

Elena Simperl Philipp Cimiano
Axel Polleres Oscar Corcho
Valentina Presutti (Eds.)

LNCS 7295

The Semantic Web: Research and Applications

9th Extended Semantic Web Conference, ESWC 2012
Heraklion, Crete, Greece, May 2012
Proceedings

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Alfred Kobsa

University of California, Irvine, CA, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

TU Dortmund University, Germany

Madhu Sudan

Microsoft Research, Cambridge, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max Planck Institute for Informatics, Saarbruecken, Germany

Elena Simperl Philipp Cimiano
Axel Polleres Oscar Corcho
Valentina Presutti (Eds.)

The Semantic Web: Research and Applications

9th Extended Semantic Web Conference, ESWC 2012
Heraklion, Crete, Greece, May 27-31, 2012
Proceedings

Volume Editors

Elena Simperl
Karlsruhe Institute of Technology, Institute AIFB
Englerstrasse 11, 76131 Karlsruhe, Germany
E-mail: elena.simperl@aifb.uni-karlsruhe.de

Philipp Cimiano
University of Bielefeld, CITEC
Morgenbreede 39, 33615 Bielefeld, Germany
E-mail: cimiano@cit-ec.uni-bielefeld.de

Axel Polleres
Siemens AG Österreich
Siemensstrasse 90, 1210 Vienna, Austria
E-mail: axel.polleres@siemens.com

Oscar Corcho
Technical University of Madrid
C/ Severo Ochoa, 13, 28660 Boadilla del Monte, Madrid, Spain
E-mail: ocorcho@fi.upm.es

Valentina Presutti
ISTC-CNR, STLab
Via Nomentana 56, 00161 Rome, Italy
E-mail: valentina.presutti@istc.cnr.it

ISSN 0302-9743
ISBN 978-3-642-30283-1
DOI 10.1007/978-3-642-30284-8
Springer Heidelberg Dordrecht London New York

e-ISSN 1611-3349
e-ISBN 978-3-642-30284-8

Library of Congress Control Number: 2012937453

CR Subject Classification (1998): H.4, H.3.3, H.5, J.4, I.2.4, C.2

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

© Springer-Verlag Berlin Heidelberg 2012

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

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

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 Extended Semantic Web Conference (ESWC) is a major venue for discussing the latest scientific results and technology innovations around semantic technologies. Building on its past success, ESWC seeks to broaden its focus to span other relevant research areas in which Web semantics plays an important role.

The goal of the Semantic Web is to create a Web of knowledge and services in which the semantics of content is made explicit and content is linked both to other content and services, allowing novel applications to combine content from heterogeneous sites in unforeseen ways and support enhanced matching between users' needs and content. These complementarities are reflected in the outline of the technical program of ESWC 2012; in addition to the research and in-use tracks, we featured two special tracks putting particular emphasis on inter-disciplinary research topics and areas that show the potential of exciting synergies for the future, eGovernment, and digital libraries. ESWC 2012 presented the latest results in research, technologies, and applications in its field. Besides the technical program organized over multiple tracks, the conference featured several exciting, high-profile keynotes in areas directly related or adjacent to semantic technologies; a workshop and tutorial program; system descriptions and demos; a poster exhibition; a project networking session; a doctoral symposium, as well as the ESWC summer school, which was held immediately prior to the conference. As prominent co-located events we were happy to welcome the OWL Experiences and Development workshop (OWLED), as well as the AI Challenge.

The technical program of the conference received 212 submissions, which were reviewed by the Program Committee of the respective tracks; each was coordinated by Track Chairs who oversaw dedicated Program Committees. The review process included paper bidding, assessment by at least three Program Committee members, paper rebuttal, and meta-reviewing for each submission subject to acceptance in the conference program and proceedings. In all, 53 papers were selected as a result of this process, following uniform evaluation criteria devised for all (each) technical track(s).

The PhD symposium received 18 submissions, which were reviewed by the PhD Symposium Program Committee. Thirteen papers were selected for presentation at a separate track and for inclusion in the ESWC 2012 proceedings.

ESWC 2012 had the pleasure and honor to welcome seven renowned keynote speakers from academia and industry, addressing a variety of exciting topics of highest relevance for the research agenda of the semantic technologies community and its impact on ICT:

- Abraham Bernstein, Full Professor of Informatics at the University of Zurich
- Jeroen van Grondelle, cofounder of Be Informed and one of the creators of its business process platform
- Alon Halevy, head of the Structured Data Management Research Group at Google
- Alek Kolcz, software engineer at Twitter
- Monica Lam, professor in the Computer Science Department at Stanford University
- Márta Nagy-Rothengass, head of the European Commission Unit Technologies for Information Management
- Julius van de Laar, a political strategist and communications expert

We would like to take the opportunity to express our gratitude to the Chairs, Program Committee members, and additional reviewers of all refereed tracks, who ensured that this year's conference maintained the highest standards of scientific quality. Our thanks are also offered to the Organizing Committee of the conference, for their dedication and hard work in selecting and coordinating a wide array of interesting workshops, tutorials, posters, and panels that completed the program of the conference. Special thanks go to the various organizations who kindly supported our conference as sponsors, to the Sponsorship Chair who coordinated these activities, and to the team around STI International who provided an excellent service in all administrative and logistic issues related to the organization of the event. Last, but not least, we should like to say thank you to the Proceedings Chair, to the development team of the Easychair conference management system and to our publisher, Springer, for their support in the preparation of this volume and the publication of the proceedings.

May 2012

Elena Simperl
Philipp Cimiano
Axel Polleres
Oscar Corcho
Valentina Presutti

Organization

Organizing Committee

General Chair

Elena Simperl Karlsruhe Institute of Technology, Germany

Program Chairs

Philipp Cimiano University of Bielefeld, Germany
Axel Polleres Siemens AG Österreich, Vienna, Austria

Poster and Demo Chairs

Barry Norton Queen Mary University of London, UK
Dunja Mladenic Jozef Stefan Institute, Slovenia

Workshop Chairs

Alexandre Passant DERI, Ireland
Raphaël Troncy EURECOM, France

Tutorials Chairs

Emanuele della Valle University of Aberdeen, UK
Irina Fundulaki FORTH-ICS, Greece

PhD Symposium Chairs

Oscar Corcho UPM, Spain
Valentina Presutti Institute of Cognitive Sciences and
Technologies, Italy

Semantic Technologies Coordinator

Olaf Hartig Humboldt Universität zu Berlin, Germany

Sponsorship Chair

Frank Dengler Karlsruhe Institute of Technology, Germany

Publicity Chair

Paul Groth VU University of Amsterdam, The Netherlands

Panel Chair

John Davies British Telecom, UK

Proceedings Chair

Antonis Bikakis University College London, UK

Treasurer

Alexander Wahler STI International, Austria

Local Organization and Conference Administration

STI International Austria

Program Committee

Track Chairs

Linked Open Data Track

Sören Auer Chemnitz University of Technology, Germany
Juan Sequeda University of Texas at Austin, USA

Machine Learning Track

Claudia d'Amato University of Bari, Italy
Volker Tresp Siemens AG, Germany

Mobile and Sensor Web Track

Alasdair J.G. Gray University of Manchester, UK
Kerry Taylor CSIRO ICT Centre, Australia

Natural Language Processing and Information Retrieval Track

Paul Buitelaar DERI, National University of Ireland, Ireland
Johanna Völker University of Mannheim, Germany

Ontologies Track

Chiara Ghidini FBK, Italy
Dimitris Plexousakis University of Crete and FORTH-ICS, Greece

Reasoning Track

Giovambattista Ianni Università della Calabria, Italy
Markus Kroetzsch University of Oxford, UK

Semantic Data Management Track

Claudio Gutierrez University of Chile, Chile
Andreas Harth Karlsruhe Institute of Technology, Germany

Services, Processes and Cloud Computing Track

Matthias Klusch	DFKI, Germany
Carlos Pedrinaci	KMi, The Open University, UK

Social Web and Web Science Track

Fabien Gandon	INRIA, France
Matthew Rowe	KMi, The Open University, UK

In-use and Industrial Track

Philippe Cudré-Mauroux	eXascale Infolab, University of Fribourg, Switzerland
Yves Raimond	BBC, UK

Digital Libraries and Cultural Heritage Track

Antoine Isaac	Europeana / Vrije Universiteit Amsterdam, The Netherlands
Vivien Petras	Humboldt University, Berlin, Germany

EGovernment Track

Asunción Gómez-Pérez	Universidad Politecnica de Madrid, Spain
Vassilios Peristeras	European Commission

Members (all tracks)

Fabian Abel	John Bateman
Sudhir Agarwal	Sean Bechhofer
Harith Alani	Zohra Bellahsene
Dean Allemang	Paolo Bellavista
José Luis Ambite	Bettina Berendt
Sofia Angeletou	Sonia Bergamaschi
Grigoris Antoniou	Luis Bermudez
Dimitris Apostolou	Leopoldo Bertossi
Lora Aroyo	Christian Bizer
Marie-Aude Aufaure	Stephan Bloehdorn
Nathalie Aussenac-Gilles	Eva Blomqvist
Daniel Gayo Avello	Andreas Blumauer
Reza B'Far	Uldis Bojars
Franz Baader	Piero Bonatti
Bruno Bachimont	Francesco Bonchi
Thomas Baker	Philippe Bonnet
Jie Bao	Kalina Bontcheva
Chitta Baral	Alex Borgida
Payam Barnaghi	Stefano Borgo
Roberto Basili	Johan Bos

Cecile Bothorel
Paolo Bouquet
Karin Breitman
Christopher Brewster
Dan Brickley
Boyan Brodaric
François Bry
Liliana Cabral
Andrea Cali
Donatella Castelli
Ciro Cattuto
Irene Celino
Stefano Ceri
Yannis Charalabidis
Vinay Chaudhri
Christian Chiarcos
Kendall Clark
Michael Compton
Mariano Consens
Bonaventura Coppola
Paulo Costa
Isabel Cruz
Carlo Curino
Richard Cyganiak
Mathieu D'Aquin
Souripriya Das
Frithjof Dau
Pieter De Leenheer
Gerard De Melo
David De Roure
Roberto De Virgilio
Mike Dean
Thierry Declerck
Emanuele Della Valle
Tommaso Di Noia
Stefan Dietze
Martin Doerr
Kai Eckert
Peter Edwards
Orri Erling
Federico Michele Facca
Nicola Fanizzi
Tim Finin
Michael Fink
Kai Fischbach

Giorgos Flouris
Bo Fu
Aldo Gangemi
Roberto Garcia
Jose Emilio Labra Gayo
Hugh Glaser
Birte Glimm
Alfio Gliozzo
Jose Manuel Gomez-Perez
Asunción Gómez-Pérez
Guido Governatori
Stefan Gradmann
Bernardo Cuenca Grau
Gregory Grefenstette
Stephan Grimm
Marko Grobelnik
Paul Groth
Tudor Groza
Michael Gruninger
Nicola Guarino
Christophe Guéret
Ramanathan Guha
Giancarlo Guizzardi
Jon Atle Gulla
Volker Haarslev
Peter Haase
Hakim Hacid
Udo Hahn
Harry Halpin
Tony Hammond
Siegfried Handschuh
Steve Harris, Garlik
Olaf Hartig
Alois Haselböck
Bernhard Haslhofer
Tom Heath
Ralf Heese
Sebastian Hellmann
Cory Henson
Martin Hepp
Ivan Herman
Stijn Heymans
Graeme Hirst
Pascal Hitzler
Aidan Hogan

Katja Hose
Andreas Hotho
Nicholas Humfrey
David Huynh
Eero Hyvönen
Krzysztof Janowicz
Jason Jung
Lalana Kagal
Marcel Karnstedt
C. Maria Keet
Gregg Kellogg
Kristian Kersting
Holger Kett
Jörg-Uwe Kietz
Ross King
Sheila Kinsella
Ralf Klischewski
Craig Knoblock
Pascale Kuntz
Oliver Kutz
Georg Lausen
Florian Lautenbacher
Agnieszka Lawrynowicz
Laurent Lefort
Jens Lehmann
Marcello Leida
Joao Leite
Alexander Lenk
Francesca Alessandra Lisi
Yong Liu
Vanessa Lopez
Nikos Loutas
Thomas Lukasiewicz
Carsten Lutz
Maria Machado
Pedro Jose Marron
Kirk Martinez
Christopher Matheus
Andrea Maurino
Diana Maynard
Philipp Mayr
John McCrae
Massimo Mecella
Paola Mello
Pablo N. Mendes

Jan Mendling
Gregoris Mentzas
Peter Mika
Ivan Mikhailov
Zoltan Miklos
Paolo Missier
Riichiro Mizoguchi
Dunja Mladenic
Alessandro Moschitti
Enrico Motta
Elena Mugellini
Felix Naumann
Roberto Navigli
Günter Neumann
Marco Neumann
Axel-Cyrille Ngonga Ngomo
Maximilian Nickel
Matthias Nickles
Andriy Nikolov
Malvina Nissim
Barry Norton
Natasha F. Noy
Leo Obrst
Silver Oliver
Alessandro Oltramari
Johan Oomen
Sascha Ossowski
Kevin Page
Jeff Z. Pan
Michael Panzer
Massimo Paolucci
Josiane Xavier Parreira
Alexandre Passant
Terry Payne
Jamie Payton
Rafael Peñaloza
Matthew Perry
Wim Peters
Danh Le Phuoc
H. Sofia Pinto
Pierluigi Plebani
Guilin Qi
Anand Ranganathan
Achim Rettinger
Chantal Reynaud

Myriam Ribiere
Mark Roantree
Andrea Rodriguez
Mariano Rodrigues-Muro
Kurt Rohloff
Dumitru Roman
Riccardo Rosati
Marco Rospocher
Edna Ruckhaus
Sebastian Rudolph
Stefanie Rühle
Marta Sabou
Harald Sack
Evan Sandhaus
Felix Sasaki
Kai-Uwe Sattler
Ulrike Sattler
Sebastian Schaffert
Bernhard Schandl
Thomas Scharrenbach
Stefan Schlobach
Jodi Schneider
Michael Schneider
Stefan Schulte
Daniel Schwabe
Tom Scott
Giovanni Semeraro
Juan F. Sequeda
Luciano Serafini
Barş Sertkaya
Ryan Shaw
Amit Sheth
Michael Sintek
Katharina Siorpaes
Sergej Sizov
Steffen Staab
Johann Stan
Milan Stankovic
Thomas Steiner
Florian Steinke
Robert Stevens
Adrian Stevenson
Giorgos Stoilos
Nenad Stojanovic
Michael Stollberg
Henry Story

Heiner Stuckenschmidt
Fabian M. Suchanek
York Sure
Vojtech Svatek
Lars Svensson
Efthimios Tambouris
Valentina Tamma
Sergio Tessaris
Bryan Thompson
Andreas Thor
Ioan Toma
Alessandra Toninelli
Sebastian Tramp
Thanh Tran
Francky Trichet
Raphael Troncy
Douglas Tudhope
Tania Tudorache
Mischa Tuffield
Giovanni Tummarello
Vassilis Tzouvaras
Jacopo Urbani
Alejandro A. Vaisman
Jacco van Ossenbruggen
Paola Velardi
Maria Esther Vidal
Boris Villazón-Terrazas
Piek Vossen
George Vouros
Denny Vrandecic
Holger Wache
Haofen Wang
Kewen Wang
Shenghui Wang
Zhe Wang
Albert Weichselbraun
Maria Wimmer
Peter Wood
Eiko Yoneki
Marcia Zeng
Ying Zhang
Jun Zhao
Antoine Zimmermann
Ingo Zinnikus

Referees

Rabeeh Abbasi
 Alessandro Adamou
 Panos Alexopoulos
 Stefano Braglia
 Janez Bran,
 Lorenz Bühmann
 Federico Chesani
 Lorand Dali
 Fabien Duchateau
 Lorena Etcheverry
 Miriam Fernandez
 Mouzhi Ge
 Michael Hartung
 Robert Isele
 Prateek Jain
 Amit Joshi
 Evangelos Kalampokis
 Matthias Knorr
 Thomas Kurz
 Günter Ladwig
 Christoph Lange
 Nadine Ludwig
 Uta Lösch
 Yue Ma
 Akshay Maan
 Brigitte Mathiak

Ivan Mikhailov
 Pasquale Minervini
 Raghava Mutharaju
 Ralf Möller
 Inna Novalija
 Andrea Giovanni Nuzzolese
 Chimezie Ogbuji
 Fabrizio Orlandi
 Matteo Palmonari
 Ajit Ranabahu
 Laurens Rietveld
 Benedicto Rodriguez-Castro
 Jürgen Rühle
 Luigi Sauro
 Michael Schmidt
 Murat Sensoy
 Inanc Seylan
 Stian Soiland-Reyes
 Seema Sundara
 Ondrej Svab-Zamazal
 Samson Tu
 Yannis Tzitzikas
 Joerg Waitelonis
 Liang Yu
 Benjamin Zapilko

Steering Committee**Chair**

John Domingue

Members

Grigoris Antoniou
 Lora Aroyo
 Sean Bechhofer
 Fabio Ciravegna

Marko Grobelink
 Eero Hyvönen
 Paolo Traverso

Sponsoring Institutions



Table of Contents

Invited Talks

Semantic Web/LD at a Crossroads: Into the Garbage Can or To Theory? (Abstract)	1
<i>Abraham Bernstein</i>	
New Audiences for Ontologies: Dealing with Complexity in Business Processes (Abstract)	2
<i>Jeroen van Grondelle</i>	
Bringing (Web) Databases to the Masses (Abstract)	3
<i>Alon Halevy</i>	
Large Scale Learning at Twitter (Abstract)	4
<i>Aleksander Kotcz</i>	
Musubi: A Decentralized Mobile Social Web (Abstract)	5
<i>Monica S. Lam</i>	
Data Value Chain in Europe (Abstract)	6
<i>Márta Nagy-Rothengass</i>	
Cutting through the Noise: How to Shape Public Perception, Frame the Debate and Effectively Engage Your Audience in the Digital Age (Abstract)	7
<i>Julius van de Laar</i>	

Linked Open Data Track

SPARQL for a Web of Linked Data: Semantics and Computability	8
<i>Olaf Hartig</i>	
Linked Data-Based Concept Recommendation: Comparison of Different Methods in Open Innovation Scenario	24
<i>Danica Damljanovic, Milan Stankovic, and Philippe Laublet</i>	
Finding Co-solvers on Twitter, with a Little Help from Linked Data	39
<i>Milan Stankovic, Matthew Rowe, and Philippe Laublet</i>	
Top- <i>k</i> Linked Data Query Processing	56
<i>Andreas Wagner, Thanh Tran Duc, Günter Ladwig, Andreas Harth, and Rudi Studer</i>	

Preserving Information Content in RDF Using Bounded Homomorphisms	72
<i>Audun Stolpe and Martin G. Skjæveland</i>	
Assessing Linked Data Mappings Using Network Measures	87
<i>Christophe Guéret, Paul Groth, Claus Stadler, and Jens Lehmann</i>	
A Novel Concept-Based Search for the Web of Data Using UMBEL and a Fuzzy Retrieval Model	103
<i>Melike Sah and Vincent Wade</i>	

Machine Learning Track

Unsupervised Learning of Link Discovery Configuration	119
<i>Andriy Nikolov, Mathieu d’Aquin, and Enrico Motta</i>	
Graph Kernels for RDF Data	134
<i>Uta Lösch, Stephan Bloehdorn, and Achim Rettinger</i>	
EAGLE: Efficient Active Learning of Link Specifications Using Genetic Programming	149
<i>Axel-Cyrille Ngonga Ngomo and Klaus Lyko</i>	
Combining Information Extraction, Deductive Reasoning and Machine Learning for Relation Prediction	164
<i>Xueyan Jiang, Yi Huang, Maximilian Nickel, and Volker Tresp</i>	
Automatic Configuration Selection Using Ontology Matching Task Profiling	179
<i>Isabel F. Cruz, Alessio Fabiani, Federico Caimi, Cosmin Stroe, and Matteo Palmonari</i>	

Natural Language Processing and Information Retrieval Track

TELEX: An RDF-Based Model for Linguistic Annotation	195
<i>Emilio Rubiera, Luis Polo, Diego Berrueta, and Adil El Ghali</i>	
LODifier: Generating Linked Data from Unstructured Text	210
<i>Isabelle Augenstein, Sebastian Padó, and Sebastian Rudolph</i>	
POWLA: Modeling Linguistic Corpora in OWL/DL	225
<i>Christian Chiarcos</i>	

Ontologies Track

Representing Mereotopological Relations in OWL Ontologies with ONTOPARTS	240
<i>C. Maria Keet, Francis C. Fernández-Reyes, and Annette Morales-González</i>	
Evaluation of the Music Ontology Framework	255
<i>Yves Raimond and Mark Sandler</i>	
The Current State of SKOS Vocabularies on the Web.....	270
<i>Nor Azlinayati Abdul Manaf, Sean Bechhofer, and Robert Stevens</i>	
The ISOcat Registry Reloaded.....	285
<i>Claus Zinn, Christina Hoppermann, and Thorsten Trippel</i>	
SCHEMA - An Algorithm for Automated Product Taxonomy Mapping in E-commerce	300
<i>Steven S. Aanen, Lennart J. Niderstigt, Damir Vandić, and Flavius Fräsincar</i>	
Voting Theory for Concept Detection	315
<i>Amal Zouaq, Dragan Gasevic, and Marek Hatala</i>	

Reasoning Track

Modelling Structured Domains Using Description Graphs and Logic Programming.....	330
<i>Despoina Magka, Boris Motik, and Ian Horrocks</i>	
Extending Description Logic Rules	345
<i>David Carral Martínez and Pascal Hitzler</i>	
Prexto: Query Rewriting under Extensional Constraints in <i>DL-Lite</i>	360
<i>Riccardo Rosati</i>	

Semantic Data Management Track

Semi-automatically Mapping Structured Sources into the Semantic Web.....	375
<i>Craig A. Knoblock, Pedro Szekely, José Luis Ambite, Aman Goel, Shubham Gupta, Kristina Lerman, Maria Muslea, Mohsen Taheriyani, and Parag Mallick</i>	
Castor: A Constraint-Based SPARQL Engine with Active Filter Processing	391
<i>Vianney le Clément de Saint-Marcq, Yves Deville, Christine Solnon, and Pierre-Antoine Champin</i>	

A Structural Approach to Indexing Triples	406
<i>François Picalausa, Yongming Luo, George H.L. Fletcher, Jan Hidders, and Stijn Vansummeren</i>	
Domain Specific Data Retrieval on the Semantic Web	422
<i>Tuukka Ruotsalo</i>	
Exchange and Consumption of Huge RDF Data	437
<i>Miguel A. Martínez-Prieto, Mario Arias Gallego, and Javier D. Fernández</i>	
Impact of Using Relationships between Ontologies to Enhance the Ontology Search Results	453
<i>Carlo Allocca, Mathieu d’Aquin, and Enrico Motta</i>	
Enhancing OLAP Analysis with Web Cubes	469
<i>Lorena Etcheverry and Alejandro A. Vaisman</i>	
Query-Independent Learning to Rank for RDF Entity Search	484
<i>Lorand Dali, Blaž Fortuna, Thanh Tran Duc, and Dunja Mladenić</i>	

Services, Processes and Cloud Computing Track

COV4SWS.KOM: Information Quality-Aware Matchmaking for Semantic Services	499
<i>Stefan Schulte, Ulrich Lampe, Matthias Klusch, and Ralf Steinmetz</i>	

Social Web and Web Science Track

Automatic Identification of Best Answers in Online Enquiry Communities	514
<i>Grégoire Burel, Yulan He, and Harith Alani</i>	
Characterising Emergent Semantics in Twitter Lists	530
<i>Andrés García-Silva, Jeon-Hyung Kang, Kristina Lerman, and Oscar Corcho</i>	
Crowdsourcing Taxonomies	545
<i>Dimitris Karampinas and Peter Triantafillou</i>	

In-use and Industrial Track

Generating Possible Interpretations for Statistics from Linked Open Data	560
<i>Heiko Paulheim</i>	
Green-Thumb Camera: LOD Application for Field IT	575
<i>Takahiro Kawamura and Akihiko Ohsuga</i>	

Assembling Rule Mashups in the Semantic Web	590
<i>Guillermo González-Moriyón, Luis Polo, Diego Berrueta, Carlos Tejo-Alonso, and Miguel Iglesias</i>	
Product Customization as Linked Data	603
<i>Edouard Chevalier and François-Paul Servant</i>	
From Web 1.0 to Social Semantic Web: Lessons Learnt from a Migration to a Medical Semantic Wiki	618
<i>Thomas Meilender, Jean Lieber, Fabien Palomares, and Nicolas Jay</i>	
Semantics Visualization for Fostering Search Result Comprehension	633
<i>Christian Stab, Kawa Nazemi, Matthias Breyer, Dirk Burkhardt, and Jörn Kohlhammer</i>	
Evaluating Scientific Hypotheses Using the SPARQL Inferencing Notation	647
<i>Alison Callahan and Michel Dumontier</i>	
Declarative Representation of Programming Access to Ontologies	659
<i>Stefan Scheglmann, Ansgar Scherp, and Steffen Staab</i>	
Clinical Trial and Disease Search with Ad Hoc Interactive Ontology Alignments	674
<i>Daniel Sonntag, Jochen Setz, Maha Ahmed-Baker, and Sonja Zillner</i>	
Towards Fuzzy Query-Relaxation for RDF	687
<i>Aidan Hogan, Marc Mellotte, Gavin Powell, and Dafni Stampouli</i>	
Learning Driver Preferences of POIs Using a Semantic Web Knowledge System	703
<i>Rahul Parundekar and Kentaro Oguchi</i>	

Digital Libraries and Cultural Heritage Track

An Approach for Named Entity Recognition in Poorly Structured Data	718
<i>Nuno Freire, José Borbinha, and Pável Calado</i>	
Supporting Linked Data Production for Cultural Heritage Institutes: The Amsterdam Museum Case Study	733
<i>Victor de Boer, Jan Wielemaker, Judith van Gent, Michiel Hildebrand, Antoine Isaac, Jacco van Ossenbruggen, and Gaus Schreiber</i>	
Curate and Storyspace: An Ontology and Web-Based Environment for Describing Curatorial Narratives	748
<i>Paul Mulholland, Annika Wolff, and Trevor Collins</i>	

Bringing Mathematics to the Web of Data: The Case of the Mathematics Subject Classification 763
Christoph Lange, Patrick Ion, Anastasia Dimou, Charalampos Bratsas, Wolfram Sperber, Michael Kohlhase, and Ioannis Antoniou

EGovernment Track

A Publishing Pipeline for Linked Government Data 778
Fadi Maali, Richard Cyganiak, and Vassilios Peristeras

Achieving Interoperability through Semantic Technologies in the Public Administration 793
Chiara Di Francescomarino, Mauro Dragoni, Matteo Gerosa, Chiara Ghidini, Marco Rospocher, and Michele Trainotti

PhD Symposium

Tackling Incompleteness in Information Extraction – A Complementarity Approach 808
Christina Feilmayr

A Framework for Ontology Usage Analysis 813
Jamshaid Ashraf

Formal Specification of Ontology Networks 818
Edelweis Rohrer

Leveraging Linked Data Analysis for Semantic Recommender Systems 823
Andreas Thalhammer

Sharing Statistics for SPARQL Federation Optimization, with Emphasis on Benchmark Quality 828
Kjetil Kjernsmo

A Reuse-Based Lightweight Method for Developing Linked Data Ontologies and Vocabularies 833
María Poveda-Villalón

Optimising XML–RDF Data Integration: A Formal Approach to Improve XSPARQL Efficiency 838
Stefan Bischof

Software Architectures for Scalable Ontology Networks 844
Alessandro Adamou

Identifying Complex Semantic Matches 849
Brian Walshe

Data Linking with Ontology Alignment	854
<i>Zhengjie Fan</i>	
A Semantic Policy Sharing Infrastructure for Pervasive Communities . . .	859
<i>Vikash Kumar</i>	
Involving Domain Experts in Ontology Construction: A Template Based Approach	864
<i>Muhammad Tahir Khan</i>	
Quality Assurance in Collaboratively Created Web Vocabularies	870
<i>Christian Mader</i>	
Author Index	875

Semantic Web/LD at a Crossroads: Into the Garbage Can or To Theory?

Abraham Bernstein

University of Zurich
bernstein@ifi.uzh.ch

From Greek mythology (abbreviated from the Wikipedia):

Scylla and Charybdis were mythical sea monsters noted by Homer. Scylla was rationalized as a rock shoal (described as a six-headed sea monster) and Charybdis was a whirlpool. They were regarded as a sea hazard located close enough to each other that they posed an inescapable threat to passing sailors; avoiding Charybdis meant passing too close to Scylla and vice versa. According to Homer, Odysseus was forced to choose which monster to confront while passing through the strait...

Since its inception Semantic Web research projects have tried to sail the strait between the Scylla of overly theoretical irrelevance and the Charybdis of non-scientific applied projects.

Like Odysseus the Semantic Web community was wooed by the neatness of theoretical explorations of knowledge representation methods that endanger to crash the community into the Scylla the rock of irrelevance.

On the other side the maelstrom of Charybdis attracts the community as it tries to fulfill the next vision of the next web thereby losing its scientific identity.

In this talk I will discuss and exemplify the strengths, weaknesses, opportunities, and pitfalls (or threats) of each of these extremes. I will use this analysis as a basis for to explore some possible strategies to navigate the potentially stormy seas of the Semantic Web community's future.

New Audiences for Ontologies: Dealing with Complexity in Business Processes

Jeroen van Grondelle

Be Informed, The Netherlands
j.vangrondelle@beinformed.com
<http://www.beinformed.com/>

Today, governments and businesses have to deal with high degrees of complexity: The products they offer are highly individualized, there are many regulations they must comply with and all this has to be dealt with under a growing rate of change. Many organizations have tried to meet this challenge by reducing complexity, through the elimination of exceptions etc. Jeroen van Grondelle, principal architect at Be Informed, argues that the only way to deal with complexity is by embracing it.

Ontologies have proven to be an excellent way to deal with all the different concepts that are introduced when products are defined and the supporting business processes are designed. When the right conceptualization is chosen, ontologies capture these policy choices and constraints in a natural way. Ontologies deal well with the heterogeneous nature of policy and regulations, which often originate from different legal sources and have different owners.

The benefits exist throughout the entire policy lifecycle. The formal, precise nature of ontologies improves the quality and consistency of the choices made and reduces ambiguity. Because ontologies are well interpretable by machines, Be Informed succeeds in inferring many of the supporting services, such as process applications and decision services, from the ontologies, thereby eliminating the need for systems development. And the ability to infer executable services also allows for advanced what-if analysis and simulation of candidate policies before they are put into effect.

Jeroen will show some examples where ontologies were successfully applied in the public sector and what the impact was on all parties involved, from policy officers to citizens. He will also present some of the research challenges encountered when these new audiences are confronted with ontologies, a technology that typically is of course unfamiliar to them.

Bringing (Web) Databases to the Masses

Alon Halevy

Google Inc., USA
halevy@google.com

The World-Wide Web contains vast quantities of structured data on a variety of domains, such as hobbies, products and reference data. Moreover, the Web provides a platform that can encourage publishing more data sets from governments and other public organizations and support new data management opportunities, such as effective crisis response, data journalism and crowd-sourcing data sets. To enable such wide-spread dissemination and use of structured data on the Web, we need to create an ecosystem that makes it easier for users to discover, manage, visualize and publish structured data on the Web.

