., to draw relevant boundaries around data. Let us suppose we are interested in events involving Arnold Schwarzenegger in the artistic context. For example, the movies that Arnold Schwarzenegger starred before starting his political career. We need to recognize "starring" situations over the knowledge about Arnold Schwarzenegger available on the Web. Such situations are represented as complex structures that relate entities and concepts according to a unifying view, e.g., Arnold Schwarzenegger having role of actor in movies during a time period. Such complex structures can be exploited for supporting a variety of knowledge interaction tasks, such as decision support, content recommendation, exploratory search, content summarization, question answering, information visualization, interface design, etc. These complex knowledge structures have been identified and described by Minsky [5], who proposed to conceptualize them as *frames*. Frames, known as *Knowledge Patterns* (KPs), have been reproposed in Semantic Web [3]. A KP can be briefly defined as "a formalized schema representing a structure that is used to organize our knowledge, as well as for interpreting, processing or anticipating information". This Ph.D. work aims at developing methods for discovering and extracting KPs on the Web and exploit them for supporting exploratory search. Exploratory search is well known in literature. [9] is a survey that presents examples of exploratory

* Advisors: Paolo Ciancarini, Valentina Presutti, and Aldo Gangemi.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 449–452, 2012.

search on the Semantic Web. Existing approaches perform exploratory search by organizing knowledge according to scatterd elements like classes or relations. Differently from them, we want to experiment with exploratory search based on KPs as a solution for providing summarizations and navigating knowledge.

We focus on analyzing and studying KPs in Wikipedia and some Linked Data data sets. Our aim is: (i) to develop methods for KP discovering and extraction on the Web, and (ii) to apply KP-based strategies to exploratory search and prove that the user experience improves as compared to state of the art tools.

We hypothesize that: (i) KPs can be discovered by analyzing the linking structure of Web resources. [1] In fact, linking things to other things is a typical cognitive metaphor used by humans on the Web for organizing knowledge. Hence, Web links (either hypertextual links or RDF triples) convey rich knowledge that can be used for extracting KPs. (ii) Exploratory search would benefit from using KPs. In our vision KPs can be applied to exploratory search in order to filter knowledge and guide a user to explore contextual relevant knowledge.

2 KP Extraction and Usage in Exploratory Search

We want to develop methods for discovering and extracting KPs by analyzing the linking structure of Web resources and exploit them for supporting exploratory search.

KPs sources. We identify at least two different kind of sources from which to extract KPs:

- **already existing KP or frame sources.** Examples are the FrameNet project [1] or the Ontology Design Pattern project [2]. These resources are modelled with a top-down approach and they are typically designed by domain experts, that firstly formalize the domain semantics and then move to data. We are aimed at transforming these resources in order to represent them homogeneously as they are expressed with different formats and semantics. Possible solutions are reengineering, refactoring based on transformation rules, key concept identification, ontology mapping, etc. Initial results have been obtained and in [6] we present a solution for reengineering FrameNet in order to produce (i) a LOD data set [3] and (ii) a collection of reusable KPs available as OWL2 ontologies; (ii)
- **the Web.** The Web provides a lot of heterogeneous sources from which KPs can be extracted. In order to narrow them we need to make some assumption. We want to take into account only text resources including links that (i) are associated with structured data, e.g., Linked Data, and (ii) can be formally interpreted. Wikipedia fits perfectly these assumptions. In fact, it provides rich content deriving from a collaboratively crowd sourcing performing an

[1] Under a number of assumptions (see Section 2).
[2] http://www.ontologydesignpatterns.org
[3] http://stlab.istc.cnr.it/stlab/FrameNetKCAP2011

encyclopedic task. Furthermore, it has a RDF dump in Linked Data, i.e., DBpedia [4], and ontologies, e.g., YAGO and the DBpedia Ontology, that allow to give formal interpretation to data. For these reasons we want to focus on the extraction of KPs from Wikipedia.

For this purpose some work has been done. In [7] we present a method that we have defined for extracting KPs from Wikipedia by analysing links and their interpretation through the DPpedia Ontology. [4] The analysis of results shows a bias due to a large number of untyped entities in DBpedia. For that reason we have investigated inductive as well abductive approaches for automatically typing DBpedia entities [8]. Experiments show that NLP is needed in order to address this task. Hence, we have investigated a NLP-based approach showing good resutls [2] [5]. With this approach we plan to type all DBpedia entities and then re-run the extraction of KP from Wikipedia.

KPs in Exploratory Search. In Exploratory Search tasks users want to discover what they are looking for by exploring knowledge. In some case, they have only a vague idea about what the nature of their search is. In these cases it is important to select relevant data in order to help users to filter and to summarize knowledge during their search. KPs can be used in exploratory search tasks for drawing boundaries around data in order to provide contextual relevant knowledge. This means that, depending on the context, different KPs could be automatically selected on the same data in order to make emerge the knowledge that is relevant to a user's search. We have started to experiment KPs in Exploratory Search with Aemoo [6]. Aemoo selects and organize the core knowledge about a DBpedia entity by applying KPs extracted from Wikipedia as lenses over data.

3 Evaluation

We want to evaluate both the performance of our extraction method and the quality of extracted KPs. On one hand, the evaluation of the method's performance can be easily analyzed by taking into account the execution time of the extraction algorithm. On the other, it is less clear how to evaluate extracted KPs. We think that a solution for evaluating the quality of extracted KPs derives from the combination of the following methods:

Gold standard based evaluation. We want to ask to a group of at least 3 expert users in knowledge representation to build a sample of KPs on some subset of some specific domain. Each expert has to formalize KPs with respect to the tasks proposed, e.g. to provide the core concepts that summarize what an

[4] We have extracted 231 KPs representing the core knowledge used for describing a specific type of entity.

[5] Typing precision is around 86%.

[6] `http://aemoo.org`. Aemoo partecipated to the last Semantic Web Challenge reaching the final round.

airplane is. The set of KPs proposed will be the same for each expert in order to evaluate the inter-rater agreement among experts. KPs over a certain threshold of agreement will be the gold standard. Hence, by applying our extraction method on the same set will be possible to compare the results to the gold standard for evaluating precision and recall of our method;

User based evaluation. After the extraction of KPs we want to ask to a group of heterogeneous users in terms of education and expertise in knowledge representation to evaluate them. The evaluation will be conducted by asking users to fill a survey about the capacity of assigned KPs to be relevant to some context, exhaustive in summarizing knowledge, appropriate, etc. This will give an idea of soundness and accuracy of extracted KPs;

Task based evaluation. A comparison between an exploratory search application based on KPs and other existing exploratory search applications will be used for evaluating the effectiveness of KPs in exploratory search tasks. In this case will be asked to a group of users to solve tasks of knowledge exploration, learning, summarization, relation finding, definition extraction, etc.

References

1. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The Berkeley FrameNet Project. In: Proc. of the 17th International Conference on Computational Linguistics, Morristown, NJ, USA, pp. 86–90 (1998)
2. Gangemi, A., Nuzzolese, A.G., Presutti, V., Draicchio, F., Musetti, A., Ciancarini, P.: Automatic Typing of DBpedia Entities. In: Proc. of the 11th International Semantic Web Conference (ISWC 2012). Springer, Boston (2012)
3. Gangemi, A., Presutti, V.: Towards a Pattern Science for the Semantic Web. Semantic Web 1(1-2), 61–68 (2010)
4. Lehmann, J., Bizer, C., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A Crystallization Point for the Web of Data. Journal of Web Semantics 7(3), 154–165 (2009)
5. Minsky, M.: A Framework for Representing Knowledge. In: Winston, P. (ed.) The Psychology of Computer Vision, McGraw-Hill (1975)
6. Nuzzolese, A.G., Gangemi, A., Presutti, V.: Gathering Lexical Linked Data and Knowledge Patterns from FrameNet. In: Proc. of the 6th International Conference on Knowledge Capture (K-CAP), Banff, Alberta, Canada, pp. 41–48 (2011)
7. Nuzzolese, A.G., Gangemi, A., Presutti, V., Ciancarini, P.: Encyclopedic Knowledge Patterns from Wikipedia Links. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 520–536. Springer, Heidelberg (2011)
8. Nuzzolese, A.G., Gangemi, A., Presutti, V., Ciancarini, P.: Type inference through the analysis of Wikipedia links. Submitted at the WWW 2012 Workshop on Linked Data on the Web (2012), http://stlab.istc.cnr.it/documents/papers/ldow2012.pdf
9. Uren, V., Lei, Y., Lopez, V., Liu, H., Motta, E., Giordanino, M.: The usability of semantic search tools: a review. The Knowledge Engineering Review 22(4), 361–377 (2007)

SPARQL Update for Complex Event Processing

Mikko Rinne

Distributed Systems Group,
Department of Computer Science and Engineering,
Aalto University, School of Science, Finland
mikko.rinne@aalto.fi

Abstract. Complex event processing is currently done primarily with proprietary definition languages. Future smart environments will require collaboration of multi-platform sensors operated by multiple parties. The goal of my research is to verify the applicability of standard-compliant SPARQL for complex event processing tasks. If successful, semantic web standards RDF, SPARQL and OWL with their established base of tools have many other benefits for event processing including support for interconnecting disjoint vocabularies, enriching event information with linked open data and reasoning over semantically annotated content. A software platform capable of continuous incremental evaluation of multiple parallel SPARQL queries is a key enabler of the approach.