I will describe some of the efforts we are conducting at Google towards this goal and the technical challenges they raise. In particular, I will describe Google Fusion Tables, a service that makes it easy for users to contribute data and visualizations to the Web, and the WebTables Project that attempts to discover high-quality tables on the Web and provide effective search over the resulting collection of 200 million tables.

Large Scale Learning at Twitter

Aleksander Kolcz

Twitter, USA
alek@ir.iit.edu

Twitter represents a large complex network of users with diverse and continuously evolving interests. Discussions and interactions range from very small to very large groups of people and most of them occur in the public. Interests are both long and short term and are expressed by the content generated by the users as well as via the Twitter follow graph, i.e. who is following whose content.

Understanding user interests is crucial to providing good Twitter experience by helping users to connect to others, find relevant information and interesting information sources.

The manner in which information is spread over the network and communication attempts are made can also help in identifying spammers and other service abuses.

Understanding users and their preferences is also a very challenging problem due to the very large volume information, the fast rate of change and the short nature of the tweets. Large scale machine learning as well as graph and text mining have been helping us to tackle these problems and create new opportunities to better understand our users. In the talk I will describe a number of challenging modeling problems addressed by the Twitter team as well as our approach to creating frameworks and infrastructure to make learning at scale possible.

Musubi: A Decentralized Mobile Social Web

Monica S. Lam

Stanford University, USA
lam@cs.stanford.edu

With the rise of cloud services, users' personal data, from photos to bank transactions, are scattered and hosted by a variety of application service providers. Communication services like email and social networking, by virtue of helping users share, have the unique opportunity to gather all data shared in one place. As users shift their communication medium from email to social networks, personal data are increasingly locked up in a global, proprietary social web.

We see the rise of the mobile phone as an opportunity to re-establish an open standard, as social data are often produced, shared, and consumed on the mobile devices directly. We propose an API where apps can interact with friends' phones directly, without intermediation through a centralized communication service. Furthermore, this information can then be made available on our own devices to personalize and improve online interactions. Based on this API, we have created a working prototype called Musubi (short for Mobile, Social, and UBIquitous) along with various social applications, all of which are available on the Android market.

Data Value Chain in Europe

Márta Nagy-Rothengass

European Commission
Marta.Nagy-Rothengass@ec.europa.eu

Data is today everywhere. The quantity and growth of generated data is enormous, its proper management challenges us individual users but also business and public organisations. We are facing data/information overflow and we are often handicapped by storing, managing, analysing and preserving all of our data.

At the same time, this growing large amount of data offers us due to its intelligent linkage, analysis and processing

- business opportunities like establishment of new, innovative digital services towards end users and organisations;
- better decision making support in business and public sector; and
- increased intelligence and better knowledge extraction.

This is the reason why many of us see data as the oil of the 21th century. We are challenged to unlock the potential and the added value of complex and big data. It has become a competitive advantage to offer the right data to the right people at the right time.

In my talk I will introduce the value chain thinking on data, than analyse its main technology and business challenges and inform about the ongoing and envisaged policy, infrastructure, research and innovation activities at European level.

Cutting through the Noise: How to Shape Public Perception, Frame the Debate and Effectively Engage Your Audience in the Digital Age

Julius van de Laar

Strategic Campaigns & Communication, Germany
julius@juliusvandelaaar.com

Successful political campaigns have mastered the tactics and strategies used to effectively present an argument, manage and respond with authority during crisis, influence the debate and shape public perception.

Yet, in today's 24/7 media environment it has become more difficult than ever to set an agenda, frame an issue or engage an audience.

Four years ago, Barack Obama set a new standard for campaigning by changing the way new media was used to build an aspirational brand, engage and empower supporters, raise money and turn out voters. As the 2012 presidential race unfolds, the campaigns are stepping up their game. And in this cycle, they are embracing digital media more than ever.

However, it's not only the President's campaign and his opponents who are faced with the challenge to create a narrative and frame the public debate. Organizations in the private sector often deal with the similar complex issues as they struggle to deliver tailored messages to target their audience, regardless of whether it's customers, investors, media, the general public or even potential employees.

From storytelling to big data lifestyle targeting: Julius van de Laar will provide a first hand account on how today's most effective campaigns leverage battle tested strategies combined with new media tools to create a persuasive narrative and how they translate into actionable strategies for the corporate context.

SPARQL for a Web of Linked Data: Semantics and Computability

Olaf Hartig

Humboldt-Universität zu Berlin
hartig@informatik.hu-berlin.de

Abstract. The World Wide Web currently evolves into a Web of Linked Data where content providers publish and link data as they have done with hypertext for the last 20 years. While the declarative query language SPARQL is the de facto for querying a-priory defined sets of data from the Web, no language exists for querying the Web of Linked Data itself. However, it seems natural to ask whether SPARQL is also suitable for such a purpose.

In this paper we formally investigate the applicability of SPARQL as a query language for Linked Data on the Web. In particular, we study two query models: 1) a *full-Web semantics* where the scope of a query is the complete set of Linked Data on the Web and 2) a family of *reachability-based semantics* which restrict the scope to data that is reachable by traversing certain data links. For both models we discuss properties such as monotonicity and computability as well as the implications of querying a Web that is infinitely large due to data generating servers.

1 Introduction

The emergence of vast amounts of RDF data on the WWW has spawned research on storing and querying large collections of such data efficiently. The prevalent query language in this context is SPARQL [16] which defines queries as functions over an RDF dataset, that is, a fixed, a-priory defined collection of sets of RDF triples. This definition naturally fits the use case of querying a repository of RDF data copied from the Web.

However, most RDF data on the Web is published following the Linked Data principles [5], contributing to the emerging Web of Linked Data [6]. This practice allows for query approaches that access the most recent version of remote data on demand. More importantly, query execution systems may automatically discover new data by traversing data links. As a result, such a system answers queries based on data that is not only up-to-date but may also include initially unknown data. These features are the foundation for true serendipity, which we regard as the most distinguishing advantage of querying the Web itself, instead of a predefined, bounded collection of data.

While several research groups work on systems that evaluate SPARQL basic graph patterns over the Web of Linked Data (cf. [9], [10,12] and [13,14]), we notice a shortage of work on theoretical foundations and properties of such queries. Furthermore, there is a need to support queries that are more expressive than conjunctive (basic graph pattern based) queries [17]. However, it seems natural to assume that SPARQL could be used in this context because the Web of Linked Data is based on the RDF data model and SPARQL is a query language for RDF data. In this paper we challenge this assumption.

Contributions. In this paper we understand queries as functions over the Web of Linked Data as a whole. To analyze the suitability of SPARQL as a language for such queries, we have to adjust the semantics of SPARQL. More precisely, we have to redefine the scope for evaluating SPARQL algebra expressions. In this paper we discuss two approaches for such an adjustment. The first approach uses a semantics where the scope of a query is the complete set of Linked Data on the Web. We call this semantics *full-Web semantics*. The second approach introduces a family of *reachability-based semantics* which restrict the scope to data that is reachable by traversing certain data links. We emphasize that both approaches allow for query results that are based on data from initially unknown sources and, thus, enable applications to tap the full potential of the Web. Nevertheless, both approaches precisely define the (expected) result for any query.

As a prerequisite for defining the aforementioned semantics and for studying theoretical properties of queries under these semantics, we introduce a theoretical framework. The basis of this framework is a data model that captures the idea of a Web of Linked Data. We model such a Web as an infinite structure of documents that contain RDF data and that are interlinked via this data. Our model allows for infiniteness because the number of entities described in a Web of Linked Data may be infinite; so may the number of documents. The following example illustrates such a case:

Example 1. Let u_i denote an HTTP scheme based URI that identifies the natural number i . There is a countably infinite number of such URIs. The WWW server which is responsible for these URIs may be set up to provide a document for each natural number. These documents may be generated upon request and may contain RDF data including the RDF triple $(u_i, \text{http://.../next}, u_{i+1})$. This triple associates the natural number i with its successor $i+1$ and, thus, links to the data about $i+1$ [19]. An example for such a server is provided by the Linked Open Numbers project¹.

In addition to the data model our theoretical framework comprises a computation model. This model is based on a particular type of Turing machine which formally captures the limited data access capabilities of computations over the Web.

We summarize the main contributions of this paper as follows:

- We present a data model and a computation model that provide a theoretical framework to define and to study query languages for the Web of Linked Data.
- We introduce a full-Web semantics and a family of reachability-based semantics for a (hypothetical) use of SPARQL as a language for queries over Linked Data.
- We systematically analyze SPARQL queries under the semantics that we introduce. This analysis includes a discussion of satisfiability, monotonicity, and computability of queries under the different semantics, a comparison of the semantics, and a study of the implications of querying a Web of Linked Data that is infinite.

Related Work. Since its emergence the WWW has attracted research on declarative query languages for the Web. For an overview on early work in this area we refer to [8]. Most of this work understands the WWW as a hypertext Web. Nonetheless, some of the foundational work can be adopted for research on Linked Data. The computation model that we use in this paper is an adaptation of the ideas presented in [11] and [15].

¹ <http://km.aifb.kit.edu/projects/numbers/>

In addition to the early work on Web queries, query execution over Linked Data on the WWW has attracted much attention recently [9][10][12][13][14]. However, existing work primarily focuses on various aspects of (query-local) data management, query execution, and optimization. The only work we are aware of that aims to formally capture the concept of Linked Data and to provide a well-defined semantics for queries in this context is Bouquet et al.’s [7]. They define three types of query methods for conjunctive queries: a bounded method which only uses RDF data referred to in queries, a direct access method which assumes an oracle that provides all RDF graphs which are “relevant” for a given query, and a navigational method which corresponds to a particular reachability-based semantics. For the latter Bouquet et al. define a notion of reachability that allows a query execution system to follow *all* data links. As a consequence, the semantics of queries using this navigational method is equivalent to, what we call, c_{All} -semantics (cf. Section 5.1); it is the most general of our reachability-based semantics. Bouquet et al.’s navigational query model does not support other, more restrictive notions of reachability, as is possible with our model. Furthermore, Bouquet et al. do not discuss full SPARQL, theoretical properties of queries, or the infiniteness of the WWW.

While we focus on the query language SPARQL in the context of Linked Data on the Web, the theoretical properties of SPARQL as a query language for a fixed, predefined collection of RDF data are well understood today [2][3][16][18]. Particularly interesting in our context are semantical equivalences between SPARQL expressions [18] because these equivalences may also be used for optimizing SPARQL queries over Linked Data.

Structure of the Paper. The remainder of this paper is organized as follows. Section 2 introduces the preliminaries for our work. In Section 3 we present the data model and the computation model. Sections 4 and 5 discuss the full-Web semantics and the reachability-based semantics for SPARQL, respectively. We conclude the paper in Section 6. For full technical proofs of all results in this paper we refer to [11].

2 Preliminaries

This section provides a brief introduction of RDF and the query language SPARQL.

We assume pairwise disjoint, countably infinite sets \mathcal{U} (all HTTP scheme based URIs²), \mathcal{B} (blank nodes), \mathcal{L} (literals), and \mathcal{V} (variables, denoted by a leading ‘?’ symbol). An RDF triple t is a tuple $(s, p, o) \in (\mathcal{U} \cup \mathcal{B}) \times \mathcal{U} \times (\mathcal{U} \cup \mathcal{B} \cup \mathcal{L})$. For any RDF triple $t = (s, p, o)$ we define $\text{terms}(t) = \{s, p, o\}$ and $\text{uris}(t) = \text{terms}(t) \cap \mathcal{U}$. Overloading function terms, we write $\text{terms}(G) = \bigcup_{t \in G} \text{terms}(t)$ for any (potentially infinite) set G of RDF triples. In contrast to the usual formalization of RDF we allow for infinite sets of RDF triples which we require to study infinite Webs of Linked Data.

In this paper we focus on the core fragment of SPARQL discussed by Pérez et al. [16] and we adopt their formalization approach, that is, we use the algebraic syntax and the compositional set semantics introduced in [16]. *SPARQL expressions* are defined recursively: i) A *triple pattern* $(s, p, o) \in (\mathcal{V} \cup \mathcal{U}) \times (\mathcal{V} \cup \mathcal{U}) \times (\mathcal{V} \cup \mathcal{U} \cup \mathcal{L})$ is a

² For the sake of simplicity we assume in this paper that URIs are HTTP scheme based URIs. However, our models and result may be extended easily for all possible types of URIs.

SPARQL expression P_1 ii) If P_1 and P_2 are SPARQL expressions, then $(P_1 \text{ AND } P_2)$, $(P_1 \text{ UNION } P_2)$, $(P_1 \text{ OPT } P_2)$, and $(P_1 \text{ FILTER } R)$ are SPARQL expressions where R is a filter condition. For a formal definition of filter conditions we refer to [16]. To denote the set of all variables in all triple patterns of a SPARQL expression P we write $\text{vars}(P)$.

To define the semantics of SPARQL we introduce *valuations*, that are, partial mappings $\mu : \mathcal{V} \rightarrow \mathcal{U} \cup \mathcal{B} \cup \mathcal{L}$. The *evaluation* of a SPARQL expression P over a potentially infinite set G of RDF triples, denoted by $\llbracket P \rrbracket_G$, is a set of valuations. In contrast to the usual case, this set may be infinite in our scenario. The evaluation function $\llbracket \cdot \rrbracket$ is defined recursively over the structure of SPARQL expressions. Due to space limitations, we do not reproduce the full formal definition of $\llbracket \cdot \rrbracket$ here. Instead, we refer the reader to the definitions given by Pérez et al. [16]; even if Pérez et al. define $\llbracket \cdot \rrbracket$ for finite sets of RDF triples, it is trivial to extend their formalism for infiniteness (cf. appendix in [11]).

A SPARQL expression P is *monotonic* if for any pair G_1, G_2 of (potentially infinite) sets of RDF triples such that $G_1 \subseteq G_2$, it holds that $\llbracket P \rrbracket_{G_1} \subseteq \llbracket P \rrbracket_{G_2}$. A SPARQL expression P is *satisfiable* if there exists a (potentially infinite) set G of RDF triples such that $\llbracket P \rrbracket_G \neq \emptyset$. It is trivial to show that any non-satisfiable expression is monotonic.

In addition to the traditional notion of satisfiability we shall need a more restrictive notion for the discussion in this paper: A SPARQL expression P is *nontrivially satisfiable* if there exists a (potentially infinite) set G of RDF triples and a valuation μ such that i) $\mu \in \llbracket P \rrbracket_G$ and ii) μ provides a binding for at least one variable; i.e. $\text{dom}(\mu) \neq \emptyset$.

Example 2. Let $P_{\text{Ex}2} = tp$ be a SPARQL expression that consists of a single triple pattern $tp = (u_1, u_2, u_3)$ where $u_1, u_2, u_3 \in \mathcal{U}$; hence, tp actually is an RDF triple. For any set G of RDF triples for which $(u_1, u_2, u_3) \in G$ it is easy to see that the evaluation of $P_{\text{Ex}2}$ over G contains a single, empty valuation μ_\emptyset , that is, $\llbracket P_{\text{Ex}2} \rrbracket_G = \{\mu_\emptyset\}$ where $\text{dom}(\mu_\emptyset) = \emptyset$. In contrast, for any other set G of RDF triples it holds $\llbracket P_{\text{Ex}2} \rrbracket_G = \emptyset$. Hence, $P_{\text{Ex}2}$ is *not* nontrivially satisfiable (although it is satisfiable).

3 Modeling a Web of Linked Data

In this section we introduce theoretical foundations which shall allow us to define and to analyze query models for Linked Data. In particular, we propose a data model and introduce a computation model. For these models we assume a *static view* of the Web; that is, no changes are made to the data on the Web during the execution of a query.

3.1 Data Model

We model the Web of Linked Data as a potentially infinite structure of interlinked documents. Such documents, which we call Linked Data documents, or *LD documents* for short, are accessed via URIs and contain data that is represented as a set of RDF triples.

Definition 1. Let $\mathcal{T} = (\mathcal{U} \cup \mathcal{B}) \times \mathcal{U} \times (\mathcal{U} \cup \mathcal{B} \cup \mathcal{L})$ be the infinite set of all possible RDF triples. A **Web of Linked Data** is a tuple $W = (D, \text{data}, \text{adoc})$ where:

³ For the sake of a more straightforward formalization we do not permit blank nodes in triple patterns. In practice, each blank node in a SPARQL query can be replaced by a new variable.

- D is a (finite or countably infinite) set of symbols that represent LD documents.
- $data$ is a total mapping $data: D \rightarrow 2^T$ such that $\forall d \in D: data(d)$ is finite and $\forall d_1, d_2 \in D: d_1 \neq d_2 \Rightarrow terms(data(d_1)) \cap \mathcal{B} \neq terms(data(d_2)) \cap \mathcal{B}$.
- $adoc$ is a partial, surjective mapping $adoc: \mathcal{U} \rightarrow D$.

While the three elements D , $data$, and $adoc$ completely define a Web of Linked Data in our model, we point out that these elements are abstract concepts and, thus, are not available to a query execution system. However, by retrieving LD documents, such a system may gradually obtain information about the Web. Based on this information the system may (partially) materialize these three elements. In the following we discuss the three elements and introduce additional concepts that we need to define queries.

We say a Web of Linked Data $W = (D, data, adoc)$ is *finite* if and only if D is finite; otherwise, W is *infinite*. Our model allows for infiniteness to cover cases where Linked Data about an infinite number of identifiable entities is generated on the fly. The Linked Open Numbers project (cf. Example [1](#)) illustrates that such cases are possible in practice. Another example is the LinkedGeoData project [4](#) which provides Linked Data about any circular and rectangular area on Earth [\[4\]](#). Covering these cases enables us to model queries over such data and analyze the effects of executing such queries.

Even if a Web of Linked Data $W = (D, data, adoc)$ is infinite, Definition [1](#) requires countability for D . We emphasize that this requirement does not restrict us in modeling the WWW as a Web of Linked Data: In the WWW we use URIs to locate documents that contain Linked Data. Even if URIs are not limited in length, they are words over a finite alphabet. Thus, the infinite set of all possible URIs is countable, as is the set of all documents that may be retrieved using URIs.

The mapping $data$ associates each LD document $d \in D$ in a Web of Linked Data $W = (D, data, adoc)$ with a finite set of RDF triples. In practice, these triples are obtained by parsing d after d has been retrieved from the Web. The actual retrieval mechanism is not relevant for our model. However, as prescribed by the RDF data model, Definition [1](#) requires that the data of each $d \in D$ uses a unique set of blank nodes.

To denote the (potentially infinite but countable) set of *all RDF triples* in W we write $AllData(W)$; i.e. it holds: $AllData(W) = \{data(d) \mid d \in D\}$.

Since we use URIs as identifiers for entities, we say that an LD document $d \in D$ *describes* the entity identified by URI $u \in \mathcal{U}$ if there exists $(s, p, o) \in data(d)$ such that $s = u$ or $o = u$. Notice, there might be multiple LD documents that describe an entity identified by u . However, according to the Linked Data principles, each $u \in \mathcal{U}$ may also serve as a reference to a specific LD document which is considered as an authoritative source of data about the entity identified by u . We model the relationship between URIs and authoritative LD documents by mapping $adoc$. Since some LD documents may be authoritative for multiple entities, we do not require injectivity for $adoc$. The “real world” mechanism for dereferencing URIs (i.e. learning about the location of the authoritative LD document) is not relevant for our model. For each $u \in \mathcal{U}$ that cannot be dereferenced (i.e. “broken links”) or that is not used in W it holds $u \notin \text{dom}(adoc)$.

A URI $u \in \mathcal{U}$ with $u \in \text{dom}(adoc)$ that is used in the data of an LD document $d_1 \in D$ constitutes a *data link* to the LD document $d_2 = adoc(u) \in D$. These data links form a

⁴ <http://linkedgedata.org>

graph structure which we call *link graph*. The vertices in such a graph represent the LD documents of the corresponding Web of Linked Data; edges represent data links.

To study the monotonicity of queries over a Web of Linked Data we require a concept of containment for such Webs. For this purpose, we introduce the notion of an induced subweb which resembles the concept of induced subgraphs in graph theory.

Definition 2. Let $W = (D, data, adoc)$ and $W' = (D', data', adoc')$ be Webs of Linked Data. W' is an **induced subweb of W** if i) $D' \subseteq D$, ii) $\forall d \in D' : data'(d) = data(d)$, and iii) $\forall u \in \mathcal{U}_{D'} : adoc'(u) = adoc(u)$ where $\mathcal{U}_{D'} = \{u \in \mathcal{U} \mid adoc(u) \in D'\}$.

It can be easily seen from Definition 2 that specifying D' is sufficient to unambiguously define an induced subweb $(D', data', adoc')$ of a given Web of Linked Data. Furthermore, it is easy to verify that for an induced subweb W' of a Web of Linked Data W it holds $AllData(W') \subseteq AllData(W)$.

In addition to the structural part, our data model introduces a general understanding of queries over a Web of Linked Data:

Definition 3. Let \mathcal{W} be the infinite set of all possible Webs of Linked Data (i.e. all 3-tuples that correspond to Definition 1) and let Ω be the infinite set of all possible valuations. A **Linked Data query** q is a total function $q: \mathcal{W} \rightarrow 2^\Omega$.

The notions of satisfiability and monotonicity carry over naturally to Linked Data queries: A Linked Data query q is *satisfiable* if there exists a Web of Linked Data W such that $q(W)$ is not empty. A Linked Data query q is *nontrivially satisfiable* if there exists a Web of Linked Data W and a valuation μ such that i) $\mu \in q(W)$ and ii) $dom(\mu) \neq \emptyset$. A Linked Data query q is *monotonic* if for every pair W_1, W_2 of Webs of Linked Data it holds: If W_1 is an induced subweb of W_2 , then $q(W_1) \subseteq q(W_2)$.

3.2 Computation Model

Usually, functions are computed over structures that are assumed to be fully (and directly) accessible. In contrast, we focus on Webs of Linked Data in which accessibility is limited: To discover LD documents and access their data we have to dereference URIs, but the full set of those URIs for which we may retrieve documents is unknown. Hence, to properly analyze a query model for Webs of Linked Data we must define a model for computing functions on such a Web. This section introduces such a model.

In the context of queries over a hypertext-centric view of the WWW, Abiteboul and Vianu introduce a specific Turing machine called Web machine [11]. Mendelzon and Milo propose a similar machine model [15]. These machines formally capture the limited data access capabilities on the WWW and thus present an adequate abstraction for computations over a structure such as the WWW. Based on these machines the authors introduce particular notions of computability for queries over the WWW. These notions are: *(finitely) computable queries*, which correspond to the traditional notion of computability; and *eventually computable queries* whose computation may not terminate but each element of the query result will eventually be reported during the computation. We adopt the ideas of Abiteboul and Vianu and of Mendelzon and Milo for our work. More precisely, we adapt the idea of a Web machine to our scenario of a Web of Linked

Data. We call our machine a *Linked Data machine* (or LD machine, for short). Based on this machine we shall define finite and eventual computability for Linked Data queries.

Encoding (fragments of) a Web of Linked Data $W = (D, data, adoc)$ on the tapes of such an LD machine is straightforward because all relevant structures, such as the sets D or \mathcal{U} , are countably infinite. In the remainder of this paper we write $enc(x)$ to denote the encoding of some element x (e.g. a single RDF triple, a set of triples, a full Web of Linked Data, a valuation, etc.). For a detailed definition of the encodings we use in this paper, we refer to the appendix in [11]. We now define LD machine:

Definition 4. An *LD machine* is a multi-tape Turing machine with five tapes and a finite set of states, including a special state called *expand*. The five tapes include two, read-only input tapes: i) an ordinary input tape and ii) a right-infinite Web tape which can only be accessed in the *expand* state; two work tapes: iii) an ordinary, two-way infinite work tape and iv) a right-infinite link traversal tape; and v) a right-infinite, append-only output tape. Initially, the work tapes and the output tape are empty, the Web tape contains a (potentially infinite) word that encodes a Web of Linked Data, and the ordinary input tape contains an encoding of further input (if any). Any LD machine operates like an ordinary multi-tape Turing machine except when it reaches the *expand* state. In this case LD machines perform the following *expand* procedure: The machine inspects the word currently stored on the link traversal tape. If the suffix of this word is the encoding $enc(u)$ of some URI $u \in \mathcal{U}$ and the word on the Web tape contains $\# enc(u) enc(adoc(u)) \#$, then the machine appends $enc(adoc(u)) \#$ to the (right) end of the word on the link traversal tape by copying from the Web tape; otherwise, the machine appends $\#$ to the word on the link traversal tape.

Notice how any LD machine M is limited in the way it may access a Web of Linked Data $W = (D, data, adoc)$ that is encoded on its Web tape: M may use the data of any particular $d \in D$ only after it performed the *expand* procedure using a URI $u \in \mathcal{U}$ for which $adoc(u) = d$. Hence, the *expand* procedure simulates a URI based lookup which conforms to the (typical) data access method on the WWW. We now use LD machines to adapt the notion of finite and eventual computability [11] for Linked Data queries:

Definition 5. A Linked Data query q is *finitely computable* if there exists an LD machine which, for any Web of Linked Data W encoded on the Web tape, halts after a finite number of steps and produces a possible encoding of $q(W)$ on its output tape.

Definition 6. A Linked Data q query is *eventually computable* if there exists an LD machine whose computation on any Web of Linked Data W encoded on the Web tape has the following two properties: 1.) the word on the output tape at each step of the computation is a prefix of a possible encoding of $q(W)$ and 2.) the encoding $enc(\mu')$ of any $\mu' \in q(W)$ becomes part of the word on the output tape after a finite number of computation steps.

Any machine for a non-satisfiable query may immediately report the empty result. Thus:

Fact 1. *Non-satisfiable Linked Data queries are finitely computable.*

In our analysis of SPARQL-based Linked Data queries we shall discuss decision problems that have a Web of Linked Data W as input. For such problems we assume the computation may only be performed by an LD machine with $enc(W)$ on its Web tape:

Definition 7. Let \mathcal{W}' be a (potentially infinite) set of Webs of Linked Data (each of which may be infinite itself); let \mathcal{X} be an arbitrary (potentially infinite) set of finite structures; and let $DP \subseteq \mathcal{W}' \times \mathcal{X}$. The decision problem for DP , that is, decide for any $(W, X) \in \mathcal{W}' \times \mathcal{X}$ whether $(W, X) \in DP$, is **LD machine decidable** if there exist an LD machine whose computation on any $W \in \mathcal{W}'$ encoded on the Web tape and any $X \in \mathcal{X}$ encoded on the ordinary input tape, has the following property: The machine halts in an accepting state if $(W, X) \in DP$; otherwise the machine halts in a rejecting state.

Obviously, any (Turing) decidable problem that does not have a Web of Linked Data as input, is also LD machine decidable because LD machines are Turing machines; for these problems the corresponding set \mathcal{W}' is empty .

4 Full-Web Semantics

Based on the concepts introduced in the previous section we now define and study approaches that adapt SPARQL as a language for expressing Linked Data queries.

The first approach that we discuss is full-Web semantics where the scope of each query is the complete set of Linked Data on the Web. Hereafter, we refer to SPARQL queries under this full-Web semantics as $\text{SPARQL}_{\text{LD}}$ queries. The definition of these queries is straightforward and makes use of SPARQL expressions and their semantics:

Definition 8. Let P be a SPARQL expression. The **SPARQL_{LD} query that uses P** , denoted by \mathcal{Q}^P , is a Linked Data query that, for any Web of Linked Data W , is defined as: $\mathcal{Q}^P(W) = \llbracket P \rrbracket_{\text{AllData}(W)}$. Each valuation $\mu \in \mathcal{Q}^P(W)$ is a **solution** for \mathcal{Q}^P in W .

In the following we study satisfiability, monotonicity, and computability of $\text{SPARQL}_{\text{LD}}$ queries and we discuss implications of querying Webs of Linked Data that are infinite.

4.1 Satisfiability, Nontrivial Satisfiability, Monotonicity, and Computability

For satisfiability and monotonicity we may show the following dependencies.

Proposition 1. Let \mathcal{Q}^P be a $\text{SPARQL}_{\text{LD}}$ query that uses SPARQL expression P .

1. \mathcal{Q}^P is satisfiable if and only if P is satisfiable.
2. \mathcal{Q}^P is nontrivially satisfiable if and only if P is nontrivially satisfiable.
3. \mathcal{Q}^P is monotonic if and only if P is monotonic.

We now discuss computability. Since all non-satisfiable $\text{SPARQL}_{\text{LD}}$ queries are finitely computable (recall Fact [1](#)), we focus on satisfiable $\text{SPARQL}_{\text{LD}}$ queries. Our first main result shows that the computability of such queries depends on their monotonicity:

Theorem 1. If a satisfiable $\text{SPARQL}_{\text{LD}}$ query is monotonic, then it is eventually computable (but not finitely computable); otherwise, it is not even eventually computable.

In addition to a direct dependency between monotonicity and computability, Theorem [1](#) shows that not any satisfiable $\text{SPARQL}_{\text{LD}}$ query is finitely computable; instead, such queries are at best eventually computable. The reason for this limitation is the infiniteness of \mathcal{U} : To (fully) compute a satisfiable $\text{SPARQL}_{\text{LD}}$ query, an LD machine requires

access to the data of *all* LD documents in the queried Web of Linked Data. Recall that, initially, the machine has no information about what URI to use for performing an expand procedure with which it may access any particular document. Hence, to ensure that all documents have been accessed, the machine must expand all $u \in \mathcal{U}$. This process never terminates because \mathcal{U} is infinite. Notice, a real query system for the WWW would have a similar problem: To guarantee that such a system sees all documents, it must enumerate and lookup all (HTTP scheme) URIs.

The computability of any Linked Data query is a general, input independent property which covers the worst case (recall, the requirements given in Definitions 5 and 6 must hold for any Web of Linked Data). As a consequence, in certain cases the computation of some (eventually computable) SPARQL_{LD} queries may still terminate:

Example 3. Let $Q^{P_{E_{\square}}}$ be a monotonic SPARQL_{LD} query which uses the SPARQL expression $P_{E_{\square}} = (u_1, u_2, u_3)$ that we introduce in Example 2. Recall, $P_{E_{\square}}$ is satisfiable but *not* nontrivially satisfiable. The same holds for $Q^{P_{E_{\square}}}$ (cf. Proposition 1). An LD machine for $Q^{P_{E_{\square}}}$ may take advantage of this fact: As soon as the machine discovers an LD document which contains RDF triple (u_1, u_2, u_3) , the machine may halt (after reporting $\{\mu_{\emptyset}\}$ with $\text{dom}(\mu_{\emptyset}) = \emptyset$ as the complete query result). In this particular case the machine would satisfy the requirements for finite computability. However, $Q^{P_{E_{\square}}}$ is still only eventually computable because there exist Webs of Linked Data that do not contain any LD document with RDF triple (u_1, u_2, u_3) ; any (complete) LD machine based computation of $Q^{P_{E_{\square}}}$ over such a Web cannot halt (cf. proof of Theorem 1).