Keywords: Complex event processing, SPARQL, RDF, Rete-algorithm.

1 Smart Cities Need SPARQL

Smart environments of the future will need to interconnect billions of sensors based on platforms from multiple vendors operated by different companies, public authorities or individuals. To mitigate the need for overlapping sensors producing duplicate measurements, interoperation of different platforms should be maximized. Highly distributed, loosely coupled solutions based on common standards are needed in such open environments. Event processing systems based on proprietary definition languages have challenges to adapt to multi-vendor contexts.

The benefit of RDF in complex event processing is that it provides a flexible representation of heterogeneous events in an open distributed environment, where new sensors must be able to add new information fields without breaking compability with existing applications. SPARQL, tailor-made to query RDF, was augmented in SPARQL 1.1 Update by the powerful capability to insert selected data into named triple stores. When combined with a continuous query processing engine, INSERT gives SPARQL queries memory and capability to communicate and collaborate with each other. As a result, interconnected SPARQL queries can be used to create complex event processing applications, capable of handling layered and heterogeneous representations of event instances. When taking into account their other benefits, semantic web standards RDF, SPARQL and OWL form a very promising base for complex event processing.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 453–456, 2012.

In the Distributed Systems Group we are working on an incremental continuous SPARQL query processor based on the Rete-algorithm [5]. The INSTANS[1] platform supports selected parts of SPARQL 1.1 Query and Update specifications. The first generation of INSTANS was coded on Scala[2] [1, 8–10]. INSTANS is currently being ported to Lisp, where the Rete-net is compiled through macro expansion in the setup phase into executable Lisp code. The Scala-version reached notification delays of 5-14 ms for the cases tested, but first measurements indicate that the Lisp-version would be 100-200 times faster.

In event processing it is equally important to detect the events which didn't happen as the ones that did. Missing events are sometimes referred to as "no-events" or "absence patterns" [4]. A "timed events" mechanism is implemented with special predicate values used to mark input to a timer-queue. Events in the timer queue can be set to trigger either after a relative time or at absolute points in time. A triggered event can be used to set a new timed event, supporting periodic operations. The whole interface is SPARQL-compliant, with the triggering of a timer changing a corresponding triple predicate from "waiting" to "triggered", the change being detectable in a SPARQL query.

2 Related Activities

Other research teams have been looking into streaming SPARQL, e.g. C-SPARQL[3] [3] and CQELS[4] [7]. Some differences to our approach are:

- Individual triples: "Data stream processing" focuses on individual time-annotated triples. We are assuming heterogeneous event formats, where it may not be known at the time of writing an event processing application, what information future sensors are going to include into an event. Possibility to layer events is also of critical importance.
- Extensions: All other solutions extend SPARQL, typically with time-based windowing or processing a stream order of data. We have used no extensions.
- Repetition of queries: Defined on windows based on time or number of triples and a repetition rate, with which queries will be re-run. Our approach is based on continuous and incremental matching of queries, where a particular segment can be isolated by filtering.

Sparkweave[5] [6] applies SPARQL queries to RDF format data using an extended Rete-algorithm, but focuses on inference and fast data stream processing of individual triples instead of heterogeneous events. Sparkweave v. 1.1 also doesn't have support for SPARQL 1.1 features such as SPARQL Update.

The Prolog-based ETALIS has a SPARQL compiler front-end called "EP-SPARQL" [2], but it is more limited than the Prolog notation and doesn't

[1] Incremental eNgine for STANding Sparql, `http://cse.aalto.fi/instans/`
[2] `http://www.scala-lang.org/`
[3] `http://streamreasoning.org/download`
[4] `http://code.google.com/p/cqels/`
[5] `https://github.com/skomazec/Sparkweave`

support (at the time of writing) SPARQL 1.1 features such as SPARQL Update, which is critical for our study. EP-SPARQL concentrates on operations on event sequences.

No other system based on collaborative SPARQL queries is known to us. Current systems in the research community are mainly concentrating on running one query at a time[6]. Even the ones allowing to register multiple simultaneous queries are not expecting the queries to communicate during runtime.

3 Measuring Success

Our event processing work focuses on two main components:

1. **Approach:** Multiple collaborating SPARQL queries and update rules processing heterogeneous events expressed in RDF.
2. **Implementation (INSTANS):** Incremental continuous query engine based on the Rete-algorithm

The overall target of the approach is that it would be easy to create and maintain efficient event processing applications for open and heterogeneous environments. Research questions are related to finding good principles and patterns for SPARQL queries used in event processing, creating a mapping to SPARQL for the main operations needed in event processing (e.g. filtering, splitting, enrichment, aggregation, pattern detection), developing efficient methods of linking event information with background knowledge, adopting ontology-based inference mechanisms in event processing and comparing to other event processing approaches.

An example application "Fast Flowers Delivery" is presented in [4]. It is a logistics management system, where flower stores send requests to an independent pool of drivers to send flowers to customers. Drivers are selected based on location and ranking. Ranking involves a periodic reporting system. Our next target is to verify that SPARQL has all the elements in place to support also this kind of event processing applications. Once the example cases have been confirmed to work, generalized solution patterns for the complex event processing elements found in literature using SPARQL building blocks will be defined.

Measuring the success of the implementation can be approached with:

- Implementation efficiency (compared to other Rete implementations)
- Algorithmic efficiency (Rete compared to other ways of processing SPARQL queries)
- Performance of the approach (compared to other event processing systems)

Targets for empirical studies are e.g. latency (notification time), throughput, memory consumption, system load, continuous operation over extended time periods and energy efficiency (especially when operating over sensors). In addition to empirical comparisons, this work is expected to provide answers for

[6] e.g. Jena (http://incubator.apache.org/jena/), Sesame (http://www.openrdf.org/)

understanding of garbage build-up in the system and for solutions to improve performance compared to basic Rete.

As a first step comparisons with C-SPARQL have been carried out and documented on the project homepage using an example "close friends" service[8], but since C-SPARQL is based on repeated execution of queries on windows, the results are very difficult to compare. A "notification delay" in C-SPARQL is dominated by the window repetition rate. Trying to minimize delay by increasing repetition rate leads to wasted computing resources and duplicate detections. Even doing so, the format of C-SPARQL only allows to execute queries once per second (far too often for most applications), whereas the notification delays for INSTANS have been clocking in at 5-14 ms (depending on hardware).

Both the approach and the implementation would be involved in testing the ease of deployment and management of the system in a distributed way in an open environment. Related questions are the processing and memory requirements of the implementation, arrangements for communication between distributed deployments and any security-related issues specific to the approach. Based on our verifications both the approach and INSTANS look very promising.

References

1. Abdullah, H., Rinne, M., Törmä, S., Nuutila, E.: Efficient matching of SPARQL subscriptions using Rete. In: Proceedings of the 27th Symposium on Applied Computing, Riva del Garda, Italy (March 2012)
2. Anicic, D., Fodor, P., Rudolph, S., Stojanovic, N.: EP-SPARQL: a unified language for event processing and stream reasoning. In: WWW 2011, pp. 635–644. ACM, Hyderabad (2011)
3. Barbieri, D.F., Braga, D., Ceri, S., Grossniklaus, M.: An execution environment for C-SPARQL queries. In: Proceedings of the 13th International Conference on Extending Database Technology - EDBT 2010, Lausanne, Switzerland, p. 441 (2010)
4. Etzion, O., Niblett, P., Luckham, D.: Event Processing in Action. Manning Publications (July 2010)
5. Forgy, C.L.: Rete: A fast algorithm for the many pattern/many object pattern match problem. Artificial Intelligence 19(1), 17–37 (1982)
6. Komazec, S., Cerri, D.: Towards Efficient Schema-Enhanced Pattern Matching over RDF Data Streams. In: 10th ISWC, Bonn, Germany. Springer (2011)
7. Le-Phuoc, D., Dao-Tran, M., Parreira, J.X., Hauswirth, M.: A Native and Adaptive Approach for Unified Processing of Linked Streams and Linked Data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 370–388. Springer, Heidelberg (2011)
8. Rinne, M., Abdullah, H., Törmä, S., Nuutila, E.: Processing Heterogeneous RDF Events with Standing SPARQL Update Rules. In: Meersman, R., Panetto, H., Dillon, T., Herrero, P. (eds.) OTM 2012, Part II. LNCS, vol. 7566, pp. 797–806. Springer, Heidelberg (2012)
9. Rinne, M., Nuutila, E., Törmä, S.: INSTANS: High-Performance Event Processing with Standard RDF and SPARQL. Poster in International Semantic Web Conference 2012, Boston, MA (2012)
10. Rinne, M., Törmä, S., Nuutila, E.: SPARQL-Based Applications for RDF-Encoded Sensor Data. In: 5th International Workshop on Semantic Sensor Networks (2012)

Online Unsupervised Coreference Resolution for Semi-structured Heterogeneous Data*

Jennifer Sleeman

Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County, Baltimore, MD 21250 USA
jsleem1@cs.umbc.edu

Abstract. A pair of RDF instances are said to corefer when they are intended to denote the same thing in the world, for example, when two nodes of type foaf:Person describe the same individual. This problem is central to integrating and inter-linking semi-structured datasets. We are developing an online, unsupervised coreference resolution framework for heterogeneous, semi-structured data. The online aspect requires us to process new instances as they appear and not as a batch. The instances are heterogeneous in that they may contain terms from different ontologies whose alignments are not known in advance. Our framework encompasses a two-phased clustering algorithm that is both flexible and distributable, a probabilistic multidimensional attribute model that will support robust schema mappings, and a consolidation algorithm that will be used to perform instance consolidation in order to improve accuracy rates over time by addressing data spareness.