The example illustrates that the computation of an eventually computable query over a particular Web of Linked Data may terminate. This observation leads us to a decision problem which we denote as $\text{TERMINATION}(\text{SPARQL}_{\text{LD}})$. This problem takes a Web of Linked Data W and a satisfiable SPARQL_{LD} query Q^P as input and asks whether an LD machine exists that computes $Q^P(W)$ and halts. For discussing this problem we note that the query in Example 3 represents a special case, that is, SPARQL_{LD} queries which are satisfiable but not nontrivially satisfiable. The reason why an LD machine for such a query may halt, is the implicit knowledge that the query result is complete once the machine identified the empty valuation μ_{\emptyset} as a solution. Such a completeness criterion does not exist for any nontrivially satisfiable SPARQL_{LD} query:

Lemma 1. *There is not any nontrivially satisfiable SPARQL_{LD} query Q^P for which exists an LD machine that, for any Web of Linked Data W encoded on the Web tape, halts after a finite number of computation steps and outputs an encoding of $Q^P(W)$.*

Lemma 1 shows that the answer to $\text{TERMINATION}(\text{SPARQL}_{\text{LD}})$ is negative in most cases. However, the problem in general is undecidable (for LD machines) since the input for the problem includes queries that correspond to the aforementioned special case.

Theorem 2. $\text{TERMINATION}(\text{SPARQL}_{\text{LD}})$ is not LD machine decidable.

4.2 Querying an Infinite Web of Linked Data

The limited computability of SPARQL_{LD} queries that our results in the previous section show, is a consequence of the infiniteness of \mathcal{U} and not of a possible infiniteness of the

queried Web. We now focus on the implications of potentially infinite Webs of Linked Data for SPARQL_{LD} queries. However, we assume a *finite* Web first:

Proposition 2. *SPARQL_{LD} queries over a finite Web of Linked Data have a finite result.*

The following example illustrates that a similarly general statement does not exist when the queried Web is infinite such as the WWW.

Example 4. Let $W_{\text{inf}} = (D_{\text{inf}}, \text{data}_{\text{inf}}, \text{adoc}_{\text{inf}})$ be an *infinite* Web of Linked Data that contains LD documents for all natural numbers (similar to the documents in Example 1). Hence, for each natural number⁵ $k \in \mathbb{N}^+$, identified by $u_k \in \mathcal{U}$, exists an LD document $\text{adoc}_{\text{inf}}(u_k) = d_k \in D_{\text{inf}}$ such that $\text{data}_{\text{inf}}(d_k) = \{(u_k, \text{succ}, u_{k+1})\}$ where $\text{succ} \in \mathcal{U}$ identifies the successor relation for \mathbb{N}^+ . Furthermore, let $P_1 = (u_1, \text{succ}, ?v)$ and $P_2 = (?x, \text{succ}, ?y)$ be SPARQL expressions. It can be seen easily that the result of SPARQL_{LD} query \mathcal{Q}^{P_1} over W_{inf} is finite, whereas, $\mathcal{Q}^{P_2}(W_{\text{inf}})$ is infinite.

The example demonstrates that some SPARQL_{LD} queries have a finite result over some infinite Web of Linked Data and some queries have an infinite result. Consequently, we are interested in a decision problem FINITENESS(SPARQL_{LD}) which asks, given a (potentially infinite) Web of Linked Data W and a satisfiable SPARQL expression P , whether $\mathcal{Q}^P(W)$ is finite. Unfortunately, we cannot answer the problem in general:

Theorem 3. *FINITENESS(SPARQL_{LD}) is not LD machine decidable.*

5 Reachability-Based Semantics

Our results in the previous section show that SPARQL queries under full-Web semantics have a very limited computability. As a consequence, any SPARQL-based query approach for Linked Data that uses full-Web semantics requires some ad hoc mechanism to abort query executions and, thus, has to accept incomplete query results. Depending on the abort mechanism the query execution may even be nondeterministic. If we take these issues as an obstacle, we are interested in an alternative, well-defined semantics for SPARQL over Linked Data. In this section we discuss a family of such semantics which we call *reachability-based semantics*. These semantics restrict the scope of queries to data that is reachable by traversing certain data links using a given set of URIs as starting points. Hereafter, we refer to queries under any reachability-based semantics as *SPARQL_{LD(R)} queries*. In the remainder of this section we formally introduce reachability-based semantics, discuss theoretical properties of SPARQL_{LD(R)} queries, and compare SPARQL_{LD(R)} to SPARQL_{LD}.

5.1 Definition

The basis of any reachability-based semantics is a notion of reachability of LD documents. Informally, an LD document is reachable if there exists a (specific) path in the

⁵ In this paper we write \mathbb{N}^+ to denote the set of all natural numbers without zero.

link graph of a Web of Linked Data to the document in question; the potential starting points for such a path are LD documents that are authoritative for a given set of entities. However, allowing for arbitrary paths might be questionable in practice because this approach would require following *all* data links (recursively) for answering a query completely. Consequently, we introduce the notion of a reachability criterion that supports an explicit specification of what data links should be followed.

Definition 9. Let \mathcal{T} be the infinite set of all possible RDF triples and let \mathcal{P} be the infinite set of all possible SPARQL expressions. A **reachability criterion** c is a (Turing) computable function $c : \mathcal{T} \times \mathcal{U} \times \mathcal{P} \rightarrow \{\text{true}, \text{false}\}$.

An example for a reachability criterion is c_{All} which corresponds to the aforementioned approach of allowing for arbitrary paths to reach LD documents; hence, for each tuple $(t, u, Q) \in \mathcal{T} \times \mathcal{U} \times \mathcal{Q}$ it holds $c_{\text{All}}(t, u, Q) = \text{true}$. The complement of c_{All} is c_{None} which *always* returns false. Another example is c_{Match} which specifies the notion of reachability that we use for link traversal based query execution [10,12].

$$c_{\text{Match}}(t, u, P) = \begin{cases} \text{true} & \text{if there exists a triple pattern } tp \text{ in } P \text{ and } t \text{ matches } tp, \\ \text{false} & \text{else.} \end{cases}$$

where an RDF triple $t = (x_1, x_2, x_3)$ *matches* a triple pattern $tp = (\tilde{x}_1, \tilde{x}_2, \tilde{x}_3)$ if for all $i \in \{1, 2, 3\}$ holds: If $\tilde{x}_i \notin \mathcal{V}$, then $\tilde{x}_i = x_i$.

We call a reachability criterion c_1 *less restrictive than* another criterion c_2 if i) for each $(t, u, P) \in \mathcal{T} \times \mathcal{U} \times \mathcal{P}$ for which $c_2(t, u, P) = \text{true}$, also holds $c_1(t, u, P) = \text{true}$ and ii) there exist a $(t', u', P') \in \mathcal{T} \times \mathcal{U} \times \mathcal{P}$ such that $c_1(t', u', P') = \text{true}$ but $c_2(t', u', P') = \text{false}$. It can be seen that c_{All} is the least restrictive criterion, whereas c_{None} is the most restrictive criterion. We now define reachability of LD documents:

Definition 10. Let $S \subset \mathcal{U}$ be a finite set of seed URIs; let c be a reachability criterion; let P be a SPARQL expression; and let $W = (D, \text{data}, \text{adoc})$ be a Web of Linked Data. An LD document $d \in D$ is (c, P) -**reachable from S in W** if either

1. there exists a URI $u \in S$ such that $\text{adoc}(u) = d$; or
2. there exist $d' \in D$, $t \in \text{data}(d')$, and $u \in \text{uris}(t)$ such that i) d' is (c, P) -reachable from S in W , ii) $\text{adoc}(u) = d$, and iii) $c(t, u, P) = \text{true}$.

Based on reachability of LD documents we define reachable parts of a Web of Linked Data. Such a part is an induced subweb covering all reachable LD documents. Formally:

Definition 11. Let $S \subset \mathcal{U}$ be a finite set of URIs; let c be a reachability criterion; let P be a SPARQL expression; and let $W = (D, \text{data}, \text{adoc})$ be a Web of Linked Data. The (S, c, P) -**reachable part of W** , denoted by $W_c^{(S,P)}$, is an induced subweb $(D_{\mathfrak{R}}, \text{data}_{\mathfrak{R}}, \text{adoc}_{\mathfrak{R}})$ of W such that $D_{\mathfrak{R}} = \{d \in D \mid d \text{ is } (c, P)\text{-reachable from } S \text{ in } W\}$.

We now use the concept of reachable parts to define SPARQL_{LD(R)} queries.

Definition 12. Let $S \subset \mathcal{U}$ be a finite set of URIs; let c be a reachability criterion; and let P be a SPARQL expression. The **SPARQL_{LD(R)} query that uses P , S , and c** , denoted by $Q_c^{P,S}$, is a Linked Data query that, for any Web of Linked Data W , is defined as $Q_c^{P,S}(W) = \llbracket P \rrbracket_{\text{AllData}(W_c^{(S,P)})}$ (where $W_c^{(S,P)}$ is the (S, c, P) -reachable part of W).

As can be seen from Definition 12, our notion of $\text{SPARQL}_{\text{LD(R)}}$ consists of a family of (reachability-based) query semantics, each of which is characterized by a certain reachability criterion. Therefore, we refer to $\text{SPARQL}_{\text{LD(R)}}$ queries for which we use a particular reachability criterion c as $\text{SPARQL}_{\text{LD(R)}}$ queries *under c -semantics*.

Definition 12 also shows that query results depend on the given set $S \subset \mathcal{U}$ of seed URIs. It is easy to see that any $\text{SPARQL}_{\text{LD(R)}}$ query which uses an empty set of seed URIs is not satisfiable and, thus, monotonic and finitely computable. We therefore consider only nonempty sets of seed URIs in the remainder of this paper.

5.2 Completeness and Infiniteness

Definition 12 defines precisely what the sound and complete result of any $\text{SPARQL}_{\text{LD(R)}}$ query $Q_c^{P,S}$ over any Web of Linked Data W is. However, in contrast to $\text{SPARQL}_{\text{LD}}$, it is not guaranteed that such a (complete) $\text{SPARQL}_{\text{LD(R)}}$ result is complete w.r.t. all data on W . This difference can be attributed to the fact that the corresponding (S, c, P) -reachable part of W may not cover W as a whole. We emphasize that such an incomplete coverage is even possible for the reachability criterion c_{All} because the link graph of W may not be connected; therefore, c_{All} -semantics differs from full-Web semantics. The following result relates $\text{SPARQL}_{\text{LD(R)}}$ queries to their $\text{SPARQL}_{\text{LD}}$ counterparts.

Proposition 3. *Let $Q_c^{P,S}$ be a $\text{SPARQL}_{\text{LD(R)}}$ query; let Q^P be the $\text{SPARQL}_{\text{LD}}$ query that uses the same SPARQL expression as $Q_c^{P,S}$; let W be a Web of Linked Data. It holds:*

1. *If Q^P is monotonic, then $Q_c^{P,S}(W) \subseteq Q^P(W)$.*
2. *$Q_c^{P,S}(W) = Q^P(W_c^{(S,P)})$. (recall, $W_c^{(S,P)}$ is the (S, c, P) -reachable part of W)*

Since any $\text{SPARQL}_{\text{LD}}$ query over a finite Web of Linked Data has a finite result (cf. Proposition 2), we use Proposition 3, case 2, to show the same for $\text{SPARQL}_{\text{LD(R)}}$:

Proposition 4. *The result of any $\text{SPARQL}_{\text{LD(R)}}$ query $Q_c^{P,S}$ over a finite Web of Linked Data W is finite; so is the (S, c, P) -reachable part of W .*

For the case of an *infinite* Web of Linked Data the results of $\text{SPARQL}_{\text{LD(R)}}$ queries may be either finite or infinite. In Example 4 we found the same heterogeneity for $\text{SPARQL}_{\text{LD}}$. However, for $\text{SPARQL}_{\text{LD(R)}}$ we may identify the following dependencies.

Proposition 5. *Let $S \subset \mathcal{U}$ be a finite, nonempty set of URIs; let c and c' be reachability criteria; and let P be a SPARQL expression. Let W be an infinite Web of Linked Data.*

1. *$W_{c_{\text{None}}}^{(S,P)}$ is always finite; so is $Q_{c_{\text{None}}}^{P,S}(W)$.*
2. *If $W_c^{(S,P)}$ is finite, then $Q_c^{P,S}(W)$ is finite.*
3. *If $Q_c^{P,S}(W)$ is infinite, then $W_c^{(S,P)}$ is infinite.*
4. *If c is less restrictive than c' and $W_c^{(S,P)}$ is finite, then $W_{c'}^{(S,P)}$ is finite.*
5. *If c' is less restrictive than c and $W_c^{(S,P)}$ is infinite, then $W_{c'}^{(S,P)}$ is infinite.*

Proposition 5 provides valuable insight into the dependencies between reachability criteria, the (in)finiteness of reachable parts of an infinite Web, and the (in)finiteness of query results. In practice, however, we are primarily interested in answering two

decision problems: FINITENESSREACHABLEPART and FINITENESS(SPARQL_{LD(R)}). While the latter problem is the SPARQL_{LD(R)} equivalent to FINITENESS(SPARQL_{LD}) (cf. Section 4.2), the former has the same input as FINITENESS(SPARQL_{LD(R)}) (that is, a Web of Linked Data and a SPARQL_{LD(R)} query) and asks whether the corresponding reachable part of the given Web is finite. Both problems are undecidable in our context:

Theorem 4. FINITENESSREACHABLEPART and FINITENESS(SPARQL_{LD(R)}) are not LD machine decidable.

5.3 Satisfiability, Nontrivial Satisfiability, Monotonicity, and Computability

We now investigate satisfiability, nontrivial satisfiability, monotonicity, and computability of SPARQL_{LD(R)} queries. First, we identify the following dependencies.

Proposition 6. Let $Q_c^{P,S}$ be a SPARQL_{LD(R)} query that uses a nonempty $S \subset \mathcal{U}$.

1. $Q_c^{P,S}$ is satisfiable if and only if P is satisfiable.
2. $Q_c^{P,S}$ is nontrivially satisfiable if and only if P is nontrivially satisfiable.
3. $Q_c^{P,S}$ is monotonic if P is monotonic.

Proposition 6 reveals a first major difference between SPARQL_{LD(R)} and SPARQL_{LD}: The statement about monotonicity in that proposition is only a material conditional, whereas it is a biconditional in the case of SPARQL_{LD} (cf. Proposition 1). The reason for this disparity are SPARQL_{LD(R)} queries for which monotonicity is independent of the corresponding SPARQL expression. The following proposition identifies such a case.

Proposition 7. Any SPARQL_{LD(R)} query $Q_{c_{\text{None}}}^{P,S}$ is monotonic if $|S| = 1$.

Before we may come back to the aforementioned disparity, we focus on the computability of SPARQL_{LD(R)} queries. We first show the following, noteworthy result.

Lemma 2. Let $Q_c^{P,S}$ be a SPARQL_{LD(R)} query that is nontrivially satisfiable. There exists an LD machine that computes $Q_c^{P,S}$ over any (potentially infinite) Web of Linked Data W and that halts after a finite number of computation steps with an encoding of $Q_c^{P,S}(W)$ on its output tape if and only if the (S, c, P) -reachable part of W is finite.

The importance of Lemma 2 lies in showing that some computations of nontrivially satisfiable SPARQL_{LD(R)} queries may terminate. This possibility presents another major difference between SPARQL_{LD(R)} and SPARQL_{LD} (recall Lemma 1 which shows that any possible computation of nontrivially satisfiable SPARQL_{LD} queries never terminates). Based on Lemma 2 we may even show that a particular class of satisfiable SPARQL_{LD(R)} queries are finitely computable. This class comprises all queries that use a reachability criterion which ensures the finiteness of reachable parts of any queried Web of Linked Data. We define this property of reachability criteria as follows:

Definition 13. A reachability criterion c ensures finiteness if for any Web of Linked Data W , any (finite) set $S \subset \mathcal{U}$ of seed URIs, and any SPARQL expression P , the (S, c, P) -reachable part of W is finite.

We may now show the aforementioned result:

Proposition 8. *Let c be a reachability criterion that ensures finiteness. $\text{SPARQL}_{\text{LD}(\text{R})}$ queries under c -semantics are finitely computable.*

While it remains an open question whether the property to ensure finiteness is decidable for all reachability criteria, it is easy to verify the property for criteria which always only accept a given, constant set of data links. For a formal discussion of such criteria, which we call *constant reachability criteria*, we refer to the appendix in [11]. c_{None} is a special case of these criteria; Proposition 5 case 1 verifies that c_{None} ensures finiteness.

Notice, for any reachability criterion c that ensures finiteness, the computability of $\text{SPARQL}_{\text{LD}(\text{R})}$ queries under c -semantics does not depend on the monotonicity of these queries. This independence is another difference to $\text{SPARQL}_{\text{LD}}$ queries (recall Theorem 1). However, for any other reachability criterion (including c_{Match} and c_{All}), we have a similar dependency between monotonicity and computability of (satisfiable) $\text{SPARQL}_{\text{LD}(\text{R})}$ queries, that we have for $\text{SPARQL}_{\text{LD}}$ queries (recall Theorem 1):

Theorem 5. *Let c_{nf} be a reachability criterion that does not ensure finiteness. If a satisfiable $\text{SPARQL}_{\text{LD}(\text{R})}$ query $Q_{c_{\text{nf}}}^{P,S}$ (under c_{nf} -semantics) is monotonic, then $Q_{c_{\text{nf}}}^{P,S}$ is either finitely computable or eventually computable; otherwise, $Q_{c_{\text{nf}}}^{P,S}$ may not even be eventually computable.*

By comparing Theorems 1 and 5 we notice that $\text{SPARQL}_{\text{LD}}$ queries and $\text{SPARQL}_{\text{LD}(\text{R})}$ queries (that use a reachability criterion which does not ensure finiteness) feature a similarly limited computability. However, the reasons for both of these results differ significantly: In the case of $\text{SPARQL}_{\text{LD}}$ the limitation follows from the infiniteness of \mathcal{U} , whereas, for $\text{SPARQL}_{\text{LD}(\text{R})}$ the limitation is a consequence of the possibility to query an infinitely large Web of Linked Data.

However, even if the computability of many $\text{SPARQL}_{\text{LD}(\text{R})}$ queries is as limited as that of their $\text{SPARQL}_{\text{LD}}$ counterparts, there is another major difference: Lemma 2 shows that for (nontrivially satisfiable) $\text{SPARQL}_{\text{LD}(\text{R})}$ queries which are not finitely computable, the computation over some Webs of Linked Data may still terminate; this includes all finite Webs (cf. Proposition 4) but also some infinite Webs (cf. proof of Lemma 2). Such a possibility does not exist for nontrivially satisfiable $\text{SPARQL}_{\text{LD}}$ queries (cf. Lemma 1). Nonetheless, the termination problem for $\text{SPARQL}_{\text{LD}(\text{R})}$ is undecidable in our context.

Theorem 6. $\text{TERMINATION}(\text{SPARQL}_{\text{LD}(\text{R})})$ is not LD machine decidable.

We now come back to the impossibility for showing that $\text{SPARQL}_{\text{LD}(\text{R})}$ queries (with a nonempty set of seed URIs) are monotonic *only if* their SPARQL expression is monotonic. Recall, for some $\text{SPARQL}_{\text{LD}(\text{R})}$ queries monotonicity is irrelevant for identifying the computability (cf. Proposition 8). We are primarily interested in the monotonicity of all other (satisfiable) $\text{SPARQL}_{\text{LD}(\text{R})}$ queries because for those queries computability depends on monotonicity as we show in Theorem 5. Remarkably, for those queries it is possible to show the required dependency that was missing from Proposition 6:

Proposition 9. *Let $Q_{c_{\text{nf}}}^{P,S}$ be a $\text{SPARQL}_{\text{LD}(\text{R})}$ query that uses a finite, nonempty $S \subset \mathcal{U}$ and a reachability criterion c_{nf} which does not ensure finiteness. $Q_{c_{\text{nf}}}^{P,S}$ is monotonic only if P is monotonic.*

6 Conclusions

Our investigation of SPARQL as a language for Linked Data queries reveals the following main results. Some special cases aside, the computability of queries under any of the studied semantics is limited and no guarantee for termination can be given. For reachability-based semantics it is at least possible that some of the (non-special case) query computations terminate; although, in general it is undecidable which. As a consequence, any SPARQL-based query system for Linked Data on the Web must be prepared for query executions that discover an infinite amount of data and that do not terminate.

Our results also show that –for reachability-based semantics– the aforementioned issues must be attributed to the possibility for infiniteness in the queried Web (which is a result of data generating servers). Therefore, it seems worthwhile to study approaches for detecting whether the execution of a SPARQL_{LD(R)} query traverses an infinite path in the queried Web. However, the mentioned issues may also be addressed by another, alternative well-defined semantics that restricts the scope of queries even further (or differently) than our reachability-based semantics. It remains an open question how such an alternative may still allow for queries that tap the full potential of the Web.

We also show that computability depends on satisfiability and monotonicity and that for (almost all) SPARQL-based Linked Data queries, these two properties directly correspond to the same property for the used SPARQL expression. While Arenas and Pérez show that the core fragment of SPARQL without OPT is monotonic [3], it requires further work to identify (non-)satisfiable and (non-)monotonic fragments and, thus, enable an explicit classification of SPARQL-based Linked Data queries w.r.t. computability.

References

1. Abiteboul, S., Vianu, V.: Queries and computation on the web. *Theoretical Computer Science* 239(2) (2000)
2. Angles, R., Gutierrez, C.: The Expressive Power of SPARQL. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) *ISWC 2008*. LNCS, vol. 5318, pp. 114–129. Springer, Heidelberg (2008)
3. Arenas, M., Pérez, J.: Querying Semantic Web Data with SPARQL. In: *PODS* (2011)
4. 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)
5. Berners-Lee, T.: *Linked Data* (2006), <http://www.w3.org/DesignIssues/LinkedData.html>
6. Bizer, C., Heath, T., Berners-Lee, T.: *Linked Data – the story so far*. *Journal on Semantic Web and Information Systems* 5(3) (2009)
7. Bouquet, P., Ghidini, C., Serafini, L.: Querying the Web of Data: A Formal Approach. In: Gómez-Pérez, A., Yu, Y., Ding, Y. (eds.) *ASWC 2009*. LNCS, vol. 5926, pp. 291–305. Springer, Heidelberg (2009)
8. Florescu, D., Levy, A.Y., Mendelzon, A.O.: Database techniques for the world-wide web: A survey. *SIGMOD Record* 27(3) (1998)
9. Harth, A., Hose, K., Karnstedt, M., Polleres, A., Sattler, K.-U., Umbrich, J.: Data Summaries for On-Demand Queries over Linked Data. In: *WWW* (2010)

10. Hartig, O.: Zero-Knowledge Query Planning for an Iterator Implementation of Link Traversal Based Query Execution. 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. 154–169. Springer, Heidelberg (2011)
11. Hartig, O.: SPARQL for a Web of Linked Data: Semantics and Computability (Extended Version). CoRR abs/1203.1569 (2012), <http://arxiv.org/abs/1203.1569>
12. Hartig, O., Bizer, C., Freytag, J.-C.: Executing SPARQL Queries over the Web of Linked Data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 293–309. Springer, Heidelberg (2009)
13. Ladwig, G., Tran, T.: Linked Data Query Processing Strategies. 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. 453–469. Springer, Heidelberg (2010)
14. Ladwig, G., Tran, T.: SIHJoin: Querying Remote and Local Linked Data. 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. 139–153. Springer, Heidelberg (2011)
15. Mendelzon, A.O., Milo, T.: Formal models of web queries. *Inf. Systems* 23(8) (1998)
16. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. *ACM Transactions on Database Systems* 34(3) (2009)
17. Picalausa, F., Vansummeren, S.: What are real SPARQL queries like? In: SWIM (2011)
18. Schmidt, M., Meier, M., Lausen, G.: Foundations of sparql query optimization. In: Proc. of the 13th Int. Conference on Database Theory, ICDT (2010)
19. Vrandečić, D., Krötzsch, M., Rudolph, S., Lösch, U.: Leveraging non-lexical knowledge for the linked open data web. In: RAFT (2010)

Linked Data-Based Concept Recommendation: Comparison of Different Methods in Open Innovation Scenario

Danica Damljanovic¹, Milan Stankovic^{2,3}, and Philippe Laublet³

¹ University of Sheffield,

Department of Computer Science, United Kingdom
d.damljanovic@dcs.shef.ac.uk

² Hypios, 187 rue du Temple, 75003 Paris, France
milan.stankovic@hypios.com

³ STIH, Université Paris-Sorbonne, 28 rue Serpente, 75006 Paris, France
philippe.laublet@paris-sorbonne.fr

Abstract. Concept recommendation is a widely used technique aimed to assist users to chose the right tags, improve their Web search experience and a multitude of other tasks. In finding potential problem solvers in Open Innovation (OI) scenarios, the concept recommendation is of a crucial importance as it can help to discover the right topics, directly or laterally related to an innovation problem. Such topics then could be used to identify relevant experts. We propose two Linked Data-based concept recommendation methods for topic discovery. The first one, hyProximity, exploits only the particularities of Linked Data structures, while the other one applies a well-known Information Retrieval method, Random Indexing, to the linked data. We compare the two methods against the baseline in the gold standard-based and user study-based evaluations, using the real problems and solutions from an OI company.

Keywords: concept recommendation, structure-based similarity, semantic similarity, information retrieval, statistical semantics, linked data, ontologies, recommender systems, concept discovery, open innovation.

1 Introduction

The ability to innovate is essential to the economic wellbeing, growth and survival of most companies, especially when the market competition becomes strong. With the global economic uncertainties in recent years, companies and innovation experts started to question the old innovation models and seek new, more efficient ones. The paradigm of Open Innovation (OI) [1] is proposed as a way to outsource the innovation and seek solutions of R&D problems outside the company and its usual network of collaborators. OI is intended to leverage the existing knowledge and ideas, that the company is unaware of, and somehow democratize the process of innovation. In recent years, one interesting realisation of OI is the one that encourages the innovation

to emerge over the Web. This realization is the core business of companies such as Hypios.com, Innocentive.com and NineSigma.com, which provide Web innovation platforms where companies with R&D needs can post problems and find innovative solutions. The companies looking to innovate, called seekers, would represent their R&D needs through an innovation problem statement describing the context of the problem to be solved. Such a statement is then published on a problem-solving platform. Experts, called solvers then submit their solutions. The seeker then selects the best contribution and acquires the rights to use it, often in exchange for a prize to the solver and any other due fees.

Identification of the potential solvers and broadcasting problems to their attention is already used by the Web innovation platforms to boost the problem-solving activity [2]. In our previous work [3] we developed a method for solver finding that leverages the user traces (e.g., blogs, publications, presentations) available in Linked Data. However, finding the users with expertise in the problem topics is often not good enough, as Web innovation platforms also seek a greater diversity in solutions in terms of domains of knowledge that they are coming from, as well as in terms of different perspectives on the problem. Existing OI research strongly argues [4] that truly innovative solutions often come from solvers whose competence is not in the topics directly found in the problem description, but rather from those who are experts in a different domain and can transfer the knowledge from one domain to another. One way to identify and involve such lateral solvers is to search for the concepts lateral to the problem. Such concepts then might be contained in the user profiles of experts likely to submit solutions, or in the possibly existing solutions in the form of research publications or patents. The key challenge thus comes down to the identification of expertise topics, directly and laterally related to the problem in question.

With the emergence of the Linked Open Data (LOD) project¹, which continues stimulating creation, publication and interlinking the RDF graphs with those already in the LOD cloud, the amount of triples increased to 31 billion in 2011, and continues to grow. The value in the linked data is the large amount of concepts and relations between them that are made explicit and hence can be used to infer relations more effectively in comparison to deriving the same kind of relations from text. We propose two independently developed methods for topic discovery based on the Linked Data. The first method called *hyProximity*, is a structure-based similarity which explores different strategies based on the semantics inherent in an RDF graph, while the second one, *Random Indexing*, applies a well-known statistical semantics from Information Retrieval to RDF, in order to identify the relevant set of both direct and lateral topics. As the baseline we use the state of the art *adWords* keyword recommender from Google that finds similar topics based on their distribution in textual corpora and the corpora of search queries. We evaluate the performance of these methods based on solution descriptions submitted to Hypios in the last year that we use to create the ‘gold standard’. In addition, we conduct the user study aimed at gaining a more fine-grained insight into the nature of the generated recommendations.

¹ <http://linkeddata.org/>

2 State of the Art

In this section we discuss the existing measures of semantic relatedness and systems that use them in different scenarios including concept recommendation, followed by the approaches which use Linked Data.

Legacy Approaches: Although our focus is semantic relatedness of concepts our challenge is quite similar to term recommendation that has been studied for decades. Semantically related terms have been used to help users choose the right tags in collaborative filtering systems [5]; to discover alternative search queries [6]; for query refinement [7]; to enhance expert finding results [8]; for ontology maintenance [9], [10], and in many other scenarios. Different techniques and different sources are used and combined to develop Measures of Semantic Relatedness (MSRs). These measures could be split into two major categories: 1) graph-based measures and 2) distributional measures. In what follows we briefly examine each category of MSRs.

Graph-based measures make use of semantics (e.g., hyponymy or meronymy) and/or lexical relationships (e.g., synonyms) within a graph to determine semantic proximity between the concepts. For example, [11] exploits the hypernym graphs of Wordnet², [7] uses Gallois lattice to provide recommendations based on domain ontologies, whereas [12] uses the ODP taxonomy³. Some approaches (e.g. [10]) rely on the graph of Wikipedia categories to provide recommendations. Different approaches use different graph measures to calculate the semantic proximity of concepts. Shortest path is among the most common of such measures. It is often enhanced by taking into account the information content of the graph nodes [13]. To the best of our knowledge these approaches have not been applied to knowledge bases of size and richness comparable to that of DBpedia⁴. Even the Wikipedia-based measures (e.g. [10]) do not go beyond exploring categories, neither leverage the rich information inherent in DBpedia. The MSR that we propose in this paper builds upon the existing graph-based measures but is highly adapted to the rich structure of Linked Data sources, as it leverages different types of relations between the concepts in the graph.

Distributional measures rely on the distributional properties of words in large text corpora. Such MSRs deduce semantic relatedness by leveraging co-occurrences of concepts. For example, the approach presented in [14] uses co-occurrence in research papers, pondered with a function derived from the tf-idf measure [15] to establish a notion of word proximity. Co-occurrence in tags [5] and in search results [16] is also commonly used. In [17], the authors introduce Normalized Web Distance (NWD) as a generalization of Normalized Google Distance (NGD) [16] MSR and investigate its performance with six different search engines. The evaluation (based on the correlation with human judgment) demonstrated the best performance of Exalead-based NWD measure, closely followed by Yahoo!, Altavista, Ask and Google. A distributional measure applied for the task similar to ours is considered in [8], where using

² <http://wordnet.princeton.edu/>

³ <http://www.dmoz.org>

⁴ While DBpedia contains more than 3.5 million concepts, the current version of Wordnet has 206941 word-sense pairs, and ODP has half a million categories.

relevance feedback the distribution of keywords in expert profiles is used to discover new keywords that could enrich the search queries used to find experts. However, since the task was focused on finding the most relevant experts (as opposed to our focus on finding people likely to propose ideas and innovative solutions), the impact of the additional keywords was not purely satisfactory, as they tended to divert the expert search from its original focus.