1 Introduction

When performing coreference resolution, as it relates to knowledge representation, one tries to determine if an instance represents a real-world entity, typically defined in a knowledge base. Various techniques have been used to perform coreference resolution including both supervised and unsupervised methods, however many approaches tend to function based on a batch data set, assume the schemas are accessible a priori and often neglect the topic of heterogeneity. In many complex computing environments, particularly among scientific and intelligence communities, data schemas may not be known a priori, data is more typically acquired over time in parts rather than all at once and often heterogeneous, i.e. originating from multiple sources. In order to support these complexities, coreference resolution algorithms need to account for this online behavior and need to support heterogeneous data. Furthermore, very little focus is given to the effects of temporal object consolidation, i.e., the merging of groups of entities over time, connected by coreferent relations.

Given the problem of online coreference resolution for heterogeneous data, an unsupervised or semi-supervised learning approach is required to support the dynamic nature of such an environment; in particular we will show that a two-phased clustering algorithm and knowledge base reasoning will provide both a flexible and scalable way to support this model with accuracy rates that approach supervised and offline methods.

* Advisor: Tim Finin.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 457–460, 2012.

2 Related Work

Though there is a significant amount of research in this area[15,14,10,8,7], we highlight a few more recent works. Araujo et al.[1] support instance matching specifically for interlinking data sets within the Linked Open Data Cloud. This work is consistent with others in that it assumes a static environment. Hu et al. [4] uses language axioms to generates a kernel based on the OWL vocabulary and ranks coreferent pairs based on confidence measures. Using language axioms can be a limitation, often data does not strictly conform to language axioms and in many cases, schemas are not accessible. In our previous work[12] only a small portion of our data contained axioms that could be used for this type of analysis. Rao et al.[9] highlight a cross document coreference resolution approach for streaming data that uses a clustering algorithm based on a doubling clustering algorithm which is similar to our approach; we however use a two-phased approach to clustering to reduce the computational costs. Song et al.[13] describe an approach to candidate selection that learns attributes that occur most frequently across their data set and a matching algorithm to designate coreferent pairs. Though supportive of heterogeneous data, the candidate selection process is limited by the key designation which could underperform when working with sparse data. It is also not clear how this approach could support temporal changes. Both Hogan et al.[3] and Shi et al.[11] do not address conflicts and rely upon inverse functional properties to perform object consolidation, which could be problematic since inverse functional properties are not always present. Our work does not rely on inverse functional properties, we address conflicts and we are specifically evaluating how consolidated instances will improve the accuracy of subsequent coreference resolution over time.

3 Approach

Our research makes four major research contributions that work together to achieve an effective approach to perform online coreference resolution. We will build a system that will bring together these contributions.

Research Contribution: Multi-dimensional Model: We are developing a probabilistic multi-dimensional attribute model that will support heterogeneous data by deriving meaning from the data and schemas using five dimensions. Dissimilarity and similarity functions are used to compare attribute values both at the individual pair level and across vectors. For example, if we are comparing two attributes that represent a person's name, we would likely use a distance function to determine how dissimilar the two strings are to each other. Structural properties take into consideration the graph itself. Statistical properties involve analytics that use knowledge of the distribution of values for an attribute. Ontological definitions use axioms defined in the ontology. Contextual information provides macro-level information that supports conceptual heterogeneity, for example using neighborhood graphs.

We are currently experimenting with a Bayesian model to represent these five dimensions. We are implementing this model to support our second phase of clustering to determine which instances should be part of the same cluster, rather than using a single distance measure. We also use attribute mapping to classify attribute types for

subsequent processing and for specializing the five dimensions. As the attribute model is used over time, we plan to develop optimal models based on data types. For example, we could measure the distance between two geographic locations using a Euclidean distance [2] rather than using a distance function that calculates the number of transitions from one string to another such as Levenshtein [5].

Research Contribution: Two-Phased Clustering: We are developing a new clustering algorithm that performs clustering in two phases. The first phase acts as a filter resulting in neighborhoods of related instances and the second phase performs the clustering of coreferent instances. The complexity of clustering algorithms can range from $O\left(n^2\right)$ to $O\left(n^3\right)$. A first phase clustering that is computationally less expensive can reduce the size of the data that must be partitioned by the second phase of clustering, as shown in previous work using a canopy approach [6]. We are building the first phase to work at a complexity under $O\left(n^2\right)$ that will roughly partition instances into neighborhoods of likeness. Currently we use a bag of words model and a canopy-like approach [6]. The second phase of clustering is applied to each partition and will use our defined attribute model to perform coreferent-based clustering of each neighborhood cluster. Currently we use agglomerative hierarchical clustering with distance metrics only, and we are developing our new algorithm to support the integration of our attribute model.