In Information Retrieval, methods based on word space models can be seen as advanced distributional measures, as they are proven to be effective at finding words that appear in similar context (e.g. synonyms). That is, words that do not necessarily appear with each other, but with the same set of other words are found to be semantically related. The idea behind word space models is to use distributional statistics to generate high-dimensional vector spaces, where words are represented by context vectors. These context vectors are then used to indicate semantic similarity [18]. Examples of such methods are Latent Semantic Analysis (LSA) and Random Indexing (RI). The latter is considered more scalable and is used to discover implicit connections from large corpora such as in [19]. However, most of distributional measures are calculated based on text analysis and mining the relationships based on the distribution of words in text. In the large graphs such as the Linked Open Data cloud, the relationships already exist - the challenge is the selection of those that will lead towards more relevant concepts. Our approaches provide a ranking mechanism for this selection and finding both latent and directly related concepts, as they explore the semantics and implicit relations that exist in the large graphs.

Linked Data-Based Approaches: DBRec [20] uses Linked Data sets (DBpedia and the music-related data sets) to recommend music artists based on the specified user interest. The system proved as effective when making discoveries of relevant artists. The system uses a measure of semantic relatedness similar to our transversal strategy, but it is specific to the music domain, and works only with concepts that have the explicit type – Artist. Similarly, a video recommendation system based on DBpedia is proposed in [21] but it is also applicable for explicitly typed concept recommendations, while for our system this is not a requirement. Our general methodology is more broadly applicable, especially in cases where the desired concepts do not have explicit types.

3 Linked Data-Based Concept Recommendation Approaches

We present two Linked Data-based methods: 1) a structure-based similarity based solely on exploration of the semantics (defined concepts and relations) in an RDF graph, 2) a statistical semantics method, Random Indexing, applied to the RDF in order to calculate a structure-based statistical semantics similarity.

In general, our methods start from a set of Initial/seed Concepts (IC), and provide a ranked list of suggested concepts relevant to IC. A concept, in our context, is a Linked Data instance, defined with its URI, which represents a topic of human interest.

3.1 Structure-Based Similarity

In a typical Linked Data set covering general knowledge concepts, such as Freebase or DBpedia, links between concepts are established over two kinds of properties:

- **Hierarchical links:** The properties that help to organize the concepts based on their types (e.g., *rdf:type*⁵ and *rdfs:subclassOf*) or categories (e.g., *dcterms:subject* and *skos:broader*). The links created by those properties connect a concept to a *category concept* – the one serving to organize other concepts into classes.
- **Transversal links:** The properties that connect concepts without the aim to establish a classification or hierarchy. The majority of properties belong to this group, and they create direct and indirect links between ordinary, non-category concepts.

In our concept discovery we will treat the two types of links differently, due to their different nature, and we will devise three different approaches in order to be able to work with different data sets that might or might not contain both types of links. An early version of our approach treating hierarchical links only is presented in [22].

Generic Approach. Our approach for suggesting concepts relevant to a number of dinitial seed concepts is based on two main principles:

- **Closer concepts are more relevant.** Closer concepts are those that are at a shorter distance from the seed concepts. In the sense of our work the distances in the graph are not necessarily defined as the shortest path between the two nodes, but can be measured using different distance functions. The distance functions adapted to the nature of the graph that is used are discussed later.
- **Concepts found several times are more relevant.** Concepts found by exploration of the graph proximity of several seed concepts are more relevant than those appearing in the proximity of just one starting concept.

These general principles allow a diversity of concrete approaches that differ in distance functions used as well as in the weights given to candidates found at certain distances. In the remainder of this section we examine a variety of such different approaches. The general approach to calculating our measure of semantic proximity of a concept candidate to the set of seed concepts is using Equation (1). We refer to our notion of semantic proximity as hyProximity.

$$hyP(c, IC) = \sum_{c_i \in IC} dv(c, c_i) \quad (1)$$

HyProximity of a concept c to the set of initial concepts IC is the sum of values of the distance functions for distances between the concept c and each concept c_i from the set of initial seed concepts IC . The distance value between the concept c and an initial concept c_i , is denoted $dv(c, c_i)$ and is inversely proportional to the value of a chosen distance function, i.e. $dv(c, c_i) = p(c, c_i) / d(c, c_i)$. Different distance functions $d(c, c_i)$ and ponderation functions $p(c, c_i)$ can be used, and we will describe some of them in the reminder of this paper. The calculation of hyProximity can be performed using

⁵ All the prefixes used in this paper can be looked up at <http://prefix.cc>

the Algorithm 1. The generation of concept candidates as well as the distance value function depend on the exploration strategy used. In the following sub-sections we present a variety of strategies.

Algorithm 1.

1. *get initial topic concepts IC*
 2. *for each seed concept c in IC:*
 - a. *while distance_level++ < maxLevel:*
 - i. *generate concept candidates for the current distance_level*
 - ii. *for each concept candidate c_i:*
 1. *value(c_i) = dv(c, c_i)*
 2. *get previousValue(c_i) from Results*
 3. *put <c_i, previousValue(c_i)+value(c_i)> to Results*
 3. *sort Results in decreasing order of hyProximity*
-

Hierarchical Distance Functions. Hierarchical approaches exploit the links established over hierarchical properties. They focus on a subset of a given data set’s graph constructed only of hierarchical properties and the concepts that they connect.

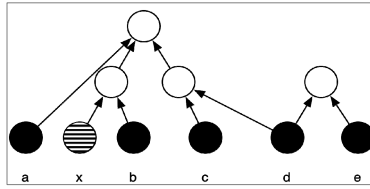


Fig. 1. A sample structure of a graph of concepts and categories

In finding candidate concepts using the hierarchical links, we can distinguish several ways to calculate distances. Our previous studies [22] allowed to isolate one particular function that gives best results, and that we will use here. Figure 1 represents an example graph of concepts (black nodes) and their categories/types⁶ (white nodes), and it will help us illustrate the distance function. Our hierarchical distance function considers all the non-category concepts that share a common category with x (in the case of our example – only the concept b) to be at distance 1. To find candidate concepts at distance n, we consider each category connected to the starting concept (x) over n links, and find all concepts connected to it over any of its subcategories. In our example, this approach would lead to considering {b,c,d} as being at distance 2 from x. Different ponderation schemes can be used along with the distance functions. A standard choice in graph distance functions is to use the informational content [13] of the category ($-\log(p)$ where p is the probability of finding the category in the graph of DBpedia categories when going from bottom up) as a pondering function. Applied to our case the pondering function $p(c, c_i)$ would take as a value the informational content of the first category over which one may find c when starting from c_i .

⁶ For the sake of simplicity, we will refer to both categories and types, as well as other possible grouping relations used to construct a hierarchy, as categories.

As the higher level categories normally have lower informational content, this function naturally gives higher hyProximity values to concept candidates found over categories closer to the initial concepts.

Transversal Distance Function. Our transversal function relies on a subset of the data set’s graph constituted of transversal properties relevant for a particular use case of interest, and the concepts that they connect. As the total number of properties in a data set might be high retrieving all the transversal links may yield time-consuming SPARQL queries. It is therefore useful to focus on those transversal properties that make connections relevant to a use case. The ways of identifying the set of useful properties for expert search are discussed in Section 4. The transversal distance function asserts the distance 1 for each link (direct or indirect) created between two concepts over one of the transversal properties. In our experiments we use the following ponderation function along with the transversal distance function: $p(c,ci) = -\log(n/M)$ where n is the number of concepts to which the candidate concept is connected over the same property that connects it to the initial concept. M is a large constant, larger than the maximum expected value of n . We use the total number of concepts in DBpedia as M in order to make the hyProximity values of the transversal strategy comparable to those of the hierarchical strategy where this same number is used to calculate the probabilities of finding a category in the graph. With such pondering function we give more importance to the concepts having a lower number of connections than to those acting as general connection hubs.

Mixed Distance Function. The mixed distance function asserts the distance n to all the concepts found at the distance n by the hierarchical function and those found at the same distance by the transversal function.

3.2 Structure-Based Statistical Semantics Similarity

Latent Semantic Analysis (LSA) [23] is one of the pioneer methods to automatically find contextually related words. The assumption behind this and other statistical semantics methods is that words which appear in the similar context (with the same set of other words) are synonyms. Synonyms tend not to co-occur with one another directly, so indirect inference is required to draw associations between words used to express the same idea [19]. This method has been shown to approximate human performance in many cognitive tasks such as the Test of English as a Foreign Language (TOEFL) synonym test, the grading of content-based essays and the categorisation of groups of concepts (see [19]). However, one problem with this method is scalability: it starts by generating a term x document matrix which grows with the number of terms and the number of documents and will thus become very large for large corpora. For finding the final LSA model, Singular Value Decomposition (SVD) and subsequent dimensionality reduction is commonly used. This technique requires the factorization of the term-document matrix which is computationally costly. Also, calculating the LSA model is not easily and efficiently doable in an incremental or out-of-memory fashion. The Random Indexing (RI) method [18] circumvents these

problems by avoiding the need of matrix factorization in the first place. RI can be seen as an approximation to LSA which is shown to be able to reach similar results (see [24] and [25]). RI can be incrementally updated and also, the term x document matrix does not have to be loaded in memory at once –loading one row at the time is enough for computing context vectors. Instead of starting with the full term x document matrix and then reducing the dimensionality, RI starts by creating almost orthogonal random vectors (index vectors) for each document. This random vector is created by setting a certain number of randomly selected dimensions to either +1 or -1. Each term is represented by a vector (term vector) which is a combination of all index vectors of the document in which it appears. For an object consisting of multiple terms (e.g. a document or a search query with several terms), the vector of the object is the combination of the term vectors of its terms.

In order to apply RI to an RDF graph we first generate a set of documents which represent this graph, by generating one virtual document for each URI in the graph. Then, we generate a semantic index from the virtual documents. This semantic index is then being searched in order to retrieve similar literals/URIs. Virtual documents can be of different depth, and in the simplest case, for a representative URI S, a virtual document of depth one is a set of triples where S is a subject - in addition if any object in the set of triples is a URI we also include all triples where that URI is the subject and the object is a literal. The reason for this is the fact that literals such as labels are often used to describe URIs. A sample virtual document of depth one is shown in Figure 2, where the graph is first expanded down one level from node S. Further on, we also expand the graph from nodes O1 and O2 to include only those statements where objects are literals. A sample raw that will be added to the term x document matrix is illustrated in Table 1.

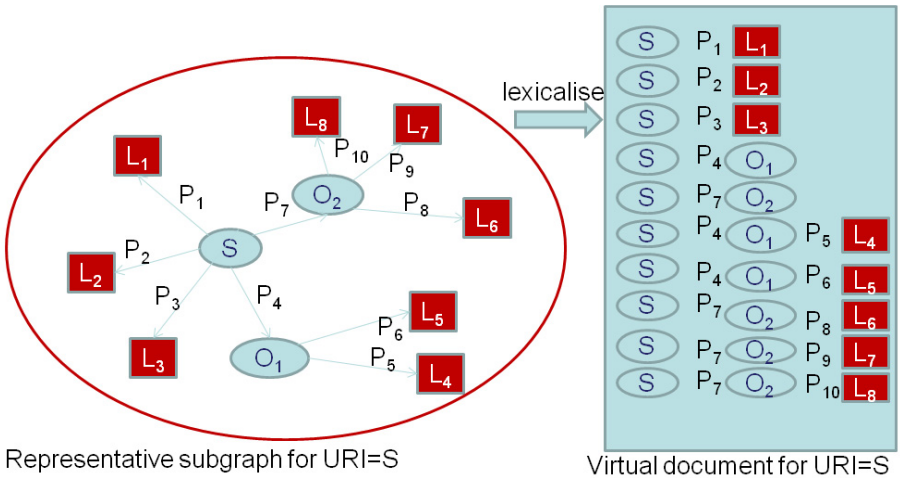


Fig. 2. From a representative subgraph to the virtual document for URI S: L - literals, O - non-literal objects (URIs), P - RDF properties

Table 1. A sample row in the term x document matrix for the virtual document in Figure 2. The number of documents is equal to the number of URIs in the graph, and the number of terms is equal to the number of URIs and literals.

	S	P ₁	..	P ₁₀	L ₁	..	L ₈	O ₁	O ₂
S	10	1	..	1	1	..	1	3	4
..

Traditionally, the semantic index captures the similarity of terms based on their contextual distribution in a large document collection, and the similarity between documents based on the similarities of the terms contained within. By creating a semantic index for an RDF graph, we are able to determine contextual similarities between graph nodes (e.g., URIs and literals) based on their neighbourhood – if the two nodes are related with a similar set of other nodes, they will appear as contextually related according to the semantic index. We use the cosine function to calculate the similarity between the input term (literal or URI) vector and the existing vectors in the generated semantic index (vector space model). While the generated semantic index can be used to calculate similarities between all combinations of term/document-term/document, we focus on document-document search only: suggesting a set of representative URIs related to a set of seed URIs or ICs.

4 Gold Standard-Based Evaluation

In this section we describe the experiments conducted in order to compare our different approaches for concept recommendation. We used 26 real innovation problems from Hypios for which the solutions submitted in the past were available. Our assumption is that a good method for concept recommendation should be able to suggest concepts that appear in the actual solutions. Although the set of concepts appearing in the solutions does not necessarily correspond to the complete set of concepts relevant for solving a problem, it constitutes a reasonable list of concepts against which we can test performance. However, in order to better confirm our results we complement this evaluation with the user study presented in Section 5.

We use 3 **performance measures**: *precision*, *recall* and the combined *F1 measure*. In the sense of this experiment, precision is the number of relevant solution concepts suggested by the system that was found in the actual solutions divided by the number of concept suggestions proposed by the system. By recall we consider the number of relevant solution concepts suggested by the system that was found in the actual solutions divided by the total number of solution concepts known for the particular problem. The F1 score is the harmonic mean between precision and recall. It serves to compare the strategies in the case when precision and recall are equally important, and can point to approaches with the best balance of the two measures.

In order to generate the suggestions using Linked Data-inspired similarity metrics described in Section 3, we used the **DBpedia data set**, as it is arguably the most complete source of topics related to the general human knowledge, with more than 3.5 million concepts. It should be noted that our methods are also applicable to other Linked Data sets. The full DBpedia dataset is also known to have a large number of

properties and hence any structure-based method is expected to be more effective if some pre-selection is conducted prior to calculating similarities. In our case, we were able to select a number of properties relevant to the Open Innovation-related scenario by analyzing the problems and solutions collected on hypios.com in the past (note that this dataset is different from the 26-problems dataset which we used in our evaluation). In order to determine this set of properties we performed DBpedia concept extraction from the text of problems and their respective solutions, using Zemanta. We then queried DBpedia to discover all the paths that connect concepts found in problems with those in the respective solutions. The output of this exercise was only a small number of properties: *dbo:product*, *dbp:prducts*, *dbo:industry*, *dbo:service*, *dbo:genre*, and properties serving to establish a hierarchical categorization of concepts, namely *dc:subject* and *skos:broader*. We therefore boosted the concepts participated in links created over those properties in comparison to the others in DBpedia. The same method for discovering relevant subset of properties could be used to adapt the approach to other domains, provided that an initial set of input concepts and desired outputs is available.

To set up the experiment and create the 'gold standard' against which we can test our methods we prepared the data as follows:

- **Extract problem URIs.** We took the 26 problem descriptions and extracted their key concepts using a natural language processing service that links the key concepts in a given English text to the DBpedia entities. We use Zemanta⁷ for this extraction, but other services such as OpenCalais⁸ or DBpedia Spotlight⁹ may also be used. This service has been shown to perform well for the task of recognizing Linked Data entities from text in recent evaluations [26].
- **Extract solution URIs.** For each problem we collected the submitted solutions (142 total), extracted the key concepts in the same way we did for problem texts.

The key concepts extracted by Zemanta were not verified by human users. While in the case of key concept extraction from problems this verification was feasible, in the case of solutions it was not, as it would violate the confidentiality agreement. We therefore had to work with automatically extracted and non-validated concepts, trusting that Zemanta's error rate would not affect the correctness of our further study, and that the potential impact of potential errors would equally affect all approaches. Note that when evaluating the baseline, we did not need to extract the key concepts, as the Google Keyword tool would generate a set of keywords that we could then compare to the words in the submitted solutions without any need for linking them to URIs. As the **baseline** we used Google Adwords Keyword Tool¹⁰. This tool is a good candidate for baseline because it is the state of the art commercial tool employing some of the best Information Retrieval practices to text. In a legacy platform that Hypios uses for finding solvers, such a tool plays the crucial role as it is capable of suggesting up to 600 similar terms which then can be used to search for solvers. This large number

⁷ developer.zemanta.com

⁸ <http://www.opencalais.com/>

⁹ <http://dbpedia.org/spotlight>

¹⁰ <https://adwords.google.com/select/KeywordToolExternal>

of suggested terms is important for the task of Web crawling in order to find relevant experts. Hypios crawls the Web in order to identify and extract the expert information and thus enrich the existing database of experts. Google Adwords is also widely used in tasks with similar purposes such as placing the adverts for consumers relevant to the page they are viewing. Using the methods for ranking concept recommendations inspired by Linked Data, our aim is to improve the baseline. Our hypothesis is that linked data-based similarity metrics described in this paper can improve the baseline. In what follows we detail the experiments conducted to test this hypothesis.

4.1 Results

We took the key concepts extracted from the problems, and fed them to our methods and to the baseline system, which all generated an independent set of recommended concepts. We then calculated the performance for each method by comparing the results with those collected in the gold standard. The results, shown in Figure 3, indicate that the mixed hyProximity measure performs best with regard to precision. This measure should therefore be used in the end-user applications, as the users can typically consult only a limited number of top-ranked suggestions. With regard to recall, Random Indexing outperforms the other approaches for 200 top-ranked suggestions. It is especially useful in cases when it is possible to consider a large number of suggestions which include false positives - such as the case when the keyword suggestions are used for expert crawling. The balanced F-measure indicates that the transversal hyProximity method might be the best choice when precision and recall are equally important, and for less than 350 suggestions. After this threshold the mixed hyProximity is a better choice. HyProximity measures improve the baseline across all performance measures, while Random indexing improves it only with regard to recall and F-measure for less than 200 suggestions. The significance of differences is confirmed by the T-test for paired values for each two methods ($p < 0.05$).

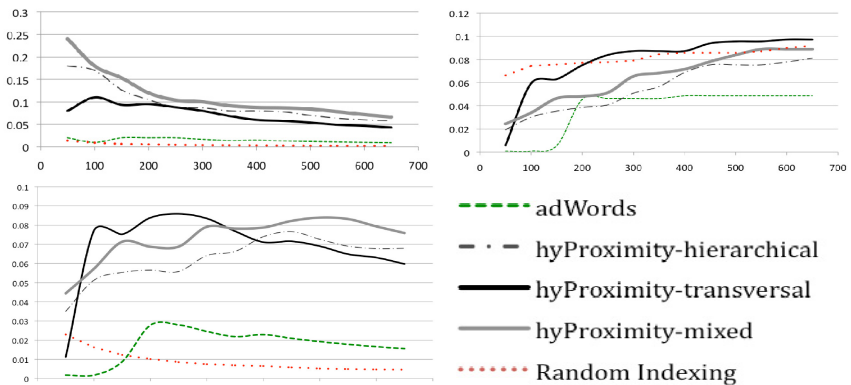


Fig. 3. Comparison of methods: precision (top-left), recall (top-right), F-measure (bottom left). On x axis: the number of suggestions provided by the systems.

The relatively low precision and recall scores for all methods, including the baseline, can be explained by the fact that our ‘gold standard’ is not complete : some concepts might not appear in solutions, even if relevant, as not all relevant experts were motivated to propose a solution. This is a natural consequence of the difficulty of the task. However, our evaluation with such an incomplete dataset still gives an insight into different flavors of our similarity measures, and to compensate for this incompleteness, we conduct a user-centric study in order to test the quality of the generated suggestions.

5 User Evaluation

We conducted a user study in order to cover the aspects of the methods’ performance that could not have been covered by the previous evaluation. The reason is that relying on the solutions received for a particular problem gives insight into a portion of the relevant topics only, as some correct and legitimate solutions might not have been submitted due to the lack of interest in the problem prize, and in such cases our gold standard would not take such topics into account. Further on, the user study allowed a more fine-grained view on the quality of recommendations, as we focused on the following two aspects:

- **Relevancy:** the quality of a concept suggestion being relevant to the given innovation problem in the sense that the concept might lead to a potential solver of a solution of this problem if used in the expert search. We used the scale from 1 to 5: (1) extremely irrelevant (2) irrelevant, (3) not sure (4) relevant (5) extremely relevant.
- **Unexpectedness:** the degree of unexpectedness of a concept suggestion for the user evaluator on the scale from 1 to 5: (1) evident suggestions e.g. those that appear in the problem description (2) easy– suggestions that the user would have easily thought of based on the initial seed concepts (3) neutral (4) unexpected - for keywords that the user would not have thought of in the given context, however the concept is known to him (5) new unexpected - for keywords that were unknown to the user as he had to look up their meaning in a dictionary or encyclopedia.

Suggestions being both relevant and unexpected would represent the most valuable discoveries for the user in the innovation process, and a good concept recommendation system for this use case should be capable of providing such suggestions.

Twelve users familiar with OI scenarios (employees of OI companies and PhD students in OI-related fields) participated in the study. They were asked to choose a subset of innovation problems from the past practice of hypios.com and evaluate the recommended concepts. This generated a total of 34 problem evaluations, consisting of 3060 suggested concepts/keywords. For the chosen innovation problem, the evaluators were presented with the lists of 30 top-ranked suggestions generated by adWords, hyProximity (mixed approach) and Random Indexing. We then asked them to rate the relevancy and unexpectedness of suggestions using the above described scales.

The choice of our subjects was based on the two criteria. Their ability to judge the relevancy in this particular sense came out of their experience with OI problems, and at the same time they were not domain experts, but had rather general knowledge so the topics that they would judge as unexpected would most likely be also unexpected for an average innovation seeker from a client company.

Table 2. Average note \pm standard deviation obtained in the study

Measure	adWords	hyProximity (mixed)	Random Indexing
<i>Relevance</i>	2.930 \pm 0.22	3.693 \pm 0.23	3.330 \pm 0.25
<i>Unexpectedness</i>	2.859 \pm 0.16	2.877 \pm 0.25	3.052 \pm 0.22
<i>Unexpectedness (relevancy \geq4)</i>	2.472 \pm 0.31	2.542 \pm 0.36	2.635 \pm 0.36
<i>Unexpectedness (relevancy =5)</i>	1.760 \pm 0.22	1.842 \pm 0.31	1.767 \pm 0.36

As shown in Table 2, the Linked Data measures outperform the baseline system across all criteria. While hyProximity scores best considering the general relevance of suggestions in isolation, Random Indexing scores best in terms of unexpectedness. With regard to the unexpectedness of the highly relevant results (relevancy \geq 4) Random indexing outperforms the other systems, however hyProximity offers a slightly more unexpected suggestions if we consider only the most relevant results (relevancy=5). We tested the differences in relevance for all methods using the paired T-test over subjects individual means, and the tests indicated that the difference in relevance between each pair is significant ($p < 0.05$). The difference in unexpectedness is significant only in the case of Random Indexing vs. baseline. This demonstrates the real ability of Linked Data-based systems to provide the user with valuable relevant concepts.

In the follow up study, we asked the raters to describe in their own words, the suggestions they were presented with from each system (identified as System 1, 2, and 3). The adjective most commonly used to describe adWords suggestions was “redundant” and “Web-oriented”. This indeed corresponds to the fact that the system is not fully adapted to the OI scenario, but also to the fact that it is based on a statistical approach, which is more influenced by the statistical properties of Web content, than by the meaning of things. HyProximity suggestions were most commonly described as “really interesting” and “OI-oriented”, while the suggestions of Random Indexing were most often characterized as “very general”. According to the preference towards more general or more specific concepts, it is therefore possible to advise the user with regard to which of the two methods is more suitable for the specific use case.

To illustrate the qualitative aspects of suggestions we provided an example of concept suggestions from all 3 systems on our website¹¹.

6 Conclusion

We presented two Linked Data-based concept recommendation methods and evaluated them against the state of the art Information Retrieval approach which served

¹¹ http://research.hypios.com/?page_id=165

as our baseline. We argue that our methods are suitable in an Open Innovation scenario where the suggested concepts are used to find potential solvers for a given problem. Our results show that both proposed methods improve the baseline in different ways, thus suggesting that Linked Data can be a valuable source of knowledge for the task of concept recommendation. The gold standard-based evaluation reveals a superior performance of hyProximity in cases where precision is preferred; Random Indexing performed better in case of recall. In addition, our user study evaluation confirmed the superior performance of Linked Data-based approaches both in terms of relevance and unexpectedness. The unexpectedness of the most relevant results was also higher with the Linked Data-based measures. Users also indicated that Random Indexing provided more general suggestions, while those provided by hyProximity were more granular. Therefore, these two methods can be seen as complementary and in our future work we will consider combining them as their different nature seem to have a potential to improve the properties of the query process.

Acknowledgments. The work of Milan Stankovic is partially funded by the grant CIFRE N 789/2009 given by a French research funding agency ANRT. Special thanks to employees of Open Innovation companies Hypios and Bluenove, as well as the PhD students from the same field from the University Paris Dauphine, for their participation in the evaluation.

References

1. Chesbrough, H.W.: Open Innovation: The New Imperative for Creating and Profiting from Technology. Harvard Business Press (2003)
2. Speidel, K.-P.: Problem-Description in Open Problem-Solving. How to overcome Cognitive and Psychological Roadblocks. In: Sloane, P. (ed.) A Guide to Open Innovation and Crowdsourcing. Advice from Leading Experts. KoganPage, London (2011)
3. Stankovic, M., Jovanovic, J., Laublet, P.: Linked Data Metrics for Flexible Expert Search on the Open Web. 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. 108–123. Springer, Heidelberg (2011)
4. Jeppesen, L.B., Lakhani, K.R.: Marginality and Problem Solving Effectiveness in Broadcast Research. *Organization Science* 20 (2009)
5. Sigurbjörnsson, B., van Zwol, R.: Flickr tag recommendation based on collective knowledge. In: Proceeding of the 17th International Conference on World Wide Web, WWW 2008, p. 327. ACM Press, New York (2008), doi:10.1145/1367497.1367542
6. Mei, Q., Zhou, D., Church, K.: Query suggestion using hitting time. In: Proceeding of the 17th ACM Conference on Information and Knowledge Mining - CIKM 2008, New York, USA, p. 469 (2008)
7. Safar, B., Kefi, H.: OntoRefiner, a user query refinement interface usable for Semantic Web Portals. In: Proceedings of Application of Semantic Web Technologies to Web Communities Workshop, ECAI 2004, pp. 65–79 (2004)
8. Macdonald, C., Ounis, I.: Expertise drift and query expansion in expert search. In: Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management - CIKM 2007, p. 341. ACM Press, New York (2007)
9. Cross, V.: Semantic Relatedness Measures in Ontologies Using Information Content and Fuzzy Set Theory. In: Proc. of the 14th IEEE Int'l Conf. on Fuzzy Systems, pp. 114–119 (2005)

10. Gasevic, D., Zouaq, A., Torniai, C., Jovanovic, J., Hatala, M.: An Approach to Folksonomy-based Ontology Maintenance for Learning Environments. *IEEE Transactions on Learning Technologies* (2011) (in press)
11. Burton-Jones, A., Storey, V.C., Sugumaran, V., Puro, S.: A Heuristic-Based Methodology for Semantic Augmentation of User Queries on the Web. In: Song, I.-Y., Liddle, S.W., Ling, T.-W., Scheuermann, P. (eds.) *ER 2003*. LNCS, vol. 2813, pp. 476–489. Springer, Heidelberg (2003)
12. Ziegler, C.-N., Simon, K., Lausen, G.: Automatic Computation of Semantic Proximity Using Taxonomic Knowledge Categories and Subject Descriptors. In: *Proceedings of the 15th ACM International Conference on Information and Knowledge Management, CIKM 2006*, Arlington, Virginia, USA, pp. 465–474. ACM, New York (2006)
13. Resnik, P.: *Using Information Content to Evaluate Semantic Similarity in a Taxonomy* (1995)
14. Matos, S., Arrais, J.P., Maia-Rodrigues, J., Oliveira, J.L.: Concept-based query expansion for retrieving gene related publications from MEDLINE. *BMC Bioinformatics* (2010)
15. Salton, G., McGill, M.: *Introduction to Modern Information Retrieval*. McGraw-Hill, New York (1983)
16. Cilibrasi, R.L., Vitanyi, P.M.B.: The Google Similarity Distance. *IEEE Transactions on Knowledge and Data Engineering* 19(3), 370–383 (2007), doi:10.1109/TKDE.2007.48
17. Gracia, J., Mena, E.: Web-Based Measure of Semantic Relatedness. In: Bailey, J., Maier, D., Schewe, K.-D., Thalheim, B., Wang, X.S. (eds.) *WISE 2008*. LNCS, vol. 5175, pp. 136–150. Springer, Heidelberg (2008)
18. Sahlgren, M.: An introduction to random indexing. In: *Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005* (2005)
19. Cohen, T., Schvaneveldt, R., Widdows, D.: Reflective random indexing and indirect inference: A scalable method for discovery of implicit connections. *Journal of Biomedical Informatics* (2009)
20. Passant, A.: dbrec — Music Recommendations Using DBpedia. 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. 209–224. Springer, Heidelberg (2010)
21. Waitelonis, J., Sack, H.: Towards Exploratory Video Search Using Linked Data. In: *2009 11th IEEE International Symposium on Multimedia*, pp. 540–545. IEEE (2009), doi:10.1109/ISM.2009.111
22. Stankovic, M., Breitfuss, W., Laublet, P.: Linked-Data Based Suggestion of Relevant Topics. In: *Proceedings of I-SEMANTICS Conference 2011*, Gratz, Austria, September 7-9 (2011)
23. Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R.: Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41, 391–407 (1990)
24. Karlgren, J., Sahlgren, M.: From words to understanding. In: Uesaka, Y., Kanerva, P., Asoh, H. (eds.) *Foundations of Real-World Intelligence*, pp. 294–308. CSLI Publications, Stanford (2001)
25. Cohen, T.: Exploring medline space with random indexing and Pathfinder networks. In: *Annual Symposium Proceedings/AMIA Symposium*, pp. 126–130 (2008)
26. Rizzo, G., Troncy, R.: NERD: Evaluating Named Entity Recognition Tools in the Web of Data. In: *ISWC 2011 Workshop on Web Scale Knowledge Extraction (WEKEX)*, Bonn, Germany (2011)

Finding Co-solvers on Twitter, with a Little Help from Linked Data

Milan Stankovic^{1,3}, Matthew Rowe², and Philippe Laublet³

¹ Hypios Research, 187 rue du Temple, 75003 Paris, France

² Knowledge Media Institute, The Open University, Milton Keynes, UK, MK7 6AA

³ STIH, Université Paris-Sorbonne, 28 rue Serpente, 75006 Paris, France

mail@milstan.net, m.c.rowe@open.ac.uk,

philippe.laublet@paris-sorbonne.fr

Abstract. In this paper we propose a method for suggesting potential collaborators for solving innovation challenges online, based on their competence, similarity of interests and social proximity with the user. We rely on Linked Data to derive a measure of semantic relatedness that we use to enrich both user profiles and innovation problems with additional relevant topics, thereby improving the performance of co-solver recommendation. We evaluate this approach against state of the art methods for query enrichment based on the distribution of topics in user profiles, and demonstrate its usefulness in recommending collaborators that are both complementary in competence and compatible with the user. Our experiments are grounded using data from the social networking service Twitter.com.