Research Contribution: Instance Consolidation: In our model, to support temporal changes, the concept of an instance is abstractly defined as a single instance or a cluster of instances that are coreferent. Given our two-phased clustering work, the results are clusters where in each cluster, we symbolically link instances using a weighted measure to allow for cluster changes over time. Features among instances are weighted in order to support subsequent instance matching using dominate cluster features. We are currently experimenting with a number of feature reduction algorithms to support subsequent instance matching.

Research Contribution: Coreference Resolution Benchmark: A challenging problem related to testing coreference resolution systems is finding data that has enough positive test cases to formulate a valid test. For this reason we are developing a set of Semantic Web coreference resolution benchmarks that could be shared with the research community. The benchmarks will exercise the coreference resolution algorithm from different perspectives.

4 Evaluation

We will evaluate our clustering algorithm with respect to offline supervised methods as a way to show comparison F-Measure scores using both the Ontology Alignment Evaluation Initiative (OAEI) data set and our custom data sets. In addition, we will measure the effectiveness of this algorithm and how it can process data incrementally over time. We will also evaluate the effectiveness of using both attribute typing and a probabilistic model by performing precision and recall comparisons. We will evaluate consolidation by determining if the consolidated clusters improve the accuracy of the system over time.

5 Conclusion

Data is noisy, heterogeneous in nature, incrementally processed, large and often based on schemas that are not known a priori. To support these complexities we are developing algorithms that work together under a common framework including a probabilistic attribute model to address the aspects such as noisiness and heterogeneity, a two-phased clustering algorithm that supports an online model to address working with data that is incrementally processed over time and an instance consolidation algorithm that will improve matching over time and addresses data spareness.

References

1. Araujo, S., Hidders, J., Schwabe, D., de Vries, A.P.: Serimi resource description similarity, rdf instance matching and interlinking. CoRR, Vol. abs/1107.1104 (2011)
2. Weisstein, E.: Distance. From MathWorld–A Wolfram Web Resource (1999-2012) (accessed May 2012)
3. Hogan, A., Harth, A., Decker, S.: Performing object consolidation on the semantic web data graph. In: Proc. I3: Identity, Identifiers, Identification. Workshop at 16th Int. World Wide Web Conf. (February 2007)
4. Hu, W., Qu, Y., Sun, X.: Bootstrapping object coreferencing on the semantic web. Journal of Computer Science and Technology 26(4), 663–675 (2011)
5. Levenshtein, V.: Binary codes capable of correcting deletions, insertions, and reversals, vol. 10(8), pp. 707–710 (1966)
6. McCallum, A., Nigam, K., Ungar, L.: Efficient clustering of high-dimensional data sets with application to reference matching. In: The Sixth International Conference on Knowledge Discovery and Data Mining, ACM SIGKDD, pp. 169–178 (2000)
7. Nikolov, A., Uren, V., Motta, E.: Data linking: Capturing and utilising implicit schema level relations. In: International Workshop on Linked Data on the Web (2010)
8. Nikolov, A., Uren, V., Motta, E., de Roeck, A.: Overcoming Schema Heterogeneity between Linked Semantic Repositories to Improve Coreference Resolution. In: Gómez-Pérez, A., Yu, Y., Ding, Y. (eds.) ASWC 2009. LNCS, vol. 5926, pp. 332–346. Springer, Heidelberg (2009)
9. Rao, D., McNamee, P., Dredze, M.: Streaming cross document entity coreference resolution. In: International Conference on Computational Linguistics (COLING). Coling 2010 Organizing Committee, pp. 1050–1058 (November 2010)
10. Seddiqui, M.H., Aono, M.: Ontology instance matching by considering semantic link cloud. In: 9th WSEAS International Conference on Applications of Computer Engineering (2010)
11. Shi, L., Berrueta, D., Fernandez, S., Polo, L., Fernandez, S.: Smushing rdf instances: are alice and bob the same open source developer? In: Proc. 3rd Expert Finder workshop on Personal Identification and Collaborations: Knowledge Mediation and Extraction, 7th Int. Semantic Web Conf. (November 2008)
12. Sleeman, J., Finin, T.: Computing foaf co-reference relations with rules and machine learning. In: The Third International Workshop on Social Data on the Web, ISWC (November 2010)
13. Song, D., Heflin, J.: Automatically Generating Data Linkages Using a Domain-Independent Candidate Selection Approach. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 649–664. Springer, Heidelberg (2011)
14. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Silk - a link discovery framework for the web of data. In: Proc. 2nd Workshop on Linked Data on the Web, Madrid, Spain (April 2009)
15. Yatskevich, M., Welty, C., Murdock, J.: Coreference resolution on rdf graphs generated from information extraction: first results. In: The ISWC 2006 Workshop on Web Content Mining with Human Language Technologies (2006)