Keywords: Linked Data, Twitter, Collaborator Recommendation.

1 Introduction

Modern challenges that science and engineering worlds are facing today are often interdisciplinary and require research cooperation of teams of people in order to produce good solutions. Analysis of tendencies in research publications [1] shows that more and more multi-university teams produce accepted papers. Similarly, industrial innovation challenges often require a collaborative effort of experts from across different disciplines to work together. In this sense, innovation problem solving platforms, such as Innocentive¹, have started to propose problem challenges for teams of problem solvers. Supporting users in the task of forming productive multidisciplinary teams therefore plays an important role in a multitude of innovation-related situations.

Existing social studies [2] on the topic of forming teams investigate people's preferences when it comes to the choice of co-workers, they underline the importance of co-worker similarity (both in terms of shared interests/expertise and in terms of shared social connections) together with the expertise of co-workers given the work task. Conveniently, more and more data about users (i.e. their social connections, topics of interest, their work) is available on the Web, this opens the way for a

¹ <http://www.innocentive.com/>

co-worker recommendation approach that takes into account those different qualities of the user and performs useful recommendations.

In this paper we are addressing the challenge of recommending potential co-solvers to people facing an innovation or research problem for which they are only partially competent. This recommendation is performed based on the data available in social networks (both the user's social graph and the user generated content), by taking into account both the compatibility of candidates with the user, and the complementarity of their competence. The research questions driving our work are: *how can we suggest appropriate co-solvers, which were potentially, previously unknown to the user?* And: *what information can be used to enrich initially available user and problem data to provide the best suggestions?* In exploring these two research questions we have devised an approach to recommend co-solvers for open innovation problems. The contributions from this work are three-fold, and are as follows: (1) Profile Expansion: we present methods for expanding topics that measure semantic relatedness between topics using the linked data graph; (2) Similarity Measures: we describe three similarity measures that exploit either relations between topic concepts or social connections; (3) Evaluation: we assess the performance of our approach when different profile expansion and similarity measures are used through two user studies.

The remainder of the paper is organized as follows: In section 2 we describe the Open Innovation related scenario of use for which we developed our approach and we present related work covering a) the recommendation of co-workers; b) expert finding, and; c) measures of semantic relatedness. Our core approach is presented in section 3 together with different alternatives for several parts of the recommendation process. In section 4 we present two user study-based evaluations of our approach executed over the social network Twitter.com. In section 5 we present our directions of future work and conclude the paper.

2 Background and Related Work

2.1 Open Innovation and the Need for Co-solvers

Open Innovation (OI) [3] has emerged as way to accelerate industrial innovation by looking for diverse and unexpected perspectives to innovation problems outside of the company. OI platforms, such as hypios.com, Innocentive.com, and NineSigma.com have emerged to support this practice. Those platforms allow innovation seekers to post their problems for solving by Web users and award the best solutions. Such innovation platforms aim to allow anyone who has some knowledge or idea to participate in the innovation process. They value diversity in solutions over the acclaim and level of expertise of solvers, as studies have shown that people with marginal competence in the problem's domain, but experts in some other domain, are able to bring more innovative solutions [4], often by transposing ideas from one field to another. Our work on co-solver recommendation, although generalisable to any use case of finding collaborators, is principally oriented at finding people to work with on the Open Innovation problems. This implies a preference towards candidates who are able to bring a novel perspective to solving the problem over those who are acclaimed experts in the exact domain of the problem.

2.2 Recommending People

A number of approaches have been proposed for challenges similar to ours, most notably for recommending people to befriend in social networks. Those recommender systems usually rely on a measure that quantifies the similarity of two users or the similarity of a user profile with a given document or query. Spertus et al. [5] present an empirical comparison of six measures of similarity for recommending communities to members of the Orkut social network. They found the cosine similarity measure to show the best empirical results against other measures. In [6] a transitive notion of user similarity is proposed to address the scarcity of user profiles in collaborative filtering systems when recommending movies. [7] relies only on Jacquard's similarity to compare different sources from which user similarity may be mined (friendships, interests, activities in same places, etc.) in terms of their performance in different scenarios. Another study, presented in [8] shows that algorithms based on a user's social connections are better in recommending known contacts, while those based on the similarity of user-generated content perform better at suggesting new contacts. Those studies however do not provide implications for the use case of recommending co-workers. In addition to similarity, some systems like Facebook, consider affinity towards a user calculated based on the level of mutual interactions on Facebook².

A number of multi-criteria recommender systems have been proposed, mostly using a (linear) combination of different similarity functions to deliver recommendations. For instance a collaborative filtering system for recommending similar users [9] uses a machine learning approach to learn weight factors of different utility functions. [10] also uses a multitude of criteria derived from different interactions users make with Web objects. [11] develops a machine learning approach for leveraging content and social criteria in providing folksonomy-based recommendations, especially on Flickr. However, to our best knowledge such approaches have not been used to suggest co-solvers.

2.3 Expert Finding

A multitude of approaches for expert finding from a range of sources like blogs, scientific literature, corporate social networks, etc. have been proposed in the past – many of which in TREC conferences [12]. Many of such approaches exploit microposts, including tweets, for expert profiling, thereby confirming the viability of this source for co-solvers. For instance, Zoltan and Johan [13] propose a system for the extraction of ontological topic concepts from tweets, topics are then weighted by their importance to the expert profile. Analysis of persistency of topics [14] as well as the awareness of profile dynamics [15,16] have been shown to improve user profiling, over static approaches. Twitter lists have been shown to be a rich source for user profiling [17]. The construction of expert profiles from Twitter data using topic models is also proposed by Wagner [18].

Most of these existing approaches focus on finding people with expertise in a particular topic, while the question of fitting an expert to an innovation task involving

²<http://techcrunch.com/2010/04/22/facebook-edgerank/>

a multitude of topics has not yet been fully explored. To the best of our knowledge, existing approaches do not respond to the needs of OI scenarios, where the requirements in terms of expertise of a potential problem solver are slightly different than those used to select experts in most expert-finding approaches. The necessary focus on getting diverse and laterally relevant experts has, to the best of our knowledge, also not been the focus of the existing expert-finding approaches.

2.4 Measures of Semantic Relatedness Usable for Profile Expansion

Our approach for co-solver recommendation exploits the semantic relatedness of individuals through the linked data graph based on their topical expertise and interests. Measures of semantic relatedness (MSRs) have been proposed for use in a variety of tasks and can be split into two major categories: 1) graph-based measures and; 2) distributional measures.

Graph-based measures make use of semantic (e.g., hyponymy or meronymy) and/or lexical (e.g., synonyms) relationships within a network (graph) of concepts to determine semantic proximity between the concepts. For example, [19] exploits the hypernym graphs of Wordnet³; whereas [20] use the ODP taxonomy⁴ and [21] relies on the graph of Wikipedia categories to provide recommendations. Among graph measures used to calculate the semantic proximity of concepts shortest path is one of the most common, and is often enhanced by taking into account the informational content of nodes in the graph [22]. To the best of our knowledge these approaches have not been applied to knowledge bases of comparable size and richness to that of DBPedia⁵. Even the Wikipedia-based measures that we found only used information about categories and did not leverage other information present in DBPedia. Our own MSR, hyProximity that we will describe shortly, builds upon the existing graph-based measures but is highly adapted to the rich structure of Linked Data sources, as it leverages different types of relations that concepts may have in the graph.

Distributional measures mostly explain semantic relatedness through common use, usually by leveraging co-occurrence of concepts, and mostly do not make connections over the meaning of terms or concepts. For example, the approach presented in [23] uses co-occurrence in text of research papers to establish a notion of word proximity, while others rely on other sources such as search results [24]. In addition to the measures of semantic relatedness, some authors propose the use of pseudo relevance-feedback to enrich an initial set of topics in a query, most notably in the domain of expert search. In particular, pseudo relevance-feedback is used to enlarge expert search queries based on additional keywords appearing in a number of top ranked documents [25] or top ranked user-profiles [25, 26]. The diversity of topics in expert profiles has been shown to negatively impact the quality of results when more narrow results are sought. No evaluation has been provided for the case when broadening of the space of found experts is desired for OI purposes.

³ <http://wordnet.princeton.edu/>

⁴ <http://www.dmoz.org/>

⁵ While DBPedia contains more than 3.5 million concepts, the current version of Wordnet has 206941 word-sense pairs, and ODP has half a million categories.

3 Recommending Co-solvers

Our general approach for recommending co-solvers is based on a user's social connections and/or the content he/she created. The general process is applicable on any social network with user-generated content, however we discuss our concrete implementation of the process on data from the social network Twitter.

3.1 General Approach

In our general approach, a Web user (called a **seed user**) approaches our system with the intention to find potential collaborators for a particular research challenge or innovation problem (called **problem** hereafter). He provides the text of the problem/challenge and gives some identifier that he uses on a social networking system, this allows access to: (1) his social connections and (2) some content that he created. The system then proceeds with the creation of profiles for both the user and the problem. Those profiles contain Linked Data identifiers of topics extracted from the provided textual elements (from the text of the problem/challenge in the case of the Problem Profile or from the content that the user has created in the case of the User Profile). Optionally, an additional phase of profile enrichment may be performed (called **Profile Expansion**). This functions by expanding the initial profiles in order to broaden their topics and thus compensate for any incompleteness. – i.e. where the topics may be too specific. Similarity scoring is performed over a base of candidate user profiles in order to select those candidate users that: (1) are the most similar to the seed user and; (2) whose profile fits the given innovation problem. Similarity scoring can work both with the initial user and problem profile as well as with the extended ones. Particular similarity functions will be further discussed in Section 3.3.

3.2 Profiling

In the profiling phase, user and problem profiles are created from the provided textual elements (posts and biography in the case of user profiles and problem text in the case of problem profiles). The topic profiles (denoted TP in equation (1)), regardless of the type of entity (user or problem) that they concern, are sets of tuples assembled from a Linked Data URI of a topic concept and a value w representing the importance of this particular topic for the profile. In essence, this topic profile is a concept vector, where each element's index represents a unique concept. The value stored in a given element is the frequency of the concept's appearance in either: a) the user's past information or; b) the problem definition. In the phase of profile expansion, the values w of additional related topics correspond to the relatedness of the topic to the given profile, as expressed by the particular measure of semantic relatedness used to perform the profile expansion.

$$TP(entity) = \{(URI_1, w_1), \dots, (URI_n, w_n)\} \quad (1) \quad SP(user) = \{user_1, \dots, user_n\} \quad (2)$$

Different operations are possible over the topic profiles. For instance, in our work we will rely on the difference $TP(\text{problem})-TP(\text{user})$, called **difference topics**, that represents the topics found in the problem topic profile that are not found in the topic profile of the seed user - this derives the topics for which there is no record of seed user's past knowledge. In addition to topic profiles, social profiles (denoted SP in equation (2)) are also created for the users, and they contain the list of user's social connections:

3.2.1 Extraction of Topics

Extraction of topics from text plays a crucial role in the profiling phase. When we refer to topics in our work, we consider concepts of general human knowledge that have unique identifiers – e.g. a URI - and are described as resources somewhere in the Linked Data cloud. Descriptions of such concepts may be found, for instance, in knowledge bases such as Freebase.org or DBpedia.org. A number of entity recognition services propose the extraction of such concepts from given text, such as Zemanta⁶, OpenCalais⁷, DBpedia Spotlight⁸ etc. In our experiments we use Zemanta for topic extraction. This service has been shown to perform well (among the best) for the task of recognizing Linked Data entities from text in recent evaluations [27].

3.2.2 User Profiling on Twitter

In our particular implementation on Twitter, we use the user's biography and a number of last tweets (300 used in particular experiments) to extract the topics. The DBpedia URIs of extracted topics are stored in TP along with the frequency of their occurrence. This approach assures that the most recent interest topics are taken into account. On the other hand it is not very restrictive, as any topic tweeted about may be considered as part of a user's profile. Our approach favours the broader view of topics in accordance with the need for laterality and inclusion of people with borderline interest/expertise in a particular topic, who might bring innovative perspectives to the problem solving process – essential for our Open Innovation scenario. In cases where it is necessary to assure that only topics for which the user is really experienced about are represented, it is possible to use a more restrictive approach such as one of those proposed in [14], [15], and [16], mostly making use of the dynamics and persistence of topics in user's tweets.

3.2.3 Creating Candidate User Profiles

Ideally, candidate users should be all the users of the Web. While it is theoretically possible to construct such a base of user profiles, most real world systems are confronted with constraints in terms of both access to user data and processing time, and thus have to restrict themselves to a set of candidate user profiles, most likely to be selected by the recommender system. In our case, for each seed user and problem, we perform searches on the raw social network data. In particular we perform two types of search: (1) to find all the users in the social proximity of the seed user, i.e. friends of friends, and (2) to find all the users corresponding to the topics found in the problem

⁶ <http://developer.zemanta.com>

⁷ <http://www.opencalais.com/>

⁸ <http://dbpedia.org/spotlight>

and the seed user profile, as found by the search function of the particular social network used – in the case of Twitter using their built-in search functionality. Users found by those different queries constitute a base of candidate user profiles for every recommendation. These two particular ways of harvesting candidate users correspond to the general intention of finding people similar to the seed user (in the sense of interests and social connections) and relevant for the topics of the problem. Users that our seed user are already friends with are eliminated from the possible recommendations as we assume that they are known to the seed user and as such would not represent valuable discoveries. In cases where they are considered relevant, it is relatively easy to tweak the system to also include the user’s friends in the recommendations.

3.3 Core Similarity Measures and Similarity Scoring

In accordance to our criteria described in the introduction, the ranking of the candidate users should favour the candidates: whose profile topics are similar to the topics in the seed user’s profile (interest similarity); whose social connections are similar to the social connections of the seed user (social similarity) and whose profile topics are similar to the difference topics (similarity with the difference topics). In order to enforce such a ranking we use measures of similarity that enable the measurement of each of the abovementioned aspects. We also use a combined measure in order to favour candidates who satisfy all of the above criteria. All the similarity measures operate over TP and SP vectors defined in §3.2. As these measures take vectors as input we can apply two well established functions for vector similarity Weighted Overlap and Cosine Similarity:

- **Weighted Overlap** represents the sum of weights in the seed user’s profile, for all topics that the seed and candidate user profiles have in common. The use of this simple measure can be observed in many approaches, like [28].
- **Cosine similarity** measure is another commonly used vector similarity measure [29, 30] and it is defined as the cosine of the angle of the two vectors to be compared.

We define three measures of similarity, each of which are used with the above functions:

- **Interest Similarity** – The similarity of topic profiles TP of the seed user and a candidate user. When applied in this way we will refer to our functions with a suffix *t*, e.g., WeightedOverlap_{*t*}, Cosine_{*t*};
- **Social Similarity** – The similarity of social connection profiles SP of the seed user and the candidate user. When used in this way, we will refer to our functions with a suffix *s*, e.g., WeightedOverlap_{*s*}, Cosine_{*s*};
- **Similarity with difference topics** – The similarity between the topic profile TP of a candidate user and the vector of difference topics. When used in this way, we will refer to our functions with a suffix *dt*, e.g., WeightedOverlap_{*dt*}, Cosine_{*dt*}.

In order to aggregate the values of all 3 different similarity measures we use a composite similarity measure. Given that the values of the elementary similarity measures are in the range [0,1], and that they are normalized to this range, we use the

product (PC) of the three elementary measures as the aggregate measure. For instance $PC(Cosine_t, Cosine_s, Cosine_{dt}) = Cosine_t \cdot Cosine_s \cdot Cosine_{dt}$. Alternatively it is possible to use the sum of weighted values of elementary similarity measures, in which case the weights may be adjusted by a machine-learning approach [9] in order to adapt to the preference of each user. The PC measure, as opposed to any linear combination of elementary functions, penalizes the candidates that rank extremely poorly at any single similarity function ($0 \cdot x = 0$), regardless of the high ranking at another function. The candidates ranked highly at only one similarity function could therefore not be ranked better than those being similar in all required aspects.

3.4 Profile Expansion Using Semantic Relatedness Measures

Prior to calculation of similarity measures it is possible to enrich the seed user profile as well as the problem profile with additional topics that are related to the topics initially found in their topic vectors. Figure 2 shows a graph through which the seed user and the problem are linked to the potential candidate co-solvers. While the topics found initially in the seed user profile (T1 and T2) and in the problem profile (topics T1, T3, T4 and T5) do lead to some potential co-solvers, it is possible to consider the semantic relatedness of topic concepts and in that way reach a larger set of candidate users and in general have a richer view of the similarity of profiles. By semantic enrichment of profiles we consider adding topics to an initial profile that are found to be related to the initial profile’s topics according to a particular notion of semantic relatedness. This notion of semantic relatedness of topics may be derived in different ways. In this section we present 3 of such different measures: a) distributional semantic relatedness measure; b) hyProximity – a Linked Data-based measure of semantic relatedness – and; c) pseudo relevance-feedback.

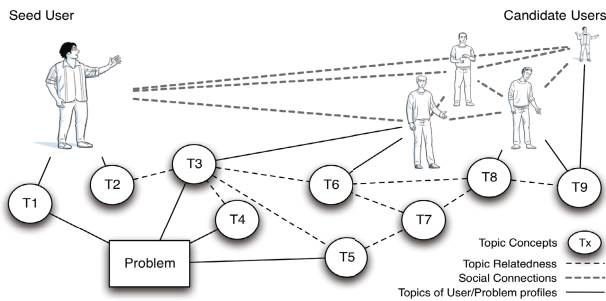


Fig. 1. The Graph of Social and Semantic Connections

The first step in profile enrichment is to generate relevant concepts (according to the measure used) for each topic present in the initial profile. This expansion is performed by first identifying related concepts to the original profile topics – e.g. T3 in Figure 1 when considering the seed user profile. The relatedness is then calculated between each profile topic and related concept – e.g. T2 with T3, and T1 with T3 – to derive individual relatedness values. The total relatedness between the profile and a given related concept is then given by the sum of these individual calculations, where

the greater the cumulative relatedness the greater the relatedness between the profile and the related concept. The top- n topics are selected based on their cumulative relatedness – we set $n=30$ for our experiments.

3.4.1 Distributional Measure of Semantic Relatedness (DMSR)

$$DMSR_{\tau}(t_1, t_2) = \frac{ocurrence(t_1, t_2)}{ocurrence(t_1) + ocurrence(t_2)} \quad (3)$$

The distributional measure of semantic relatedness (DMSR) relies on co-occurrence of topics in profiles to conclude that two topics are similar. It is inspired from the distributional measures used in recommending keywords for search query term suggestion [24]. DMSR, expressed with the formula (3) considers two topics to be similar if they co-occur often in profiles. It represents the ratio of the number of joint occurrences of two topics t_1 and t_2 in user profiles and the sum of their individual occurrences. It is calculated over τ - the set of all user profiles taken into account as potential user candidates.

3.4.2 hyProximity (HPSR)

HyProximity is a measure of semantic relatedness (HPSR) that relies on Linked Data graphs of topic concepts. We have developed this notion for the purposes of discovering laterally relevant topics for solver finding purposes. While only a short summary of the notion is presented here, our previous paper [31] provides more detailed descriptions together with corresponding algorithms and evaluations.

In data sets rich with topics of general human knowledge, such as DBpedia and Freebase, concepts are usually related using properties that might be classified in two main types:

- **Hierarchical links** - Those created over properties that help organize the concepts in classes (e.g., `rdf:type`⁹ and `rdfs:subclassOf`) or categories (e.g., `dcterms:subject` and `skos:broader`). The links created by those properties connect a concept to a *category* – the one serving to organize other concepts into classes.
- **Transversal links** – Those created between ordinary, non-category concepts over properties that connect concepts without the aim to establish a classification or hierarchy. Those properties might create explicit links (connecting two concepts directly) or implicit links (when two concepts have the same value for the same property). Through the analysis of concepts appearing in past open innovation problems and their solutions received on `hypios.com`, we have discovered that only a small set of properties participate in forming all of the links between concepts observed in OI scenarios. This set of properties, referred to as P , consists of `dbo:product`, `dbo:industry`, `dbo:service`, `dbo:product` and `dbp:products`.

$$HPSR(t_1, t_2) = \sum_{C_i \in \mathcal{C}(t_1, t_2)} ic(C_i) + \sum_{p \in P} link(p, t_1, t_2) \cdot pond(p, t_1) \quad (4)$$

Our approach uses the links created over these two types of properties in different ways, appropriate to the nature of those links. According to the equation (4) the value

⁹ All the prefixes used in this paper can be looked up at <http://prefix.cc>

of our HPSR measure for two topics t_1 and t_2 is the sum of valorisations of the connections achieved over hierarchical links (first component of the formula) and those achieved over transversal links (second component of the formula). In the treatment of hierarchical links we take all the common categories $C(t_1, t_2)$ of the two topics, and then for each common category C_i we count the informational content [22] of this category as $-\log(pb)$ where pb is the probability of finding the category in the graph of DBpedia categories when going from the bottom up. The sum of values for all common categories represents the strength of links established between t_1 and t_2 over the hierarchical properties. The transversal links are treated slightly differently. For each property p , from the previously defined set of relevant properties P , we count the number of links connecting t_1 and t_2 over the property p (given by function $link(p, t_1, t_2)$) and weight them using weighting function $pond(p, t_1)$. The value of the weighting function is calculated as $-\log(n/M)$, where n is the number of other concepts to which the concept t_1 is connected over the same property p ; and M is the large constant larger than any possible value of n (in our case it is the total number of concepts in DbPedia).

As our formula is not symmetric, i.e., $HPSR(a,b)$ is not equal to $HPSR(b,a)$, it is always calculated by putting the topics that belong to the seed user or to the difference topics as the first parameter, and the topics from the candidate user profiles as the second parameter.

3.4.3 Pseudo Relevance-Feedback (PRF)

Pseudo relevance-feedback (PRF) is a technique used to enlarge search queries, in which a number of best-ranked profile suggestions/results from a search or recommendation system is taken and the co-occurrence analysis is performed on them to discover the topics that appear frequently in the results, there are then used to enlarge the initial topics and re-run the recommendation process. We mentioned some of the examples of the use of pseudo relevance-feedback in the Background section. Computation wise, the measure of relatedness based on pseudo relevance feedback (denoted PRF) is calculated by the same formula (3) as DMSR with the exception that only a number (10 in our case) of the initially best ranked candidates are used as τ instead of considering the whole set of potential candidates.

4 Evaluation

In order to evaluate the performance of different similarity measures and approaches to profile expansion when providing co-solver recommendations we performed two experiments involving Twitter users. Recent studies show the growth of scholarly Twitter accounts¹⁰ and its use in communication in scientific communities, especially the Computer Science research community [32], thus making Twitter resourceful for co-solvers recommendations. We first created three multidisciplinary innovation problems¹¹, inspired from descriptions of existing research challenges and projects that we found online, each involving the topics related to Semantic Web and to at least one

¹⁰ Third of all scholars are said to have a Twitter account today

<http://www.scribd.com/doc/37621209/2010-Twitter-Survey-Report>

¹¹ Available on our website http://research.hypios.com/?page_id=184

other domain (in particular: Sensors, Online Advertising and Venture Capital Investments). We then used different alternatives of our method to suggest possible collaborators corresponding to our raters, by relying on candidate user profiles created according to our approach described in 3.2.2 and 3.2.3. In the first experiment (§4.1) we used a gold standard obtained from 3 raters and then assessed the performance of different permutations of profile expansion methods with the interest similarity measure and the difference topic measure. We omitted social similarity from this stage due to the differences in the social networks of the 3 solvers – as each solver has different potential candidates – the gold standard in this case uses the intersection of candidates recommended to each solver. In the second experiment (§4.2) we evaluated all profile expansion methods with all ranking methods using a group of 12 raters. In this case we did not take interrater agreement for the gold standard, but instead evaluated performance on an individual basis. We were therefore able to include social similarity as a ranking technique and evaluate its performance. Performing these two studies allows the comparison between performance of different profile expansion and similarity measures when a) recommending co-solvers to a group of users – in the case of experiment 1, and; b) recommending co-solvers to individual users in experiment 2. To gauge the performance of different permutations of profile expansion and similarity measures we used the following evaluation metrics:

Discounted Cumulative Gain. (DCG) quantifies the value of items in the top- n ranked suggestions as well as the quality of their ranking. For each ranking resulting from a particular ranking alternative, we take the 10 best-ranked user candidates and look at the ratings users generated for them. If the user candidate found at position i is rated positively by users we take *rating* to be 1, otherwise we consider it being equal to 0. The importance of positively ranked candidates found on lower positions in a particular ranking is downgraded logarithmically in order to favour ranking alternatives that put the best candidates towards the top.

Average Precision. (AveP_n) computes the average value of precision as a function of recall, on the interval recall $\in [0,1]$. For each position in a concrete ranking we calculate the precision up to that point, and the gain in recall achieved at that point as opposed to the previous point. The product of those gives an idea of the value a user would gain by looking at the suggestions up to a particular rank.

$$DCG = rating_1 + \sum_{i=2}^{10} \frac{rating_i}{\log_2 i} \quad (5) \quad AveP_n = \sum_{i=1}^n precision(i) \cdot \Delta recall(i) \quad (6)$$

4.1 Evaluation 1

In our first evaluation, we approached a group of 3 researchers from the field of the Semantic Web and presented them with our 3 problems for which they were collectively, as a group, only partially competent. For each rater we generated co-solver suggestions using different combinations of similarity measures and profile expansion approaches. We then took the top-10 suggestions from each different

method and mixed all the suggestions together in one randomized list. This resulted in reasonably sized lists (i.e., 30-50), as some users were recommended by several methods, but on the other hand limited the possibilities of evaluation to the methods defined prior to user rating. The raters then rated candidates by answering if they would work with the suggested user on the given innovation problem. Raters were instructed to base their ratings on a holistic perception of suitability of the suggested user, and only positively rate the users who are both competent and who seem to be potential co-workers. Prior to calculation of any performance measures we calculated the inter-rater agreement using the kappa statistic defined in [33] for each pair of raters. The value of k was, at first, inferior to the threshold of 0.6 for some of the rater pairs. We then allowed the raters to anonymously see the ratings of other group members and alter their ratings if they felt the others were right. After the second round the inter-rater agreement was superior to 0.6 for all 3 problems and for all problems (0.74 on average). We then used the majority agreement between raters to derive our decision labels for each candidate user (i.e., recommended or not recommended).

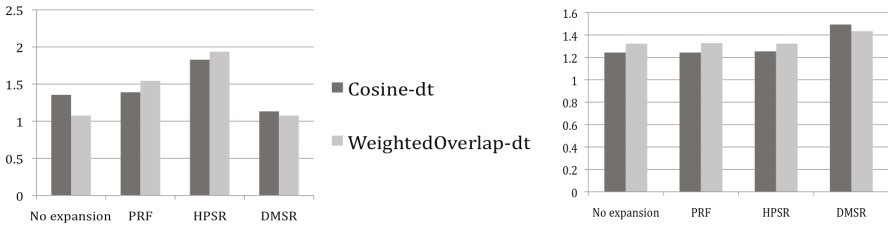


Fig. 2. DCG of rankings obtained with different methods of expansion applied to problem profiles (right) and to seed user profiles (left)

DCG values for similarity measures based on Cosine and Weighted Overlap functions run with all three expansions methods are shown on Figure 2. In the case of rankings based on the similarity with the difference topics, it is clear that the HPSR method of profile enrichment dominates the other expansion methods. This method is much less efficient when it comes to ranking based on the interest similarity of candidate users to the seed users, where DMSR slightly outperforms the other methods, with a little improvement over the standard approach. The figure shows average values over all 3 problems, and the differences in method performance have been confirmed to be significant by the T test ($p < 0.05$) for HPSR with DMSR in similarity with difference topics, but not in the case of interest similarity. It should be noted that our expansion methods are not applicable to the calculation of social similarity as this measure relies on SP vectors that contain no topics. It is indeed reasonable to expect that distributional measures, based on the distribution of topics in user profiles would work well on user-to-user similarity. The enrichment of problem topics using Linked Data-based measures, on the other hand, has already been shown to perform well in keyword suggestion scenarios [31] and it is reasonable to expect that an enrichment based on the meaning of topics would allow better mapping of the problem's conceptual space and reach users whose profiles have a more complete coverage of this space.

4.2 Evaluation 2

In our second evaluation, we solicited ratings from 12 individual Twitter users, experts in the field of Semantic Web. Similar to the previous study, we provided them with the list of candidate co-solvers and asked them to select those that they would work with on a given problem, for which they were only partially competent. The same 3 problems were used by each rater, thereby resulting in 36 sets of rated suggestions. This time, in order to generate a more reusable gold standard, we asked the raters to evaluate all the possible user candidates we collected whose profiles contained at least one of the difference topics. Each set of suggestions for each problem-user pair contained 80-240 suggestions to evaluate. This was time-consuming for raters but resulted in a gold standard that covered virtually all candidates that could be recommended by any variation of our approach. Such a gold standard allowed us to perform additional comparisons of methods, and especially focus on composite similarity measures that were not the subject of the first study.

Ranked candidate lists were generated using the following combinations of similarity functions and profile expansion methods:

- $PC(\text{Cosine}_s, \text{Cosine}_{dt}, \text{Cosine}_t)$: composite function that is a product of interest, social and the similarity with difference topics counted using cosine similarity.
- $PRF(PC(\text{Cosine}_s, \text{Cosine}_{dt}, \text{Cosine}_t))$: PRF problem profile expansion with composite similarity.
- $PC(\text{Cosine}_s, \text{HPSR}(\text{Cosine}_{dt}), \text{Cosine}_t)$: HPSR expansion performed on difference topics prior to calculating the similarity with difference topics.
- $PC(\text{Cosine}_s, \text{Cosine}_{dt}, \text{DMSR}(\text{Cosine}_t))$: DMSR expansion performed over the seed user profile prior to calculating interest similarity.
- $PC(\text{Cosine}_s, \text{HPSR}(\text{Cosine}_{dt}), \text{DMSR}(\text{Cosine}_t))$: composite function in which HPSR is used to expand profile topics and DMSR to expand seed user topic profile prior to calculating the similarities.

In the above user study we described the results obtained when using hyProximity and Distributional profile expansion measures over the elementary similarity functions. For brevity, in this section, we omit these results from the second study and concentrate on the remaining permutations and their combinations using a composite similarity function – something that was not possible in the first study. We focus on composite measures as they allow us to gain a more complete insight in the impact of profile expansion on our multi-criteria recommendation task as a whole. As shown on Figure 3, according to the DCG measure for the first 10 ranked suggestions, the approaches with topic enrichment by either PRF, HPSR or DMSR consistently show better results than the basic approach, on all 3 problems. The overall values are on an expected level for the relevance scale used. HPSR performs slightly better than the other methods in most cases. However the mixed aggregate function (where HPSR is applied to the enrichment of problems and DMSR to the enrichment of user profiles) mostly gives lower results than the individual enrichment approaches. The cause of this might simply be that expanding both problem and seed user profiles induces too much of a difference with regards to the input data and might divert the co-solver search to a non-desired direction. The results shown on the Figure 4 represent the case when the Cosine similarity function is used. When the Weighted Overlap is used, results show negligible differences with the order of best alternatives unchanged, and are omitted for brevity reasons.

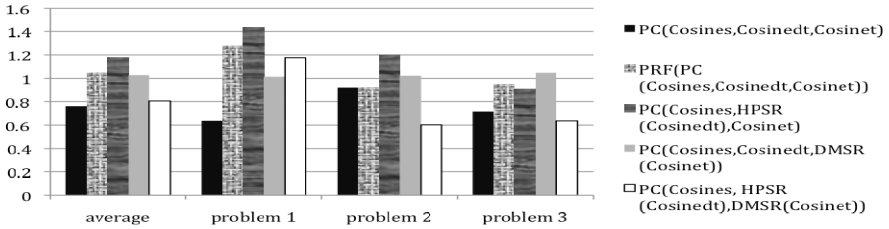


Fig. 3. Average DCG for all raters for different alternatives of composite similarity functions

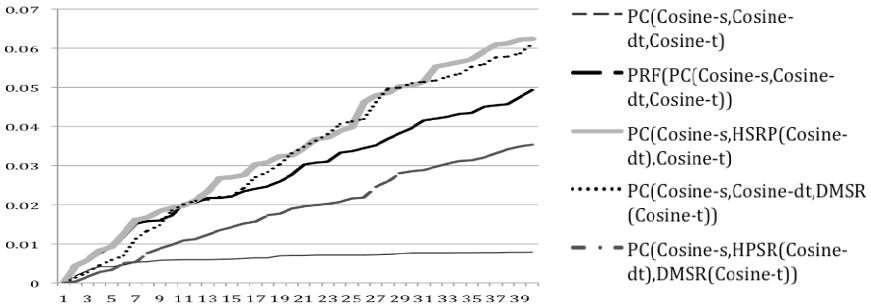


Fig. 4. AvePn of composite approaches (y axis), counted at different rank positions from 1-40 (x axis). Better approaches reach higher AveP at lower ranks.

Similar results are observed with the Average Precision used as the performance metric (Figure 4). It shows that even on a larger set of best ranked candidates (40) the individual expansion methods dominate the mixed one. All the expansion methods also dominate the basic approach. The methods that gain higher values of AveP at lower numbers of rank positions are the ones that give more valuable suggestions at higher ranks and allow the user to discover valuable collaborators while going through a lower number of suggestions. In this case, HSPR enrichment has slightly better results than the other methods. In order to give a better insight into the usefulness of the results generated with our approach we provide an example of co-solver suggestions on our website¹².

5 Conclusions and Future Work

In this paper we proposed an approach for suggesting co-solvers to a person engaged in solving a problem (industrial or research challenge) that has only partial competence and is looking for a complementary and compatible collaborator. Our work explored two central questions: *how can we suggest appropriate co-solvers, which were potentially, previously unknown to the user?* And: *what information can be used to enrich initially available user and problem data to provide the best*

¹² http://research.hypios.com/?page_id=184

suggestions? In addition to standard profiling techniques aimed at creating profiles with topic concepts referring to resources defined in the Linked Data Cloud, we proposed ways for expanding the profiles with additional relevant profiles in order to improve the final co-solver suggestions based on commonly used similarity methods. Through two user studies we have demonstrated that our Linked Data-based measure of semantic relatedness HPSR performs better than state of the art measures when applied with the expansion of problem profiles, while a commonly used measure based on the distribution of topics in user profiles performs better when it comes to expanding the seed user profile. When applied to composite similarity measures that reflect the social and interest similarity of candidate users (suggested co-solvers) with the seed user (user requiring help), as well as their complementarity with regards to the problem, all of the profile expansion methods outperform the simple approach, while HPSR used to expand problem profiles is slightly better than the others. Combined expansion of both problem and seed user's profiles does not outperform individual expansions. One of the future work directions will be to experiment with the graph-based measures of similarity, in addition to the vector measures presented in this paper.

Acknowledgments. The work of Milan Stankovic has been partially funded by ANRT under the grant number CIFRE N 789/2009. The work of Matthew Rowe was supported by the EU-FP7 projects WeGov (grant no. 248512) and Robust (grant no. 257859).

References

1. Jones, B.F., Wuchty, S., Uzzi, B.: Multi-university research teams: shifting impact, geography, and stratification in science. *Science* 322(5905), 1259–1262 (2008)
2. Hinds, P.J., Carley, K.M., Krackhardt, D., Wholey, D.: Choosing Work Group Members: Balancing Similarity, Competence, and Familiarity. *1. Organizational Behavior and Human Decision Processes* 81(2), 226–251 (2000)
3. Chesbrough, H.W.: *Open Innovation: The New Imperative for Creating and Profiting from Technology*. Harvard Business Press (2003)
4. Jeppesen, L.B., Lakhani, K.R.: Marginality and Problem Solving Effectiveness in Broadcast Research. *Organization Science* 20 (2009)
5. Spertus, E., Sahami, M., Buyukkocuten, O.: Evaluating similarity measures. In: *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, New York, USA (2005)
6. Luo, H., Niu, C., Shen, R., Ullrich, C.: A collaborative filtering framework based on both local user similarity and global user similarity. *Machine Learning* 72(3), 231–245 (2008)
7. Guy, I., Jacovi, M., Perer, A., Ronen, I., Uziel, E.: Same places, same things, same people?: mining user similarity on social media. In: *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*, pp. 41–50. ACM (2010)
8. Chen, J., Geyer, W., Dugan, C., Muller, M., Guy, I.: Make new friends, but keep the old: recommending people on social networking sites. In: *Proceedings of the 27th International Conference on Human Factors in Computing Systems*. ACM (2009)
9. Lakiotaki, K., Matsatsinis, N., Tsoukias, A.: Multicriteria User Modeling in Recommender Systems. *IEEE Intelligent Systems* 26(2), 64–76 (2011)

10. Kazienko, P., Musial, K., Kajdanowicz, T.: Multidimensional Social Network in the Social Recommender System. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 41(4), 746–759 (2011)
11. Siersdorfer, S., Sizov, S.: Social recommender systems for web 2.0 folksonomies. In: *Proceedings of the 20th ACM Conference on Hypertext and Hypermedia, HT 2009*, p. 261. ACM Press, New York (2009)
12. *Text Retrieval Conference Proceedings (1992-2010)*
13. Zoltan, K., Johann, S.: Semantic analysis of microposts for efficient people to people interactions. In: *10th Roedunet International Conference, RoEduNet 2011*, pp. 1–4, 23–25 (2011)
14. Ziaimatin, H.: DC Proposal: Capturing Knowledge Evolution and Expertise in Community-Driven Knowledge Curation Platforms. 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. 381–388. Springer, Heidelberg (2011)
15. Abel, F., Gao, Q., Houben, G.J., Tao, K.: Analyzing Temporal Dynamics in Twitter Profiles for Personalized Recommendations in the Social Web. In: *Web Science Conference, Koblenz (2011)*
16. Stan, J., Do, V.-H., Maret, P.: Semantic User Interaction Profiles for Better People Recommendation. In: *Advances in Social Networks Analysis and Mining, ASONAM 2001 (2011)*; Rowe, M., Angeletou, S., Alani, H.: Predicting Discussions on the Social Semantic Web. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) *ESWC 201. LNCS*, vol. 6644, pp. 405–420. Springer, Heidelberg (2011)
17. Wagner, C.: Exploring the Wisdom of the Tweets: Towards Knowledge Acquisition from Social Awareness Streams. 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. 493–497. Springer, Heidelberg (2010)
18. Burton-Jones, A., Storey, V.C., Sugumaran, V., Puro, S.: A Heuristic-Based Methodology for Semantic Augmentation of User Queries on the Web. In: Song, I.-Y., Liddle, S.W., Ling, T.-W., Scheuermann, P. (eds.) *ER 2003. LNCS*, vol. 2813, pp. 476–489. Springer, Heidelberg (2003)
19. Ziegler, C.-N., Simon, K., Lausen, G.: Automatic Computation of Semantic Proximity Using Taxonomic Knowledge Categories and Subject Descriptors. In: *Proceedings of the 15th ACM International Conference on Information and Knowledge Management, CIKM 2006, Arlington, Virginia, USA*, pp. 465–474. ACM, New York (2006)
20. Strube, M., Ponzetto, S.P.: WikiRelate! Computing semantic relatedness using Wikipedia. In: *Proceedings of the National Conference on Artificial Intelligence*, vol. 21, p. 1419. AAAI Press, MIT Press, Menlo Park, Cambridge (1996, 2006)
21. Resnik, P.: Using Information Content to Evaluate Semantic Similarity in a Taxonomy (1995)
22. Matos, S., Arrais, J.P., Maia-Rodrigues, J., Oliveira, J.L.: Concept-based query expansion for retrieving gene related publications from Medline. *BMC Bioinformatics* 11, 212 (2010)
23. Cilibrasi, R.L., Vitanyi, P.M.B.: The Google Similarity Distance. *IEEE Transactions on Knowledge and Data Engineering* 19(3), 370–383 (2007), doi:10.1109/TKDE.2007.48
24. Macdonald, C., Ounis, I.: Expertise drift and query expansion in expert search. In: *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management - CIKM 2007*, vol. 341. ACM Press, New York (2007)

25. Serdyukov, P., Chernov, S., Nejdl, W.: Enhancing Expert Search Through Query Modeling. In: Amati, G., Carpineto, C., Romano, G. (eds.) ECIR 2007. LNCS, vol. 4425, pp. 737–740. Springer, Heidelberg (2007)
26. Rizzo, G., Troncy, R.: NERD: Evaluating Named Entity Recognition Tools in the Web of Data. In: Proceedings of the 11th International Semantic Web Conference 2011, Bonn, Germany (2011)
27. Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., Ma, W.-Y.: Mining user similarity based on location history. In: Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS 2008, p. 1. ACM Press, New York (2008)
28. Balog, K., de Rijke, M.: Finding similar experts. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2007, p. 821. ACM Press, New York (2007), doi:10.1145/1277741.1277926
29. Viswanathan, K.K., Finin, T.: Text Based Similarity Metrics and Deltas for Semantic Web Graphs. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., et al. (eds.) Proceedings of the 9th International Semantic Web Conference - ISWC 2010, Shanghai, China (2010)
30. Stankovic, M., Breitfuss, W., Laublet, P.: Linked-Data Based Suggestion of Relevant Topics. In: Proceedings of I-SEMANTICS Conference 2011, Gratz, Austria, September 7-9 (2011)
31. Letierce, J., Passant, A., Decker, S., Breslin, J.: Understanding how Twitter is used to spread scientific messages. In: Proceedings of the WebSci 2010, Raleigh, NC, US (2010)
32. Fleiss, J.: Statistical Methods for Rates and Proportions. Wiley-Interscience (1981)

Top- k Linked Data Query Processing

Andreas Wagner, Thanh Tran Duc, Günter Ladwig,
Andreas Harth, and Rudi Studer

Institute AIFB, Karlsruhe Institute of Technology, Germany
{a.wagner, ducthanh.tran, guenter.ladwig, harth, studer}@kit.edu

Abstract. In recent years, top- k query processing has attracted much attention in large-scale scenarios, where computing only the k “best” results is often sufficient. One line of research targets the so-called *top- k join* problem, where the k best final results are obtained through joining partial results. In this paper, we study the top- k join problem in a Linked Data setting, where partial results are located at different sources and can only be accessed via URI lookups. We show how existing work on top- k join processing can be adapted to the Linked Data setting. Further, we elaborate on strategies for a better estimation of scores of unprocessed join results (to obtain *tighter bounds* for early termination) and for an *aggressive pruning* of partial results. Based on experiments on real-world Linked Data, we show that the proposed top- k join processing technique substantially improves runtime performance.

1 Introduction

In recent years, the amount of Linked Data has increased rapidly. According to the Linked Data principles¹, dereferencing a Linked Data URI via HTTP should return a machine-readable description of the entity identified by the URI. Each URI therefore represents a virtual “data source” (see Fig. 1).

In this context, researchers have studied the problem of Linked Data query processing [3, 5, 6, 10, 11, 16]. Processing structured queries over Linked Data can be seen as a special case of federated query processing. However, instead of relying on endpoints that provide structured querying capabilities (e.g., SPARQL interfaces), only HTTP URI lookups are available. Thus, entire sources have to be retrieved. Even for a single trivial query, hundreds of sources have to be processed in their entirety [10]. Aiming at delivering up-to-date results, sources often cannot be cached, but have to be fetched from external hosts. Thus, *efficiency* and *scalability* are essential problems in the Linked Data setting.

A widely adapted strategy for dealing with efficiency and scalability problems is to perform top- k processing. Instead of computing all results, *top- k query processing* approaches produce only the “best” k results [8]. This is based on the observation that results may vary in “relevance” (which can be quantified via a ranking function), and users, especially on the Web, are often interested in only a few relevant results. Let us illustrate top- k Linked Data query processing:

¹ <http://www.w3.org/DesignIssues/LinkedData.html>

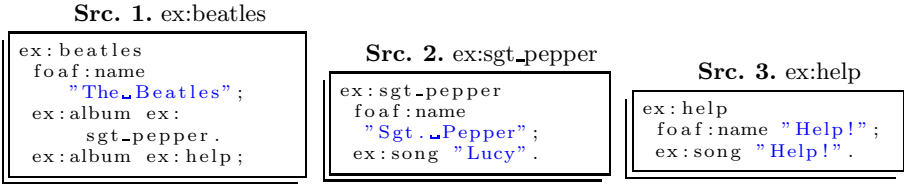


Fig. 1. Linked Data sources describing “The Beatles” and their songs “Help!” and “Lucy”

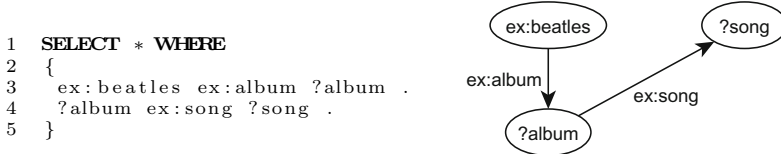


Fig. 2. Example query returning songs in Beatles albums. The query comprises two triple patterns q_1 (line 3) and q_2 (line 4).

Example 1. For the query in Fig. 2, the URIs `ex:beatles`, `ex:help` and `ex:sgt_pepper` are dereferenced to produce results for `?song` and `?album`. The results are retrieved from different sources, which vary in “relevance” (i.e., `ex:help` provides the precise name for the song “Help!”, while `ex:sgt_pepper` merely holds “Lucy” as name for a song, which is actually called “Lucy in the Sky with Diamonds”). Such differences are captured by a ranking function, which is used in top- k query processing to measure the result relevance. For the sake of simplicity, assume a ranking function assigns triples in `ex:beatles` (s_1) a score of 1, triples in `ex:sgt_pepper` (s_2) score 2, and those in `ex:help` (s_3) a score of 3. Further, assume our ranking function assigns increasing values with increasing triple relevance.

While being appealing, top- k processing has not been studied in the Linked Data (and the general RDF) context before. Aiming at the adaption of top- k processing to the Linked Data setting, we provide the following contributions:

- Top- k query processing has been studied in different contexts [8]. Closest to our work is top- k querying over Web-accessible databases [20]. However, the Linked Data context is unique to the extent that only URI lookups are available for accessing data. Instead of retrieving partial results matching query parts from sources that are exposed via query interfaces (of the corresponding database endpoints), we have to retrieve entire sources via URI lookups. To the best of our knowledge, this is the first work towards top- k Linked Data query processing.
- We show that in a Linked Data setting, more detailed score information is available. We propose strategies for using this knowledge to provide tighter score bounds (and thus allow an earlier termination) as compared to top- k processing in other scenarios [2,13,17]. Further, we propose an aggressive technique for pruning partial query results that cannot contribute to the final top- k result.

- We perform an experimental evaluation on Linked Data datasets and queries to show that top- k processing leads to increased performance (35 % on average). We further show that our proposed top- k optimizations increase the performance compared to a baseline implementation by 12 % on average.

Outline. In Section 2 we introduce the problem of Linked Data query processing. In Section 3, we show how to adapt top- k join processing methods to the Linked Data setting. In Sections 3.3 and 3.4, we propose two optimizations: tighter bounds on future join results and a way to prune unnecessary partial results. We present our evaluation in Section 4 and discuss related work in Section 5, before concluding with Section 6.

2 Linked Data Query Processing

Data Model. We use RDF [9] as our data model. However, for clarity of presentation, we do not consider the special RDF semantics (e.g., RDF blank nodes) and focus on the main characteristics of RDF. Namely, RDF can be considered as a general model for graph-structured data encoded as $\langle s, p, o \rangle$ triples:

Definition 1 (RDF Triple, RDF Graph). *Given a set of URIs U and a set of literals L , $t = \langle s, p, o \rangle \in U \times U \times (U \cup L)$ is a RDF triple, and a set of RDF triples is called a RDF graph.*

The Linked Data principles used to access and publish RDF data on the Web, mandate that (1) HTTP URIs shall be used as URIs and that (2) dereferencing a URI returns a description of the resource identified by the URI. Thus, a URI d can be seen as a *Linked Data source*, whose content, namely a set of RDF triples T^d , is obtained by dereferencing d . Triples in T^d contain other HTTP URI references (*links*), connecting d to other sources. The union set of sources in U forms a *Linked Data graph* $G = \{t | t \in T^{d_i} \wedge d_i \in U\}$.

Query Model. The standard language for querying RDF is SPARQL [15]. Previous work on Linked Data query processing focused on processing *basic graph patterns* (BGP), which is a core feature of SPARQL.

Definition 2 (Triple Pattern, Basic Graph Pattern). *A triple pattern has the form $q = \langle s, p, o \rangle$, where s , p and o is either a URI, a literal or a variable. A basic graph pattern is a set of triple patterns $Q = \{q_1, \dots, q_n\}$.*

Result Model. Often, every triple pattern in a BGP query Q shares a common variable with at least one other pattern such that Q forms a connected graph. Computing *results* to a BGP query over G amounts to the task of *graph pattern matching*. Basically, a result to a query Q evaluated over G (given by $\mu_G(Q)$) is a subgraph of G that matches Q . The set of all results for query Q is denoted by $\Omega_G(Q)$.

Query Processing. Traditionally, a query Q is evaluated by obtaining bindings for each of its triple patterns and then performing a series of equi-joins between bindings obtained for patterns that share a variable. In the Linked Data context, BGP queries are evaluated against all sources in the Linked Data graph G . While some sources may be available locally, others have to be *retrieved via HTTP dereferencing during query processing*.

For this, exploration-based *link traversal* [6,5] can be performed *at runtime*. The link traversal strategy assumes that Q contains at least one URI d as “entry point” to G . Starting from triples in T^d , G is then searched for results by following links from d to other sources. Instead of exploring sources at runtime, knowledge about (previously processed) Linked Data sources in the form of statistics has been exploited to determine and rank relevant sources [3,10] *at query compilation time*. Existing approaches assume a *source index*, which is basically a map that associates a triple pattern q with sources containing triples that match q . Let the result of a lookup in the source index for q be $source(q)$.

Given a source index, Linked data query processing can be conceived as a series of *operators*. We identify the *source scan* as a distinctive operator in Linked Data query processing. Given a source d , $scan(d)$ outputs all triples in T^d . A *selection* $\sigma_{T^d}(q)$ is performed on T^d to output triples that match a triple pattern q . Two triple patterns q_i and q_j that share a common variable are combined via an *equi-join* operator $q_i \bowtie q_j$ (i.e., bindings for q_i respectively q_j are joined). In general, $Q_i \bowtie Q_j$ joins any subexpression $Q_i \subset Q$ with one other $Q_j \subset Q$ ($Q_i \cap Q_j = \emptyset$). Note, in the following, we refer to an equi-join simply as join. Also, we have $\bigcup(I_1, \dots, I_n)$, which outputs the *union* of its inputs I_i . For clarity of presentation, we assume triple patterns form a connected graph such that a join is the only operator used to combine triples from different patterns. Then, Linked Data query processing can be modeled as a tree-structured plan as exemplified in Fig. 3 (a).

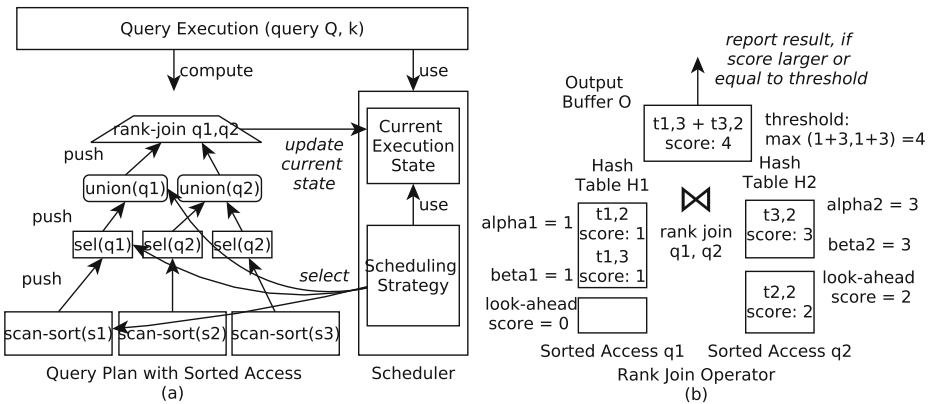


Fig. 3. (a) Query plan providing a sorted access, query execution and the scheduler. (b) Rank join operator with data from our “Beatles” example.

Query plans in relational databases generally consist of access plans for individual relations. Similarly, Linked Data query plans can be seen as being composed of *access plans* at the bottom-level (i.e., one for each triple pattern). An *access plan* for query $Q = \{q_1, \dots, q_n\}$ is a tree-structured query plan constructed in the following way: (1) At the lowest level, leaf nodes are source scan operators, one for every source that is relevant for triple pattern q_i (i.e., one for every $d \in \text{source}(q_i)$). (2) The next level contains selection operators, one for every scan operator. (3) The root node is a union operator $\bigcup(\sigma_{T^{d_1}}(q_i), \dots, \sigma_{T^{d_n}}(q_i))$, which combines the outputs of all selection operators for q_i (with $d_i \in \text{source}(q_i)$). At the next levels, the outputs of the access plans (of their root operators) are successively joined to process all triple patterns of the query, resulting in a *tree of operators*.

Example 2. Fig. 3 (a) shows an example query plan for the query in Fig. 2. Instead of *scan* and *join*, their top- k counterparts *scan-sort* and *rank-join* are shown (explained in the next section). There are three source scan operators, one for each of the sources: `ex:beatles` (s1), `ex:sgt_pepper` (s2), and `ex:help` (s3). Together with selection and union operators, they form two access plans for the patterns q_1 and q_2 . The output of these access plans is combined using one join operator.

Push-Based Processing. In previous work [10,11], push-based execution using symmetric hash join operators was shown to have better performance than pull-based implementations (such as [6]). In a push-based model, operators push their results to subsequent operators instead of pulling from input operators, i.e., the execution is driven by the incoming data. This leads to better behavior in network settings, because, unlike in pull-based execution models, the query execution is not *blocked*, when a single source is delayed [11].

3 Top- k Join Linked Data Query Processing

Top- k query processing [14,7,17] aims at a more efficient query execution by focusing on the k best results, while skipping the computation of remaining results. This *early termination* can lead to a significant reduction in the number of inputs to be read and processed, which translates to performance improvements. We now discuss how existing *top- k join* (also called *rank join*) strategies can be adopted to the Linked Data query processing problem as presented before. Further, we present an optimization towards tighter bounds and an aggressive result pruning. Throughout the query processing we do not approximate, thus, our approach always reports correct and complete top- k final results.

3.1 Preliminaries

Besides the source index employed for Linked Data query processing, we need a *ranking function* as well as a *sorted access* for top- k processing [7,14,17].

Ranking Function. We assume the existence of a ranking function for determining the “importance” of triples and (partial) query results in G :

Definition 3 (Ranking Function). *Let a ranking function $v : G \mapsto [0, 1]$ assign scores to triples in G . Further, let higher scores denote higher triple importance. Given Q as a query over G and $\Omega_G(Q)$ as its results, v ranks results in $\Omega_G(Q)$ (i.e., $v : \Omega_G(Q) \mapsto [0, 1]$) as an aggregation of their triple scores: $\mu \in \Omega_G(Q): v(\mu) = \Delta(v(t_1), \dots, v(t_n)), t_i \in \mu$, where Δ is a monotonic aggregation function.*

Scores for triples can, e.g., be obtained through PageRank inspired ranking [4] or witness counts [1].

Sorted Access. A *sorted access* on a given join input allows to access input elements in *descending* score order. In a database setting, a sorted access can be efficiently provided by using a score index over the input data. In particular, while work on top- k join processing over Web-accessible databases [20] aims at a similar setting, it also assumes such a complete index. However, in the Linked Data context, only source statistics are assumed to be available, while the contained triples are not indexed (e.g., for the sake of result freshness). Following this tradition, we only assume that score bounds are known (i.e., computed at indexing time) for the sources, while triples are ranked and sorted on-the-fly.

Definition 4 (Source Score Bounds). *Given a source $d \in U$, its upper bound score $v_u(d)$ is defined as the maximal score of the triples contained in d , i.e., $v_u(d) = \max\{v(t) | t \in T^d\}$. Conversely, the lower bound score is defined as $v_l(d) = \min\{v(t) | t \in T^d\}$.*

For each triple pattern in the source index, we store its list of relevant sources in descending order of their upper bound scores v_u . This allows sources for each *union* operator to be retrieved sequentially in the order of their upper bound scores. Further, as triples for a given source are not sorted, we replace each *scan* operator with a *scan-sort* operator. A *scan-sort* operator, after retrieving a source d , sorts its triples T^d according to their scores. However, if two (or more) sources (say, d_i and d_j) have overlapping source score bounds (i.e., $v_l(d_i) < v_u(d_j) < v_u(d_i)$), and both are inputs for the same union, the output of the union will not be ordered if these sources are retrieved individually. We address this problem by treating both sources as “one source”. That is, d_i and d_j are scanned and sorted via the same *scan-sort* operator. Fig. 3 (a) shows an access plan with *scan-sort* operators, which provide a sorted access to the bindings of q_1 and q_2 . Last, note that $v_u(d)$ respectively $v_l(d)$ does not necessarily have to be precise, it could also be estimated (e.g., based on scores of similar sources).

3.2 Push-Based Top- k Join Processing

Based on the ranking function, source index, and our sorted access mechanism, we can now adapt top- k strategies to the Linked Data setting. However, previous work on the top- k join problem uses pull-based processing, i.e., join operators actively

“pull” their inputs in order to produce an output [7,17,20]. In compliance with [17], we adapt the *pull/bound rank join* (PBRJ) algorithm template for a *push*-based execution in the Linked Data setting. For simplicity, the following presentation of the PBRJ algorithm assumes *binary* joins (i.e., each join has two inputs).

In a pull-based implementation, operators call a `next` method on their input operators to obtain new data. In a push-based execution, the control flow is inverted, i.e., operators have a `push` method that is called by their input operators. Algorithm 1 shows the `push` method of the PBRJ operator. The input from which the input element r was pushed is identified by $i \in \{1, 2\}$. Note, by input element we mean either a triple (if the input is a *union* operator) or a partial query result (if the input is another *rank join* operator). First, the input element r is inserted into the hash table H_i (line 3). Then, we probe the other input’s hash table H_j for valid join combinations (i.e., the join condition evaluates to “true”; see line 4), which are then added to the output queue O (line 5). Output queue O is a priority queue such that the result with the highest score is always first. The threshold Γ is updated using the *bounding strategy* \mathcal{B} , providing an upper bound on the scores of future join results (i.e., result combinations comprising “unseen” input elements). When a join result in queue O has a score equal to or greater than the threshold Γ , we know there is no future result having a higher score. Thus, the result is ready to be reported to a subsequent operator. If output O contains k results, which are ready to be reported, the algorithm stops reading inputs (so-called early termination).

As reported in [17], the PBRJ has two parameters: its *bounding strategy* \mathcal{B} and its *pulling strategy* \mathcal{P} . For the former, the *corner-bound* is commonly employed and is also used in our approach. The latter strategy, however, is proposed for a pull-based execution and is thus not directly applicable. Similar to the idea behind the pulling strategy, we aim to have control over the results that are pushed to subsequent operators. Because a push-based join has no influence over the data flow, we introduce a *scheduling strategy* to regain control. Now, the `push` method only adds join results to the output queue O , but does not push them to a subsequent operator. Instead the pushing is performed in a separate `activate` method as mandated by the scheduling strategy.

Algorithm 1. PBRJ.*push*(r)

Input: Pushed input element r on input $i \in \{1, 2\}$
Data: Bounding strategy \mathcal{B} , output queue O , threshold Γ , hash tables H_1, H_2

- 1 if $i = 1$ then $j = 2$;
- 2 else $j = 1$;
- 3 Insert r into hash table H_i ;
- 4 Probe H_j for valid join combinations with r ;
- 5 foreach valid join combination o do Insert o into O ;
- 6 $\Gamma \leftarrow \mathcal{B}.\text{update}()$;

Bounding Strategies. A bounding strategy is used to update the current threshold (i.e., the upper bound on scores of future join results). As only those results in the output queue can be reported that have a score equal to or greater

than the threshold Γ , it is essential that the upper bound is as low (tight) as possible. In other words, a tight upper bound allows for an early termination of the top- k join procedure, which results in less sources being loaded and triples being processed. The most common choice for \mathcal{B} is the *corner bound* strategy:

Definition 5 (Corner-Bound). For a rank join operator, we maintain an upper bound α_i and a lower bound β_i (both initialized as ∞) on the scores of its input elements from $i \in \{1, 2\}$, where α_i is the score of the first (highest) element r_i^{max} received on input i , $\alpha_i = v(r_i^{max})$, and β_i is the score of the most recently received input element \hat{r}_i , $\beta_i = v(\hat{r}_i)$. Then, the threshold Γ for future join results is given by $\max\{\Delta(\alpha_1, \beta_2), \Delta(\alpha_2, \beta_1)\}$, i.e., the score for the join between r_1^{max} and \hat{r}_2 or between \hat{r}_1 and r_2^{max} .

Scheduling Strategies. Deciding which input to pull from has a large effect on operator performance [17]. Previously, this decision was captured in a *pulling strategy* employed by the join operator implementation. However, in push-based systems, the execution is not driven by results, but by the input data. Join operators are only activated when input is actively being pushed from operators lower in the operator tree. Therefore, instead of pulling, we propose a *scheduling strategy* that determines which operators in a query plan are scheduled for execution. That is, we move the control over which input is processed from the join operator to the query engine, which orchestrates the query execution.

Algorithm 2 shows the `execute` method that takes a query Q and the number of results k as input and returns the top- k results. First, we obtain a query plan P from the `plan` method (line 1). We then use the scheduling strategy \mathcal{S} to obtain the next operator that should be scheduled for execution (line 2). The scheduling strategy uses the current execution state as captured by the operators in the query plan to select the next operator. We then activate the selected operator (line 4). We select a new operator (line 5) until we either have obtained the desired number of k results or there is no operator to be activated, i.e., all inputs have been exhausted (line 3).