Composition of Linked Data-Based RESTful Services

Steffen Stadtmüller

Institute of Applied Informatics and Formal Descriptions Methods (AIFB)
Karlsruhe Institute of Technology, Germany
Steffen.Stadtmueller@kit.edu

Abstract. We address the problem of developing a scaleable composition framework for Linked Data-based services, that retains the advantages of the loose coupling fostered by REST.

1 Problem Statement

The Linking Open Data community has gained momentum over the last years. At the same time there is a strong movement in the Web community toward a resourceful model of services based on Representational State Transfer (REST [3]) which propagates the primacy of loose coupling. Flexibility, adaptivity and robustness are direct consequences from the loose coupling and are achieved with links between resources, which allow clients to navigate from one resource to another during their interaction [11]. REST is particularly useful for software architectures in distributed data driven environments such as the Web [10].

Following the motivation to look beyond the exposure of fixed datasets, an extension of Linked Data with REST technologies has been proposed and explored for some time [1,16].

The composition of RESTful resources originating from different providers suffers particularly from the necessary manual effort to use them. The reliance on natural language descriptions has led to mashup designs in which programmers are forced to write glue code with little or no automation and to manually consolidate and integrate the exchanged data.

Our contributions toward a scaleable loosely coupled composition will be

- an analysis of how self-descriptive resources have to be designed to enable composition;
- a service model for REST based on state transition systems as formal grounding for our composition;
- a declarative rule-based execution language to allow an intuitive specification of the interaction with resources from different providers;
- an execution engine as artifact to perform the defined interactions, which we want to evaluate with regard to scaleability.

The rest of the paper is structured as follows: In Section 2 we detail the existing work. In Section 3 we describe the methods with which we intend to leverage the advantages of Linked Data based REST architectures. We conclude in Section 4.

P. Cudré-Mauroux et al. (Eds.): ISWC 2012, Part II, LNCS 7650, pp. 461–464, 2012.

2 Related Work

Pautasso introduces an extension to BPEL [9] to allow a composition of REST and traditional web services.

There are several approaches that extend the existing WS-* stack with semantic capabilities by leveraging ontologies and rule-based descriptions (e.g., [14,2,6]). In contrast to WS-* are REST architectures build around another kind of abstraction: the resource. Therefore our approach is more focused on resource/data centric scenarios in distributed environments (e.g., in the Web).

RESTdesc [15] is an approach in which RESTful Linked Data resources are described in N3-Notation. The composition of resources is based on an N3 reasoner and stipulates manual interventions of users to decide which links should be followed.

Hernandez et al. [5] proposes a model for semantically enabled REST services as a combination of pi-calculus and an extension of triple space computing by Simperl et al. [12]. Similar to the idea of triple spaces is the composition of RESTful Linked Data resources in a process space, proposed by Krummenacher et al. [7] based on resources descriptions using graph patterns. Speiser and Harth [13] propose similar descriptions for Linked Data Services. Our approach shares the idea that graph pattern described resources read input from and write output to a shared space. We want to improve on this approach by providing a rigid service model and a more explicit way of defining the interaction with resources.

3 Methodology

In this section, we describe in more detail how we want to address the challenges we face in the development of a flexible and scalable composition framework.

3.1 Resource Descriptions

In a RESTful interaction with Linked Data resources only the HTTP methods can be applied to the resources. The semantics of the HTTP methods itself is defined by the IETF[1] and do not need to be explicitly described.

The state of Linked Data resources is expressed with RDF. It is sensible to serialise the input data, i.e., data that is submitted to resources to manipulate their state, in RDF as well. To convey the resulting state change after application of a HTTP method we use RDF output messages. In previous work [8] we analysed the potential of graph patterns, based on the syntax of SPARQL, to describe required input as well as their relation to output messages. The resulting graph pattern descriptions are attached to the resource. Therefore the resources stay self-descriptive.

[1] http://www.ietf.org/rfc/rfc2616.txt

3.2 REST Service Model

A REST service can be identified with the resources it exposes. An interaction within a REST architecture is based on the manipulation of the states of the exposed resources.

We develop a service model, that allows to formalise the functionalities exposed by a service based on Linked Data resources. The formal service model serves as rigid specification of how the use of individual HTTP methods influences resource states and how these state changes are conveyed to interacting clients.