Algorithm 2. `execute(Q, k)`

Input: Query Q , #results k
Data: Query plan P , scheduling strategy \mathcal{S}
Output: Query results Ω_G

- 1 $P \leftarrow \text{plan}(Q)$;
- 2 $op \leftarrow \mathcal{S}.\text{nextOp}(P)$;
- 3 **while** $|\Omega_G| < k \wedge op \neq \text{null}$ **do**
- 4 $op.\text{activate}()$;
- 5 $op \leftarrow \mathcal{S}.\text{nextOp}(P)$;
- 6 **return** Ω_G

Algorithm 3. `PBRJ.activate`

Data: Output queue O , threshold Γ , subsequent operator out

- 1 **while** $v(O.\text{peek}()) \geq \Gamma$ **do**
- 2 $r \leftarrow O.\text{dequeue}()$;
- 3 $out.\text{push}(r)$;

Algorithm 3 shows the `activate` method (called by `execute`) for the rank join operator. Intuitively, the `activate` method triggers a “flush” of the operator’s output buffer O . That is, all computed results having a score larger than or equal to the operator’s threshold Γ (line 11) are reported to the subsequent operator (lines 2-3). An `activate` method for a `scan-sort` operator of a source d simply pushes all triples in d in a sorted fashion. Further, `activate` for selection and union operators causes them to push their outputs to a subsequent operator.

Now, the question remains *how* a scheduling strategy should select the next operator (`nextOp` method). We can apply the idea behind the state-of-the-art pulling strategy [17] to perform *corner-bound-adaptive* scheduling. Basically, we choose the input which leads to the highest reduction in the corner-bound:

Definition 6 (Corner-Bound-Adaptive Scheduling). *Given a rank join operator, we prefer the input that could produce join results with the highest scores. That is, we prefer input 1 if $\Delta(\alpha_2, \beta_1) > \Delta(\alpha_1, \beta_2)$, otherwise we prefer input 2. In case of ties, the input with the least current depth, or the input with the least index is preferred. The scheduling strategy then “recursively” selects and activates operators that may provide input elements for the preferred input. That is, in case the chosen input is another rank join operator, which has an empty output queue, the scheduling strategy selects and activates operators for its preferred input in the same manner.*

Example 3. Assume $k = 1$ and let $t_{i,j}$ denote the j^{th} triple in source i (e.g., $t_{1,2} = \langle \text{ex:beatles}, \text{ex:album}, \text{ex:sgt_pepper} \rangle$). First, our scheduling strategy prefers the input 1 and selects (via `nextOp`) and activates `scan-sort`(s_1), `sel`(q_1), and `union`(q_1). Note, also input 2 would have been a valid choice, as the threshold (respectively α , β) is not set yet. The rank join reads $t_{1,2}$ and $t_{1,3}$ as new inputs elements from `union`(q_1), and both elements are inserted into H_1 ($\alpha_1 = \beta_1 = 1$). The scheduler now prefers input 2 (as input 1 is exhausted) and selects and activates `scan-sort`(s_3), `sel`(q_2), and `union`(q_2), because source 3 has triples with higher scores than source 2. Now, `union`(q_2) pushes $t_{3,2}$ and α_2 respectively β_2 is set to $v(t_{3,2}) = 3$. Employing a summation as Δ , the threshold Γ is set to 4 (as $\max\{1 + 3, 1 + 3\} = 4$). Then, $t_{3,2}$ is inserted into H_2 and the joins between $t_{3,2}$ and elements in H_1 are attempted; $t_{1,3} \bowtie t_{3,2}$ yields a result μ , which is then inserted into the output queue. Finally, as $v(\mu) = 4 \geq \Gamma = 4$ is true, μ is reported as the top-1 result and the algorithm terminates. Note, not all inputs have been processed, i.e., source 2 has not been scanned (cf. Fig. 3).

3.3 Improving Threshold Estimation

We now present two modifications to the corner-bound bounding strategy that allow us to calculate a more precise (*tighter*) threshold $\tilde{\Gamma}$, thereby achieving earlier result reporting and termination.

Star-Shaped Entity Query Bounds. A star-shaped entity query is a set of triple patterns Q_s that share a common variable at the subject position. We observed that in Linked Data query processing, every result to such a query

is contained in one single source. This is because a result here is an entity, and information related to that entity comes exclusively from the one source representing that particular entity. Exploiting this knowledge, a more precise corner-bound for joins of a star-shaped query (part) can be calculated. Namely, we can derive that, in order to be relevant, sources for Q_s must satisfy all triple patterns in Q_s (because they must capture all information for the requested entities). Given relevant sources for Q_s are denoted as D and the source upper bound is given by $v_u(d)$ for $d \in D$, the upper bound score $v_u^Q(Q_s)$ for results matching Q_s can be derived based on the maximum source upper bound $v_u^{max}(D) = \max\{v_u(d) | d \in D\}$. More precisely, $v_u^Q(Q_s) = \Delta(v_u^Q(q_1), \dots, v_u^Q(q_n))$ with $q_i \in Q_s$ and $v_u^Q(q_i) = v_u^{max}(D)$, because every triple that contributes to the result must be contained in a source $d \in D$, and thus, must have a score $\leq v_u^{max}(D)$.

Look-Ahead Bounds. The corner-bound strategy uses the last-seen scores β_i of input elements to calculate the current threshold. We observed that when an input element r_i is received by an operator on input i , the next input element r_i^{next} (and its score $v(r_i^{next})$) is often already available in the pushing operator. The next element is available because (1) scan-sort operators materialize their complete output before pushing to subsequent operators, (2) rank join operators maintain an output queue that often contains more than one result with scores greater than or equal to the current threshold Γ , and (3) given a source d_i has been pushed by a scan-sort operator, the source score upper bound of d_{i+1} (i.e., the next source to be loaded) is available. By using the score of the next instead of the last-seen input element, we can provide a more accurate threshold Γ , because we can estimate the maximal score of unseen elements from that particular input more accurately. If available, we therefore define $\tilde{\beta}_i = v(r_i^{next})$ as the score of the next input element. Otherwise, we use the last-seen score β_i , i.e., $\tilde{\beta}_i = \beta_i$ (see Fig. 3 (b)).

Threshold Calculation. By applying both strategies, we can now refine the bound as $\tilde{\Gamma} = \max\{\min\{\Delta(\alpha_1, \tilde{\beta}_2), v_u^Q(Q_s)\}, \min\{\Delta(\alpha_2, \tilde{\beta}_1), v_u^Q(Q_s)\}\}$. The following theorem (see proof in our report [19]) allows to use $\tilde{\Gamma}$ for top- k processing:

Theorem 1. $\tilde{\Gamma}$ is correct (i.e., there is no unseen result constituting to the final top- k results) and more precise than Γ (i.e., $\tilde{\Gamma} \leq \Gamma$ holds at all times).

3.4 Early Pruning of Partial Results

Knowledge about sources can also be exploited to *prune partial results* from the output queues to reduce the cost of a join as well as the memory space needed to keep track of input elements in a join operator. The idea of pruning has been pursued by approximate top- k selection [18] approaches. However, we do not approximate, but only prune those partial results that are *guaranteed* not to be part of the final top- k results. Intuitively, we can prune a partial result, if its score together with the maximal possible score for the “unevaluated” query part, is smaller than the lowest of the k so far computed complete results. Note,

the opportunity for pruning arises only when k (or more) complete results have been produced (by the root join operator).

More precisely, let Q be a query and $\mu(Q_f)$ a partial query result, with Q_f as “finished” part and Q_r as “remaining” part ($Q_f \subset Q$ and $Q_r = Q \setminus Q_f$). The upper bound on the scores of all final results based on $\mu(Q_f) \in \Omega_G(Q_f)$ can be obtained by aggregating the score of $\mu(Q_f)$ and the maximal score $v_u^Q(Q_r)$ of results $\mu(Q_r) \in \Omega_G(Q_r)$. $v_u(Q_r)$ can be computed as the aggregation of maximal source upper bounds obtained for every triple pattern in $Q_r = \{q_1, \dots, q_m\}$, i.e., $v_u^Q(Q_r) = \Delta(v_u^Q(q_1), \dots, v_u^Q(q_m))$, where $v_u^Q(q_i) = \max\{v_u(d) \mid d \in \text{source}(q_i)\}$. A tighter bound for $v_u^Q(Q_r)$ can be obtained, if Q_r contains one or more entity queries (see previous section) and aggregating their scores in a greedy fashion. Last, the following theorem can be established (see proof in [19]):

Theorem 2. *A result $\mu_f \in \Omega_G(Q_f)$ cannot be part of the top- k results for Q if $\Delta(v(\mu_f), v_u^Q(Q_r)) < \min\{v(\mu) \mid \mu \in \Omega_G^k(Q)\}$, where $\Omega_G^k(Q)$ are the currently known k results of Q .*

4 Experimental Evaluation

In the following, we present our evaluation and show that (1) top- k processing outperforms state-of-the-art Linked Data query processing, when producing only a number of top results, and (2) our tighter bounding and early pruning strategy outperform baseline rank join operators in the Linked Data setting.

Systems. In total, we implemented three different systems, all based on push-based join processing. For all queries, we generated left-deep query plans with random orders of join operators. All systems use the same plans and are different only in the implementation of the join operator.

First, we have the push-based symmetric hash join operator (*shj*) [10,11], which does not employ top- k processing techniques, but instead produces all results and then sorts them to obtain the requested top- k results. Also, there are two implementations of the rank join operator. Both use the corner-bound-adaptive scheduling strategy (which has been shown to be optimal in previous work [17]), but with different bounding strategies. The first uses the corner-bound (*rc-cc*) from previous work [17], while the second (*rc-tc*) employs our optimization with tighter bounds and early result pruning. The *shj* baseline is used to study the benefits of top- k processing in the Linked Data setting, while *rc-cc* is employed to analyze the effect of the proposed optimizations.

All systems were implemented in Java 6. Experiments were run on a Linux server with two Intel Xeon 2.80GHz Dual-Core CPUs, 8GB RAM and a Seagate ST31000340AS 1TB hard disk. Before each query execution, all operating system caches were cleared. The presented values are averages collected over three runs.

Dataset and Queries. We use 8 queries from the Linked Data query set of the FedBench benchmark. Due to schema changes in DBpedia and time-outs observed during the experiments (> 2 min), three of the 11 FedBench queries were omitted. Additionally, we use 12 queries we created. In total, we have 20

queries that differ in the number of results they produce (from 1 to 10118) and in their complexity in terms of the number of triple patterns (from 2 to 5). A complete listing of our queries can be found in [19].

To obtain the dataset, we executed all queries directly over the Web of Linked Data using a link-traversal approach [6] and recorded all Linked Data sources that were retrieved during execution. In total, we downloaded 681,408 Linked Data sources, comprising a total of 1,867,485 triples. From this dataset we created a source index that is used by the query planner to obtain relevant sources for the given triple patterns.

Scores were randomly assigned to triples in the dataset. We applied three different score distributions: uniform, normal ($\mu = 5, \sigma^2 = 1$) and exponential ($\lambda = 1$). This allows us to abstract from a particular ranking and examine the applicability of top- k processing for different classes of functions. We used summation as the score aggregation function Δ .

We observed that network latency greatly varies between hosts and evaluation runs. In order to systematically study the effects of top- k processing, we thus decided to store the sources locally, and to simulate Linked Data query processing on a single machine (as done before [10,11]).

Parameters. Parameter $k \in \{1, 5, 10, 20\}$ denotes the number top- k results to be computed. Further, there are the three different score distributions $d \in \{u, n, e\}$ (uniform, normal and exponential, respectively).

Overall Results. Fig. 4a shows an overview of processing times for all queries ($k = 1, d = n$). We can see that for all queries the rank join approaches perform better or at least equal to the baseline *shj* operator. On average, the execution times for *rj-cc* and *rj-tc* were 23.13s and 20.32s, whereas for *shj* it was 43.05s. This represents an improvement in performance of the *rj-cc* and *rj-tc* operators over the *shj* operator by factors of 1.86 and 2.14, respectively.

The improved performance of the rank join operators is due to top- k processing, because these operators do not have to process all input data in order to produce the k top results, but can terminate early. On the other hand, the *shj* implementation produces all results. Fig. 4b shows the average number of retrieved sources for different values of k . We can see clearly that the rank join approaches retrieve fewer sources than the baseline approach. In fact, *rj-cc* and *rj-tc* retrieve and process only 41% and 34%, respectively, of the sources that the *shj* approach requires. This is a significant advantage in the Linked Data context, where sources can only be retrieved in their entirety.

However, we also see that the rank join operators sometimes do not perform better than *shj*. In these cases, the result is small (e.g., Q18 has only two results). The rank join operators have to read all inputs and compute all results in these cases. For example, for Q20 the rank join approaches retrieve and process all 35103 sources, just as the *shj* approach does.

Bounding Strategies. We now examine the effect of the bounding strategies on overall execution time. The average processing times mentioned earlier represent

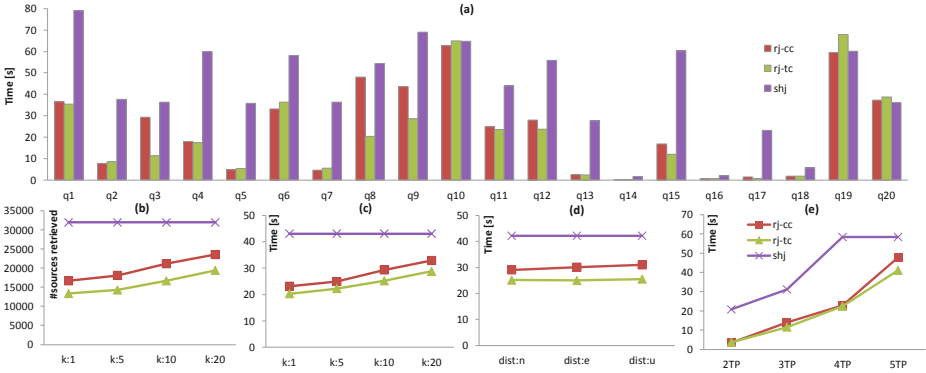


Fig. 4. (a) All queries with their evaluation times ($k = 1, d = n$). (b) Average number of sources over all queries (different $k, d = n$). (c) Average evaluation time over all queries (different $k, d = n$). (d) Average evaluation time over all queries (different score distributions, $k = 10$). (e) Average evaluation time over all queries with varying number of triple patterns ($k = 1, d = n$).

an improvement of 12% of *rj-tc* over *rj-cc*. For Q3, the improvement is even higher, where *rj-tc* takes 11s, compared to 30s for *rj-cc*.

The improved performance can be explained with the tighter, more precise bounding strategy of *rj-tc* compared to *rj-cc*. For example, our bounding strategy can take advantage of a large star-shaped subexpression with 3 patterns in Q3, leading to better performance because of better upper bound estimates. Moreover, we observed that the look-ahead strategy helps to calculate a much tighter upper bound especially when there are large score differences between successive elements from a particular input.

In both cases, a tighter (more precise) bound means that results can be reported earlier and less inputs have to be read. This is directly reflected in the number of sources that are processed by *rj-tc* and *rj-cc*, where on average, *rj-tc* requires 23% fewer sources than *rj-cc*. Note, while in Fig. 4a *rj-tc*’s performance often seems to be comparable to *rj-cc*, Fig. 4b makes the differences more clear in terms of the number of retrieved sources. For instance, both systems require an equal amount of processing times for Q17. However *rj-tc* retrieves 7% less sources. Such “small” savings did not show properly in our evaluation (as we retrieved sources locally), but would effect processing time in a real-world setting with network latency.

Concerning the outlier Q19, we noticed that *rj-tc* did read slightly more input (2%) than *rj-cc*. This behavior is due to our implementation: Sources are retrieved in parallel to join execution. In some cases, the join operators and the source retriever did not stop at the same time.

We conclude that *rj-tc* performs equally well or better than *rj-cc*. For some queries (i.e., entity queries and inputs with large score differences) we are able to achieve performance gains up to 60% compared to the *rj-cc* baseline.

Early Pruning. We observed that this strategy leads to lower buffer sizes (thus, less memory consumption). For instance with Q9, *rj-tc* could prune 8% of its buffered data. However, we also noticed that the number of sources loaded and scanned is actually the key factor. While pruning had positive effects, the improvement is small compared to what could be achieved with tighter bounds (for Q9 73% of total processing time was spent on loading and scanning sources).

Effect of Result Size k . Fig. 4c depicts the average query processing time for all three approaches at different k (with $d = n$). We observed that the time for *shj* is constant in k , as *shj* always computes all results, and that the rank join approaches outperform *shj* for all k . However, with increasing k , more inputs need to be processed. Thus, the runtime differences between the rank join approaches and *shj* operator become smaller. For instance, for $k = 1$ the average time saving over all queries is 46% (52%) for *rj-cc* (*rj-tc*), while it is only 31% (41%) for $k = 10$.

Further, we can see in Fig. 4c that *rj-tc* outperforms *rj-cc* over all values for k . The differences are due to our tighter bounding strategy, which substantially reduces the amount of required inputs. For instance, for $k = 10$, *rj-tc* requires 21% less inputs than *rj-cc* on average.

We see that *rj-tc* and *rj-cc* behave similarly for increasing k . Both operators become less efficient with increasing k (Fig. 4c).

Effect of Score Distributions. Fig. 4d shows average processing times for all approaches for the three score distributions. We see that the performance of both rank join operators varied only slightly w.r.t. different score distributions. For instance, *rj-cc* performed better by 7% on the normal distribution compared to the uniform distribution. The *shj* operator has constant evaluation times over all distributions.

Effect of Query Complexity. Fig. 4e shows average processing times (with $k = 1, d = n$) for different numbers of triple patterns. Overall, processing times increase for all systems with an increasing number of patterns. Again, we see that the rank join operators outperform *shj* for all query sizes. In particular, for 5 queries patterns, we noticed the effects of our entity bounds more clearly, as those queries often contained entity queries up to the length of 3.

5 Related Work

The top- k join problem has been addressed before, as discussed by a recent survey [8]. The J* rank join, based on the A* algorithm, was proposed in [14]. Other rank join algorithms, HRJN and HRJN*, were introduced in [7] and further extended in [12]. In contrast to previous works, we aim at the Linked Data context. As recent work [6, 3, 10, 11] has shown, Linked Data query processing introduces various novel challenges. In particular, in contrast to the state-of-the-art *pull*-based rank join, we need a push-based execution for queries over Linked Data. We therefore adapt pull strategies to the push-based execution model (based on

operator scheduling). Further, our work is different from prior work on Web-accessible databases [20], because we rely exclusively on simple HTTP lookups for data access, and use only basic statistics in the source index.

There are different bounding strategies: In [217], the authors introduced a new Feasible-Region (FR) bound for the general setting of n -ary joins and multiple score attributes. However, it has been proven that the PBRJ template is *instance-optimal* in the restricted setting of binary joins using corner-bound and a single score attribute [217]. We adapt the corner-bound to the Linked Data setting and provide tighter, more precise bounds that allow for earlier termination and better performance.

Similar to our pruning approach, [18] estimates the likelihood of partial results contributing to a final result (if the estimate is below a given threshold partial results are pruned). However, [18] addressed the *selection top- k problem*, which is different to our top- k join problem. More importantly, we do not rely on probabilistic estimates for pruning, but employ accurate upper bounds. Thus, we do not approximate final top- k results.

6 Conclusion

We discussed how existing top- k join techniques can be adapted to the Linked Data context. Moreover, we provide two optimizations: (1) tighter bounds estimation for early termination, and (2) aggressive result pruning. We show in real-world Linked Data experiments that top- k processing can substantially improve performance compared to the state-of-the-art baseline. Further performance gains could be observed using the proposed optimizations. In future work, we like to address different scheduling strategies as well as further Linked Data aspects like network latency or source availability.

References

1. Elbassuoni, S., Ramanath, M., Schenkel, R., Sydow, M., Weikum, G.: Language-model-based ranking for queries on RDF-graphs. In: CIKM, pp. 977–986 (2009)
2. Finger, J., Polyzotis, N.: Robust and efficient algorithms for rank join evaluation. In: SIGMOD, pp. 415–428 (2009)
3. Harth, A., Hose, K., Karnstedt, M., Polleres, A., Sattler, K., Umbrich, J.: Data summaries for on-demand queries over linked data. In: World Wide Web (2010)
4. Harth, A., Kinsella, S., Decker, S.: Using Naming Authority to Rank Data and Ontologies for Web Search. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 277–292. Springer, Heidelberg (2009)
5. Hartig, O.: Zero-Knowledge Query Planning for an Iterator Implementation of Link Traversal Based Query Execution. 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. 154–169. Springer, Heidelberg (2011)
6. Hartig, O., Bizer, C., Freytag, J.-C.: Executing SPARQL Queries over the Web of Linked Data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 293–309. Springer, Heidelberg (2009)

7. Ilyas, I.F., Aref, W.G., Elmagarmid, A.K.: Supporting top-*k* join queries in relational databases. *The VLDB Journal* 13, 207–221 (2004)
8. Ilyas, I.F., Beskales, G., Soliman, M.A.: A survey of top-*k* query processing techniques in relational database systems. *ACM Comput. Surv.* 58, 11:1–11:58 (2008)
9. Klyne, G., Carroll, J.J., McBride, B.: Resource description framework (RDF): concepts and abstract syntax (2004)
10. Ladwig, G., Tran, T.: Linked Data Query Processing Strategies. 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. 453–469. Springer, Heidelberg (2010)
11. Ladwig, G., Tran, T.: SIHJoin: Querying Remote and Local Linked Data. 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. 139–153. Springer, Heidelberg (2011)
12. Li, C., Chang, K.C.-C., Ilyas, I.F., Song, S.: Ranksql: query algebra and optimization for relational top-*k* queries. In: *SIGMOD*, pp. 131–142 (2005)
13. Mamoulis, N., Yiu, M.L., Cheng, K.H., Cheung, D.W.: Efficient top-*k* aggregation of ranked inputs. *ACM Trans. Database Syst.* (2007)
14. Natsev, A., Chang, Y.-C., Smith, J.R., Li, C.-S., Vitter, J.S.: Supporting incremental join queries on ranked inputs. In: *VLDB*, pp. 281–290 (2001)
15. Prud’hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. *W3C Recommendation* (2008)
16. Schmedding, F.: Incremental SPARQL evaluation for query answering on linked data. In: *Workshop on Consuming Linked Data in Conjunction with ISWC* (2011)
17. Schnaitter, K., Polyzotis, N.: Optimal algorithms for evaluating rank joins in database systems. *ACM Trans. Database Syst.* 35, 6:1–6:47 (2010)
18. Theobald, M., Weikum, G., Schenkel, R.: Top-*k* query evaluation with probabilistic guarantees. In: *VLDB*, pp. 648–659 (2004)
19. Wagner, A., Tran, D.T., Ladwig, G., Harth, A., Studer, R.: Top-*k* linked data query processing (2011), <http://www.aifb.kit.edu/web/Techreport3022>
20. Wu, M., Berti-Equille, L., Marian, A., Procopiuc, C.M., Srivastava, D.: Processing top-*k* join queries. In: *VLDB*, pp. 860–870 (2010)

Preserving Information Content in RDF Using Bounded Homomorphisms

Audun Stolpe and Martin G. Skjæveland

Department of Informatics, University of Oslo, Norway
{audus,martige}@ifi.uio.no

Abstract. The topic of study in the present paper is the class of RDF homomorphisms that substitute one predicate for another throughout a set of RDF triples, on the condition that the predicate in question is not also a subject or object. These maps turn out to be suitable for reasoning about similarities in information content between two or more RDF graphs. As such they are very useful e.g. for migrating data from one RDF vocabulary to another. In this paper we address a particular instance of this problem and try to provide an answer to the question of when we are licensed to say that data is being transformed, reused or merged in a non-distortive manner. We place this problem in the context of RDF and Linked Data, and study the problem in relation to SPARQL construct queries.

1 Introduction

As yet, the World Wide Web shows a bias towards getting the information to flow, at the expense of maintaining the integrity of the circulated information. Maintaining integrity is usually recognised as a very real and increasingly acute need, though. Take public sector information: open public sector information is a valuable national resource, and there is widespread agreement that dissemination promotes transparent and accountable government, improves quality of service, and in general serves to maintain a well-informed public. Yet, whilst the political pressure for reusable public sector information is building momentum, as witnessed e.g. by the European Public Sector Information Directive of 2003, governments as suppliers and authoritative sources of information on the Web must nevertheless acknowledge the challenges related to maintaining the primary nature of its information. This points to a general tension between two fundamental requirements of the data-oriented Web: Keep the data freely flowing, but shepherd the data into sanctioned use. In the present paper we shall place this problem in the context of RDF and Linked Data, and study it in relation to SPARQL construct queries.

Example 1. The Cultural Heritage Management Office in Oslo is the City of Oslo's adviser on questions of cultural conservation of architecturally and culturally valuable buildings, sites and environments. It maintains a list of protected buildings, known as 'the yellow list', which has been transformed to RDF and published as Linked Data [11]. A small excerpt is given below:

```
<http://sws.ifi.uio.no/gulliste/kulturminne/208/5/6643335/597618>
rdf:type gul:Kontor ;
hvor:gateadresse "Akersgata 44" ; geo:long "10.749" ;
hvor:postnummer "0180" ; geo:lat "59.916" .
```

Note that there is no explicit representation of city or country, and no grouping of similar data. Suppose now that we wish to lift all available information about culturally valuable buildings in Norway to the national level. We do so by adding Oslo and Norway as parts of the address data. Also, we add types to buildings by linking to the relevant class from the CIDOC CRM standard for cultural heritage information (<http://www.cidoc-crm.org/>). For heuristic purposes we also group geographical information and address information respectively around suitably typed nodes:

```
CONSTRUCT{ ?x rdf:type ?y, cidoc:E25.Man-Made_Feature;
  vcard:adr [ rdf:type vcard:Address;
    vcard:street-address ?street; vcard:zip-code ?code;
    vcard:locality geonames:3143242; # Oslo
    vcard:country-name "Norway"@en ] ;
  vcard:geo [ rdf:type geo:Point; geo:lat ?lat; geo:long ?long ] }
WHERE{ ?x rdf:type ?y;
  hvor:gateadresse ?street; hvor:postnummer ?code;
  geo:lat ?lat; geo:long ?long . }
```

The structural change to the data caused by the construct query is rather thoroughgoing and extensive. Yet, there is still a principled relationship between structural elements of the two graphs, e.g. the property `hvor:gateadresse` morphs into the sequence of properties `vcad:adr`, `vcad:street-address`. Moreover, the pairs of resources that are linked by `hvor:gateadresse` in the former graph remain linked by `vcad:adr`, `vcad:street-address` in the latter graph, and no other pairs are similarly related. Indeed, the transformation can easily be seen to be systematic in the sense that all pairs related in the same manner in the source graph are transformed uniformly in terms of the same structural element in the target graph. It is also non-distortive in the sense that no other pair of resources are so related. Contrast with the case in which we replace `hvor:postnummer` with `vcad:locality`, whilst keeping everything else as-is. We would then not be able to distinguish between cities and zip-codes in the target graph, and would in that sense have distorted the information from the source.

The purpose of the present paper is to give these intuitions mathematical content. That is, we wish to formulate a criterion to sort conservative from non-conservative ways of transforming data. Since we take construct queries as our paradigm of RDF transformation, this means sorting conservative from non-conservative construct queries. It is important to note that the uniformity and non-distortiveness criteria we shall propose are purely structural, and do not heed the semantics of the vocabulary elements involved. To the question ‘what makes the chain `vcad:adr`, `vcad:zip-code` an adequate representation of `hvor:gateadresse`?’ the only answer is ‘because somebody wishes it to be so’. What our criteria have to offer is thus nothing more than a clear notion of what

you are rationally committed to, in terms of the structure of the target graph, once you have made your choice of representatives for elements of the source graph. We will do so by studying a class of RDF homomorphisms that substitutes one edge for another throughout a set of RDF triples, subject to the condition that the edge in question is not also a vertex.

The paper is organised as follows: [Section 2](#) defines the general concept of an RDF homomorphism, and distinguishes the subset of conditional edge-substitutions mentioned above. We shall call them p-maps. [Section 3](#) recapitulates the basic syntax and semantics of the SPARQL query language. [Section 4](#) defines the notion of a bounded p-map and argues that it gives an adequate criterion of conservativeness. [Section 5](#) generalises the conservativeness criterion to handle more sophisticated construct queries, e.g. that of [Example 1](#). [Section 6](#) presents essential results on the computational properties of computing p-maps, whilst [Section 7](#) closes with a summary and a few pointers to future lines of research.

Related Work. Our homomorphisms differ from those of [\[1,2\]](#) which essentially rename blank nodes in order to mimic the semantics of RDF as defined in [\[6\]](#). To the best of our knowledge, our particular notion of RDF homomorphism, and the use of it, is novel. Considered as an embedding of one graph into another a p-map can be viewed in two different ways which, although they amount to the same formally, give rather different gestalts to the central issue. Looked at from one angle, our problem (i.e. embedding a source into a target) resembles data exchange: Given one source of data marked up in one way, one wants to migrate the data to some target repository in a way that conforms to the target’s schema. Yet, it differs from the problem studied in [\[4\]](#) in that our setup takes the target to be fixed and possibly non-empty. Looked at from another angle, the problem concerns how to *extend* an RDF graph conservatively. More specifically, it concerns the problem of how to ensure that a transformation of source data into a target repository does not interfere with the assertive content of the source. Yet, it is unlike logic-based conservative extensions [\[5,7,8\]](#) in that the logical vocabulary is being replaced as the source is ‘extended’ into the target. As such bounded p-maps may also have a role to play in *data fusion*, which is defined as “the process of fusing multiple records representing the same real-world object into a single, consistent, and clean representation” [\[3\]](#).

2 RDF Graphs and RDF Homomorphisms

Let U , B and L be pairwise disjoint infinite sets of *URI references*, *blank nodes* and *literals*, respectively, and let \mathcal{U} denote the union of these sets. An *RDF triple* is a member of $\mathcal{T} := (U \cup B) \times U \times \mathcal{U}$. We shall write RDF triples as $\langle a, p, b \rangle$ and say that a and b are *vertexes*, and p the *edge* of the triple. An *RDF graph* is a finite set of RDF triples. The *vocabulary* of an RDF graph G is the set $\mathcal{U}_G = V_G \cup E_G$, where V_G is the set of vertexes and E_G the set of edges in G . Note that V_G and E_G need not be disjoint—a matter of some importance as we shall see later. $\pi_i(t)$ denotes the i -th element of the tuple or sequence t .

Definition 1. An RDF homomorphism of RDF graph G to RDF graph H is a function $h : \mathcal{U}_G \rightarrow \mathcal{U}_H$ which induces a function $h : G \rightarrow H$ such that $h(\langle a, p, b \rangle) = \langle h(a), h(p), h(b) \rangle \in H$.

Definition 2. A p-map $h : G \rightarrow H$ is an RDF homomorphism $h : G \rightarrow H$ where $h(u) = u$ for all $u \in V_G$.

Thus, a p-map is an RDF homomorphism in which the only elements that are allowed to vary are edges: If $h : G \rightarrow H$ is an RDF homomorphism between RDF graphs G and H , then $h(g) \in H$ for all triples $g \in G$, while if h is a p-map, then $\langle a, h(p), b \rangle \in H$ for all triples $\langle a, p, b \rangle \in G$. This is a natural class of homomorphisms to study for our purposes since edges are typically vocabulary elements, while vertexes contain the “real” data. Note, though, that the definition of a p-map is not without subtleties, given that a single element in an RDF graph may be both a vertex and an edge:

Proposition 1. Let h be a p-map of G and let $\langle a, p, b \rangle$ be any arbitrarily chosen triple in G . Then $h(\langle a, p, b \rangle) = \langle a, p, b \rangle$ whenever $p \in V_G$.