We model a Linked Data-based RESTful service as a REST state transition system (RSTS). A state in the RSTS is defined as the set of states of all resources that are exposed by the service. The transitions between states are described with state change functions and output functions for every HTTP method respectively. The intuition behind the state change functions is that a state transition in the RSTS is effected by influencing resource states with HTTP methods. The intuition behind output functions is, that the application of an HTTP method on a resource also results in a defined output, that communicates the effected state change to the interacting client with an RDF message.

3.3 Execution Language

To allow programmers to formalise their desired interactions we develop a declarative rule-based execution language. The head of a rule corresponds to an update function of the RSTS in that they describe an HTTP method that is to be applied to a resource. The rule bodies are conjunctive queries that allow programmers to express their intention under which condition a method is to be applied. The use of conjunctive queries is motivated by the idea that clients have to maintain a knowledge space (KS) in which they store their knowledge about the states of the resources they interact with [7]. KS is filled with the RDF data the client receives after applying an HTTP method, as defined by the output functions of the RSTS.

We plan to develop an interpreter for our execution rule language as execution engine that can be integrated in applications. To achieve a fast scalable interpreter we plan to build the execution engine with a query engine based on the Rete algorithm [4], which allows a multithreaded, parallel evaluation of multiple queries.

We want to evaluate the performance of our engine with regard to (1) the amount of the communicated data, (2) the number of the composed services, (3) the complexity of the queries. We intent to implement several composition scenarios with a focus on real world services. We want to measure the execution time of the scenario implementations and compare the performance with implementations of the same scenarios based on standard SPARQL query engines, with function mapping[2] for remote procedure calls, and other production rule engines (e.g., drools[3], JESS[4]).

[2] http://www.w3.org/TR/rdf-sparql-query#FunctionMapping
[3] http://www.jboss.org/drools/
[4] http://www.jessrules.com/

4 Conclusion

We have proposed to exploit the advantages resulting from the combination of REST architectures and Linked Data for a composition framework for REST services. We have scetched a declarative rule-based execution language with an with a state transition system as formal grounding and the challenges we address with this language, as well as an execution engine. For evaluation we intend to analyse real world scenarios build with existing services.

References

1. Berners-Lee, T.: Read-write linked data (August 2009), http://www.w3.org/DesignIssues/ReadWriteLinkedData.html
2. 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 (2006)
3. Fielding, R.: Architectural Styles and the Design of Network-based Software Architectures. Ph.D. thesis, University of California, Irvine (2000)
4. Forgy, C.: Rete: A fast algorithm for the many pattern/many object pattern match problem. AIJ 19(1), 17–37 (1982)
5. Hernández, A.G., García, M.N.M.: A formal definition of restful semantic web services. In: WS-REST, pp. 39–45 (2010)
6. Kopecky, J., Vitvar, T., Fensel, D.: Microwsmo: Semantic description of restful services. Tech. rep., WSMO Working Group (2008)
7. Krummenacher, R., Norton, B., Marte, A.: Towards Linked Open Services and Processes. In: Berre, A.J., Gómez-Pérez, A., Tutschku, K., Fensel, D. (eds.) FIS 2010. LNCS, vol. 6369, pp. 68–77. Springer, Heidelberg (2010)
8. Norton, B., Stadtmüller, S.: Scalable discovery of linked services. In: RED (2011)
9. Pautasso, C.: Restful web service composition with bpel for rest. DKE 68(9), 851–866 (2009)
10. Pautasso, C., Wilde, E.: Why is the web loosely coupled?: a multi-faceted metric for service design. In: WWW, pp. 911–920 (2009)
11. Richardson, L., Ruby, S.: RESTful Web Services. O'Reilly Media (2007)
12. Simperl, E., Krummenacher, R., Nixon, L.: A Coordination Model for Triplespace Computing. In: Murphy, A.L., Dell'Acqua, P. (eds.) COORDINATION 2007. LNCS, vol. 4467, pp. 1–18. Springer, Heidelberg (2007)
13. Speiser, S., Harth, A.: Integrating Linked Data and Services with Linked Data Services. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part I. LNCS, vol. 6643, pp. 170–184. Springer, Heidelberg (2011)
14. Studer, R., Grimm, S., Abecker, A. (eds.): Semantic Web Services: Concepts, Technologies, and Applications. Springer (2007)
15. Verborgh, R., Steiner, T., Deursen, D.V., de Walle, R.V., Valls, J.G.: Efficient Runtime Service Discovery and Consumption with Hyperlinked RESTdesc. In: NWeSP (2011)
16. Wilde, E.: Rest and rdf granularity (May 2009), http://dret.typepad.com/dretblog/2009/05/rest-and-rdf-granularity.html

Author Index