3 Syntax and Semantics of SPARQL

To make this paper reasonably self-contained, we introduce a minimum of SPARQL syntax and semantics, considering only the select-project-join fragment. For a complete exposition of SPARQL, consult e.g. [1,9,10].

Assume the existence of a set of variables \mathcal{V} disjoint from \mathcal{U} . A SPARQL graph pattern is defined recursively as follows:

Definition 3. A SPARQL graph pattern S is either a triple pattern t in $(\mathcal{V} \cup U) \times (\mathcal{V} \cup U) \times (\mathcal{V} \cup U \cup L)$, or a conjunction $S_1 \& S_2$ of SPARQL graph patterns.

According to this definition SPARQL graph patterns do not contain blank nodes. As shown in [1] it is easy to extend the definition in this respect, but as blank nodes behave like variables in select queries, we shall not care to do so. We use $var(S)$ to denote the set of variables occurring in a set of triples S , and $var_p(S)$ to denote those occurring as edges, i.e. in the second element of triples.

Definition 4. A conjunctive SPARQL query, or just ‘select query’, is a pair $\langle S, x \rangle$, where S is a SPARQL graph pattern and x a subset of $var(S)$.

A variable mapping is a partial function $\mu : \mathcal{V} \rightarrow \mathcal{U}$ that extends in the natural way to a function on triples which respects RDF triple grammar, i.e. $\mu(S) \subseteq \mathcal{T}$ for any triple pattern S . The domain of a function f is denoted $dom(f)$, and the range by $ran(f)$. The semantics of the select-project-join fragment of select and construct SPARQL queries can now be given by the following series of definitions, modelled after [1,9]:

Definition 5. μ_1 and μ_2 are compatible variable mappings if for every $x \in dom(\mu_1) \cap dom(\mu_2)$ we have $\mu_1(x) = \mu_2(x)$.

Definition 6. Let Ω_1 and Ω_2 be sets of variable mappings. We define the join of Ω_1 and Ω_2 as $\Omega_1 \bowtie \Omega_2 := \{\mu_1 \cup \mu_2 \mid \mu_1 \in \Omega_1, \mu_2 \in \Omega_2, \mu_1 \text{ and } \mu_2 \text{ compatible}\}$.

Definition 7. The evaluation of S over an RDF graph G , written $\llbracket S \rrbracket_G$, is

1. $\{\mu \mid \text{dom}(\mu) = \text{var}(t) \text{ and } \mu(t) \in G\}$, if S is a triple pattern t ,
2. $\llbracket S_1 \rrbracket_G \bowtie \llbracket S_2 \rrbracket_G$, if S is a conjunction $S_1 \& S_2$

Definition 8. The answer to a query $\langle S, \mathbf{x} \rangle$ over a graph G , written $\langle S, \mathbf{x} \rangle(G)$, is the set $\{\mu(\mathbf{x}) \mid \mu \in \llbracket S \rrbracket_G\}$.

Definition 9. A SPARQL template is a SPARQL graph pattern in which blank nodes may occur as vertexes.

Definition 10. Let C be a SPARQL template, S a SPARQL graph pattern, G an RDF graph, and $\text{blank}(T)$ be the set of blank nodes occurring in a set of triples T . We define the set of renaming functions $\{\rho_\mu \mid \mu \in \llbracket S \rrbracket_G\}$ relative to C and S as follows:

- for every ρ_μ , $\text{dom}(\rho_\mu) = \text{blank}(C)$ and $\text{ran}(\rho_\mu) \subseteq (B \setminus \text{blank}(G))$,
- every ρ_μ is injective, and
- for all $\mu_1, \mu_2 \in \llbracket S \rrbracket_G$, if $\mu_1 \neq \mu_2$, then $\text{ran}(\rho_{\mu_1}) \neq \text{ran}(\rho_{\mu_2})$.

Definition 11. A SPARQL construct query, or just ‘construct query’, is a pair $\langle C, S \rangle$, where C is a SPARQL template and S a SPARQL graph pattern such that $\text{var}(C) \subseteq \text{var}(S)$. The answer to a construct query $\langle C, S \rangle$ over an RDF graph G , written $\langle C, S \rangle(G)$, is the RDF graph $\cup_{\mu \in \llbracket S \rrbracket_G} (\mu(\rho_\mu(C)))$.

We end this section with a lemma that links the principal notions introduced so far. It shows, essentially, that answers to queries and evaluations of SPARQL patterns are interchangeable idioms for talking about transformations of RDF graphs:

Lemma 1. Let G and H be RDF graphs, and h any function from \mathcal{U}_G to \mathcal{U}_H . Then,

1. $\langle S, \mathbf{x} \rangle(G) \subseteq \langle h(S), \mathbf{x} \rangle(H)$ iff $\llbracket S \rrbracket_G \subseteq \llbracket h(S) \rrbracket_H$.
2. $\langle h(S), \mathbf{x} \rangle(H) \subseteq \langle S, \mathbf{x} \rangle(G)$ iff $\llbracket h(S) \rrbracket_H \subseteq \llbracket S \rrbracket_G$.

Proof. The claim follows immediately from [Definition 8](#) and the fact that $\text{dom}(h) \cap \mathcal{V} = \emptyset$, whence $\text{var}(S) = \text{var}(h(S))$. \square

4 Degrees of Conservativeness

Having assembled the requisite preliminaries, we turn to the problem of analysing the notion of a conservative construct query. We shall limit the analysis in this section to the simple case where the query transforms RDF triples to RDF triples. Let G be any RDF graph. As a tentative characterisation we may say that a construct query is conservative if applied to G it evaluates to a graph

that conservatively transforms the sub-graph of G that matches the pattern in the `SELECT` clause. This pushes the question back to what it means for an RDF graph to be a conservative transformation of another. As the reader may suspect already, we shall take the existence of a particular kind of p-map between two graphs to provide a sufficient condition. As homomorphisms, p-maps in general reflect the structure of the source in the target. A simple consequence of this is that queries over the source can be translated to queries over the target without loss of tuples in the result set:

Theorem 1. *Let G and H be RDF graphs, $h : G \rightarrow H$ a p-map, and S a SPARQL pattern such that $\text{var}_p(S) = \emptyset$. Then $\langle S, \mathbf{x} \rangle(G) \subseteq \langle h(S), \mathbf{x} \rangle(H)$.*

The existence of a p-map between the source and the target graph may thus be taken to account for the systematicity of a construct query, as alluded to in [Section 1](#). It does not account for non-distortiveness for which we also need to reflect the structure of the result back into the source. We shall consider three ways of doing that, represented by *bounds* on p-maps:

Definition 12. *A p-map $h : G \rightarrow H$ is bounded, and called a p1-, p2- or p3-map, respectively, if it satisfies one of the following conditions; for all $a, p, b \in \mathcal{U}$:*

$$\langle a, h(p), b \rangle \in H \Rightarrow \langle a, p, b \rangle \in G \quad (\text{p1})$$

$$\langle a, h(p), h(b) \rangle \in H \text{ or } \langle h(a), h(p), b \rangle \in H \Rightarrow \langle a, p, b \rangle \in G \quad (\text{p2})$$

$$\langle h(a), h(p), h(b) \rangle \in H \Rightarrow \langle a, p, b \rangle \in G \quad (\text{p3})$$

As we shall see, each bound reflects a different aspect of the structure of the target in the source. It is easy to check that (p1) is strictly stronger than (p2), and that (p2) is strictly stronger than (p3). To be sure, there are other bounds, but these are particularly simple and natural. We shall need the following lemma:

Lemma 2. *If $\text{var}_p(t) = \emptyset$ and $\langle t, \mathbf{x} \rangle(G) \neq \emptyset$ for a triple pattern t , then $\pi_2(h(t)) = h(\pi_2(t))$ for any p-map h .*

Turning now to condition (p1) we obtain the converse of [Theorem 1](#):

Theorem 2. *If $\langle S, \mathbf{x} \rangle(G) \neq \emptyset$, $\text{var}_p(S) = \emptyset$ and h is a p1-map $h : G \rightarrow H$, then $\langle h(S), \mathbf{x} \rangle(H) \subseteq \langle S, \mathbf{x} \rangle(G)$.*

Proof. Assume the conditions of the theorem hold. By [Lemma 1](#) it suffices to show that $\llbracket h(S) \rrbracket_H \subseteq \llbracket S \rrbracket_G$. The proof proceeds by induction on the complexity of S . For the base case, suppose S is a triple pattern t and that $\mu \in \llbracket h(t) \rrbracket_H$. By [Definition 7\(II\)](#) it follows that $\mu(h(t)) \in H$. By the suppositions of the theorem we have $\text{var}_p(t) = \emptyset$ and $\langle t, \mathbf{x} \rangle(G) \neq \emptyset$, whence [Lemma 2](#) yields $\pi_2(h(t)) = h(\pi_2(t)) = h(p)$ for some $p \in \pi_2(G)$. Therefore $\mu(h(t)) = \langle a, h(p), b \rangle \in H$ for some a, b , whence $\langle a, p, b \rangle \in G$ since h is a p1-map. By [Definition 7\(II\)](#) $\text{dom}(\mu) = \text{var}(h(t)) = \text{var}(t)$, so $\mu \in \llbracket t \rrbracket_G$ as desired. For the induction step, assume the property holds for simpler patterns, and consider $S = S_b \& S_c$ such that the suppositions of the theorem hold, and such that $\mu \in \llbracket h(S_b \& S_c) \rrbracket_H$. It is

easy to check that $\llbracket h(S_b \& S_c) \rrbracket_H = \llbracket h(S_b) \& h(S_c) \rrbracket_H$, whence $\mu \in \llbracket h(S_b) \rrbracket_H \bowtie \llbracket h(S_c) \rrbracket_H$, by [Definition 7\(2\)](#). It follows from [Definition 6](#) that $\mu = \mu_b \cup \mu_c$ for compatible μ_b and μ_c such that $\mu_b \in \llbracket h(S_b) \rrbracket_H$ and $\mu_c \in \llbracket h(S_c) \rrbracket_H$. Now, since $\langle S_b \& S_c, \mathbf{x} \rangle(G) \neq \emptyset$ and $\text{var}_p(S_b \& S_c) = \emptyset$, by the supposition of the case, we have $\langle S_b, \mathbf{y} \rangle(G) \neq \emptyset$ and $\text{var}_p(S_b) = \emptyset$ for \mathbf{y} such that $y_i \in \text{dom}(\mu_b)$ for all $y_i \in \mathbf{y}$ and similarly for S_c . Therefore the induction hypothesis applies, so $\mu_b \in \llbracket S_b \rrbracket_H$ and $\mu_c \in \llbracket S_c \rrbracket_H$ by [Lemma 1](#). We have already assumed that μ_b and μ_c are compatible, so $\mu_b \cup \mu_c \in \llbracket S_b \rrbracket_G \bowtie \llbracket S_c \rrbracket_G = \llbracket S_b \& S_c \rrbracket_G$ by [Definition 7\(2\)](#). Since $\mu_b \cup \mu_c = \mu$, we are done. \square

[Theorem 1](#) and [Theorem 2](#) show that p1-maps induces a transformation between RDF graphs that is exact in the sense that the diagram in [Figure 1](#) commutes. That is, whatever answer a query Q yields over G , $h(Q)$ yields precisely the same answer over H . Interestingly, the converse is also true, if a function induces an, in this sense, exact transformation between graphs, then it is a p1-map:

Theorem 3. *Let h be any function from \mathcal{U} to itself. If for all SPARQL patterns S we have $\langle S, \mathbf{x} \rangle(G) = \langle h(S), \mathbf{x} \rangle(H)$, then h is a p1-map of G to H .*

Proof. The proof is by induction on the complexity of S . The induction step is easy, so we show only the base case where S is a triple pattern t . Suppose that h is not a homomorphism between G and H . Then there is a triple $\langle a, p, b \rangle \in G$ such that $\langle h(a), h(p), h(b) \rangle \notin H$. Let $t := \langle x, p, y \rangle$ and $\mathbf{x} := \langle x, y \rangle$. Then $h(t) = \langle x, h(p), y \rangle$, and $\langle a, b \rangle \in \langle t, \mathbf{x} \rangle(G) \setminus \langle h(t), \mathbf{x} \rangle(H)$. We therefore have $\langle t, \mathbf{x} \rangle(G) \not\subseteq \langle h(t), \mathbf{x} \rangle(H)$. Suppose next that h does not satisfy (p1). Then there is a triple $\langle a, h(p), b \rangle \in H$ such that $\langle a, p, b \rangle \notin G$, and $t := \langle x, p, y \rangle$ separates G and H by a similar argument. \square

The class of p1-maps thus completely characterises the pairs of graphs for which there is an exact triple-to-triple translation of select queries from one to the other. Note that exactness here does not mean that the source and target are isomorphic. The target may contain more information in the form of triples, as long as these triples do not have source edges that map to them. Indeed, a p1-map need not even be injective:

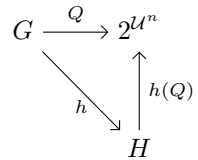


Fig. 1.

Example 2. Assume we have the following RDF graphs: $G := \{\langle a, p, b \rangle, \langle a, q, b \rangle\}$, $H_1 := \{\langle a, r, b \rangle\}$ and $H_2 := \{\langle a, r, b \rangle, \langle c, s, d \rangle\}$. Then $\{p \mapsto r, q \mapsto r\}$ is a p1-map of G to H_1 , and of G to H_2 , given that h is the identity on vertices.

Characterisation results similar to [Theorem 2](#) and [Theorem 3](#) are easily forthcoming for p2- and p3-maps as well. The proofs are reruns with minor modifications of that for p1-maps.

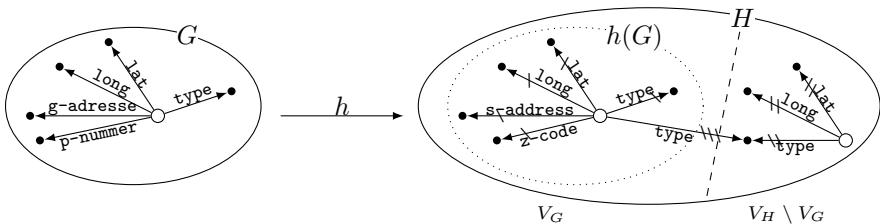
Theorem 4. *Let h be any function from \mathcal{U} to itself and suppose $\langle S, \mathbf{x} \rangle(G) \neq \emptyset$ and $\text{var}_p(S) = \emptyset$. Then, $h : G \rightarrow H$ is a p2-map iff $\mathbf{u} \in \langle h(S), \mathbf{x} \rangle(H) \setminus \langle S, \mathbf{x} \rangle(G)$ implies $u \notin \mathcal{U}_G$ for any $u \in \mathbf{u}$, and $h : G \rightarrow H$ is a p3-map iff $\mathbf{u} \in \langle h(S), \mathbf{x} \rangle(H) \setminus \langle S, \mathbf{x} \rangle(G)$ implies $u \notin \mathcal{U}_G$ for some $u \in \mathbf{u}$.*

Different bounds may be used to exercise different levels of control depending on the nature or interpretation of an edge. More specifically, different predicates may be restricted in different ways depending on the intended interpretation of those predicates: p1-maps are suitable for that part of a data-set to which one would wish to remain absolutely faithful, typically the domain-specific information that is collected and managed by the issuer of the data-set. p2-maps are suitable for situations where you would want to merge domain-specific knowledge from two different sources whilst keeping the information from each of the sources unchanged. They are more forgiving than p1-maps in the sense that they allow a relation to grow as long as every added pair relates new objects only. Finally, p3-maps would typically be applied to vocabulary elements that are most aptly considered as part of the logical or general-purpose vocabulary. For instance, applied to `rdf:type`, they allow types to be added to source elements as long as those types are not already used in the source. In other words, p3-maps allow additional typing as long as the added types do not occur in the source.

Example 3. Let G be the excerpt of triples listed in [Example 1](#) and assume it is transformed into the following target H :

```
<http://sws.ifi.uio.no/gulliste/kulturminne/208/5/6643335/597618>
  rdf:type gul:Kontor, cidoc:E25.Man-Made_Feature ;
  vcard:street-address "Akersgata 44" ; geo:long "10.749" ;
  vcard:zip-code "0180" ; geo:lat "59.916" .
<http://sws.ifi.uio.no/gulliste/kulturminne/999/2/6644406/596768>
  rdf:type cidoc:E25.Man-Made_Feature ;
  geo:long "10.731" ; geo:lat "59.926" .
```

The map of the source into the target shows the features of bounded p-maps given in the preceding paragraph. The edges `hvor:gateadresse` and `hvor:postnummer` are mapped to respectively `vcard:street-address` and `vcard:zip-code` under the (p1) bound, indicating that these edges relate the exact same data as their counterparts in the source. The edges `geo:lat` and `geo:long` are mapped to themselves under bound (p2), meaning that new relationship may be added as long as they relate only new data elements, i.e. elements not originating from the source. The edge `rdf:type` is also mapped to itself under a (p3) bound allowing new types to be added to the buildings in the yellow list (and new buildings be given old types). The transformation is illustrated below:



Marked arrows, \rightarrow , \Rightarrow and $\Rightarrow\Rightarrow$, represent triples satisfying bound (p1), (p2) and (p3), respectively. The set of target vertexes is partitioned into two sets, V_G and $V_H \setminus V_G$, illustrated by the dashed line.

Conservativeness (in our sense) is preserved by composition:

Theorem 5. *Let h_1 be a p-map of G to H that satisfies a bound p_i , and h_2 a p-map of H to I that satisfies a bound p_j , and suppose p_i logically entails p_j . Then $h_2 \circ h_1$ is a p-map that satisfies p_j .*

As the theorem shows, a composition of two bounded p-maps will satisfy the weakest of the two bounds. Since they are both conservative in the above-mentioned sense, we can claim that the use of bounded p-maps counteract cumulative error in iterated data transformation.

Note that SPARQL graph patterns and SPARQL templates are similar to RDF graphs in the sense that they too are sets of triples. Thus, we may extend the notion of a p-map accordingly by including variables in the domain and letting the p-map be the identity on those variables. Clearly, if h is a p-map of the latter sort, then we have $var(t) = var(h(t))$ for any triple pattern t . Moreover, for triple patterns where no edge is a variable, μ and h commute: $\mu(h(t)) = h(\mu(t))$. This allows us to prove the following result:

Theorem 6. *Let $\langle C, S \rangle$ be a construct query, where C contains no variables as edges. If h is a p-map of S to C which is bounded by one of (p1)–(p3), then h is a p-map under the same bound of $\langle S, S \rangle (G)$ to $\langle C, S \rangle (G)$.*

Proof. In the limiting case that $\llbracket S \rrbracket_G = \emptyset$, we have $\langle C, S \rangle (G) = \emptyset$ as well, whence the theorem holds vacuously. For the principal case where $\llbracket S \rrbracket_G \neq \emptyset$ suppose $g \in \langle S, S \rangle (G) = \cup_{\mu \in \llbracket S \rrbracket_G} (\mu(\rho_\mu(S)))$. By **Definition 3** we have that S does not contain blank nodes, so $g \in \cup_{\mu \in \llbracket S \rrbracket_G} (\mu(S))$. It follows that $g = \mu(t)$ for a triple pattern t in S and some $\mu \in \llbracket S \rrbracket_G$. By assumption, h is a p-map of S to C , whence $h(t)$ is a triple pattern in C , and since $var(h(t)) = var(t)$, it follows that $\mu(h(t)) \in \cup_{\mu \in \llbracket S \rrbracket_G} (\mu(\rho_\mu(C)))$. It remains to show that $h(g) = \mu(h(t))$. Since $g = \mu(t)$ it suffices to show that $h(\mu(t)) = \mu(h(t))$, which is just the commutativity of μ and h . The relationships between the different graphs are illustrated in **Figure 2**. Now assume that h from S to C is restricted by a bound (p1)–(p3), indicated by (p) in the figure. Then for every $t \in C$ there is a $t' \in S$ such that the bound holds. For all μ such that $\mu(t) \in \langle C, S \rangle (G)$ we have $\mu(t') \in \langle S, S \rangle (G)$, but then the p-map h must be restricted by the same bound as between $\langle S, S \rangle (G)$ and $\langle C, S \rangle (G)$. □

Thus, if there is a bounded p-map from the WHERE block to the CONSTRUCT block in a construct query, then any sub-graph that matches the former can be p-mapped with the same bound into the result of the query. By the properties of bounded p-maps, therefore, we are licensed to say that the construct query is a conservative transformation.

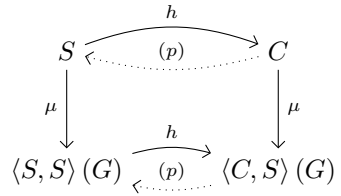


Fig. 2.

5 Generalising the Conservativeness Criterion

We take ourselves to have demonstrated that a bounded p-map is an interesting kind of structure-preserving map for the purpose of maintaining information content across repeated transformation of RDF data. Needless to say triple-to-triple transformations are very restrictive. As [Example 1](#) shows, many construct queries seem to have a legitimate claim to conservativity even though they fall outside of this class. The purpose of the present section is therefore to put the concept of a p-map to more creative use and expand the class of RDF graphs that we recognise as conservative in relation to others. More specifically, we shall generalise the notion of a p-map from a triple-to-triple transformation to a sub-graph to sub-graph transformation, no longer requiring that pairs of vertexes be consistently and non-distortively related by triples—only that they be so related by sub-graphs. The point is to have a transformation that is conservative in much the same sense as a bounded p-map is. That is, the transformation should satisfy the property that if a sub-graph of H is expressed purely in terms of representatives of structural elements of G , then it reflects an actual sub-graph of G . We shall illustrate this approach by considering a function that maps *paths* in G to paths in H . This particular choice is motivated by [Example 1](#) and similar ones which show that many construct queries can indeed be considered as transformations mapping triples to paths, or paths to paths more generally.

Definition 13. *A walk is a non-empty sequence α of triples $\alpha := \langle g_1, g_2, \dots, g_n \rangle$ such that $\pi_3(g_i) = \pi_1(g_{i+1})$ for $1 \leq i < n$. We shall let $\text{len}(\alpha) = n$ denote the length of α , whilst $\text{px}(\alpha) := \pi_1(g_1)$ and $\text{dt}(\alpha) := \pi_3(g_n)$ will denote the proximal and distal vertexes of α , respectively. A cycle is a walk α where $\text{dt}(\alpha) = \text{px}(\alpha)$. A path is a walk where no proper segment forms a cycle.*

Note that we only consider finite paths. The set of paths in an RDF graph G , denoted G^\dagger , is thus finite too. Cycles are allowed, as long as they do not contain smaller cycles, and triples are considered as unary paths (or unary cycles if the vertexes are the same). We next introduce two equivalence relations on paths:

Definition 14. *Let α and β be paths. We define the equivalence relations $=_V$ and $=_E$ on paths as*

1. $\alpha =_V \beta$ iff $\text{px}(\alpha) = \text{px}(\beta)$ and $\text{dt}(\alpha) = \text{dt}(\beta)$
2. $\alpha =_E \beta$ iff α equals β , except that the respective proximal and distal vertexes of α and β may differ.

In the case where α is a path and g a triple we shall abuse this notation slightly, writing $\alpha =_E g$ to mean $\alpha =_E \langle g \rangle$, and similarly for $=_V$.

Clearly, if two paths are both $=_V$ -equivalent and $=_E$ -equivalent, then they are the same path. Neither relation factors blank nodes into the notion of sub-graph equivalence. This is an obvious further development which we shall comment on below. By the use of the relations $=_E$ and $=_V$ it is possible to impose restrictions that intuitively constrain the transformation of paths to paths in the same way that a bound constrains the transformation of triples to triples.

Consider the diagram in **Figure 3**. Here κ_G and κ_H are functions that return the closure under composition—a notion yet to be defined—of the two RDF graphs G and H . We shall say that a bound on maps of paths corresponds to a bound on p-maps if for every f, G^\uparrow and H^\uparrow where $f : G^\uparrow \rightarrow H^\uparrow$, there is an $h : \kappa_G(G^\uparrow) \rightarrow \kappa_H(H^\uparrow)$ such that f satisfies the path map bound iff h satisfies the p-map bound.

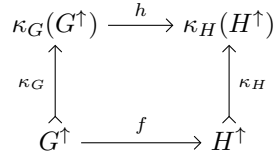


Fig. 3.

If that is the case, then a consistent relation between triples in the top row is reflected by a consistent relation between paths in the lower row, whence f may be used *in loco parentis* for h to measure the conservativeness of H wrt. G . To that end, we next define the closure under composition of an RDF graph, and state a few properties of this operation:

Definition 15. A composition function κ for an RDF graph G is a function of type $\kappa : G^\uparrow \rightarrow \mathcal{T}$ such that

1. $\alpha =_V \kappa(\alpha)$,
2. $\alpha =_E \beta$ iff $\kappa(\alpha) =_E \kappa(\beta)$,
3. $\pi_2(\kappa(\alpha)) = \pi_2(\alpha)$, if $len(\alpha) = 1$, otherwise $\pi_2(\kappa(\alpha)) \notin \mathcal{U}_G$.

According to this definition, a composition function is such that every non-unary path in G is correlated with a triple whose edge is new to G . Essentially for this reason, composition functions always exist:

Lemma 3. There is a composition function for every RDF graph G .

A composition function for G extends G , and puts G^\uparrow and $\kappa(G^\uparrow)$ in one-to-one correspondence, as one would expect:

Proposition 2. If κ is a composition function for G , then $G \subseteq \kappa(G^\uparrow)$.

Lemma 4. A composition function κ for G is a bijection between G^\uparrow and $\kappa(G^\uparrow)$.

Turning to path-maps, that is, to functions of type $f : \mathcal{T}^\uparrow \rightarrow \mathcal{T}^\uparrow$, we are not interested in all path-maps, only those that can be used to ‘emulate’ p-maps. For want of a better name we shall call them c-maps:

Definition 16. A c-map is a path-map f where:

1. the relation $\{(g, g') \mid (\langle g \rangle, \langle g' \rangle) \in f\}$ is a p-map,
2. $\alpha =_V f(\alpha)$,
3. if $\alpha =_E \beta$, then $f(\alpha) =_E f(\beta)$,
4. $len(\alpha) \leq len(f(\alpha))$.

The class of c-maps thus consists of those path-maps that behave like a p-map on unary paths **(1)**, are sensitive to $=_V$ - and $=_E$ -equivalence **(2)**, **(3)**, and never truncate paths **(4)**. The next theorem shows that every c-map of G^\uparrow to H^\uparrow induces a p-map of $\kappa_G(G)$ to $\kappa_H(H)$, for some κ_G and κ_H :

Lemma 5. *Let κ_G, κ_H be composition functions for RDF graphs G and H , respectively. Then every path-map $f : G^\uparrow \rightarrow H^\uparrow$ induces a p-map h_f of $\kappa_G(G^\uparrow)$ to $\kappa_H(H^\uparrow)$, defined by letting h_f be the identity on vertexes and putting $h_f(\pi_2(\kappa_G(\alpha))) = \pi_2(\kappa_H(f(\alpha)))$.*

We now generalise the p-map bounds of [Definition 12](#) to bounds on c-maps:

Definition 17. *Suppose α and β are paths in the RDF graphs G and H , respectively. We shall say that a c-map $f : G^\uparrow \rightarrow H^\uparrow$ is respectively a c1-, c2- or c3-map if one of the following conditions holds:*

$$f(\alpha) =_E \beta \quad \Rightarrow \quad f(\gamma) = \beta \text{ for some } \gamma \in G^\uparrow \quad (\text{c1})$$

$$f(\alpha) =_E \beta \text{ and } \{px(\beta), dt(\beta)\} \cap V_G \neq \emptyset \quad \Rightarrow \quad f(\gamma) = \beta \text{ for some } \gamma \in G^\uparrow \quad (\text{c2})$$

$$f(\alpha) =_E \beta \text{ and } \{px(\beta), dt(\beta)\} \subseteq V_G \quad \Rightarrow \quad f(\gamma) = \beta \text{ for some } \gamma \in G^\uparrow \quad (\text{c3})$$

That these bounds are in fact generalisations of those of [Definition 12](#) is established by the following theorem:

Theorem 7. *Let f be a c-map of RDF graph G to RDF graph H , κ_G and κ_H composition functions for G and H , respectively, and h_f the induced p-map of $\kappa_G(G^\uparrow)$ to $\kappa_H(H^\uparrow)$. Then f is a cn-map iff h_f is a pn-map, for $n = 1, 2$ or 3 .*

Proof. Suppose f is a c3-map and assume that there is a triple $g := \langle a, h_f(p), b \rangle \in \kappa_H(H^\uparrow)$ where $a, b \in V_G$. We need to show that $\langle a, p, b \rangle \in \kappa_G(G^\uparrow)$. Given that κ_G and κ_H are surjective by [Lemma 4](#), let $\alpha \in G^\uparrow$ and $\beta \in H^\uparrow$ be such that $\pi_2(\kappa_G(\alpha)) = p$ and $\kappa_H(\beta) = g$. By the definition of h_f given in [Lemma 5](#), $f(\alpha) =_E \beta$, so by bound (c3) there is a $\gamma \in G^\uparrow$ where $f(\gamma) = \beta$. We have $g =_V \beta =_V \gamma$ by [Definition 16](#) and [Definition 15](#), and $\gamma =_E \alpha$ by [Definition 16](#), since $f(\alpha) =_E \beta$. This means that $\kappa_G(\gamma) = \langle a, p, b \rangle$. For the converse direction, suppose h_f is a p3-map and assume, for some $\alpha \in G^\uparrow$ and $\beta \in H^\uparrow$, that $f(\alpha) =_E \beta$ and $a, b \in V_G$, where $a := px(\beta)$ and $b := dt(\beta)$. By the definition of h_f we have $h_f(\pi_2(\kappa_G(\alpha))) = \pi_2(\kappa_H(\beta))$, so let $\kappa_H(\beta) := \langle a, h_f(p), b \rangle$. Since h_f satisfies (p1), we have $\langle a, p, b \rangle \in \kappa_G(G^\uparrow)$. By [Lemma 4](#), there is a $\gamma \in G^\uparrow$ such that $\kappa_G(\gamma) = \langle a, p, b \rangle$. Since $\kappa_G(\gamma) =_E \kappa_G(\alpha)$, we have $f(\gamma) =_E f(\alpha)$, and given that $\gamma =_V f(\gamma) =_V \beta$, we arrive at $f(\gamma) = \beta$. It is easy to adjust the membership of $a, b, px(\beta), dt(\beta)$ in V_G in the proof and confirm the claim for two other pair of corresponding bounds. \square

Expanding the class of conservative construct queries to also handle paths requires the following generalisation of [Theorem 6](#):

Theorem 8. *Let $\langle C, S \rangle$ be a construct query, where C contains no blank nodes and no variables as edges. If f is a cn-map of S to C , then f is a cn-map of $\langle S, S \rangle(G)$ to $\langle C, S \rangle(G)$, for $n = 1, 2, 3$.*

Proof (Sketch). Let α be a chain in S and let f be a c-map of C to S . If $f(\alpha)$ contains blank nodes then, due to the relabelling function ρ , $f(\alpha)$ may be instantiated differently for each pair of objects that matches the vertexes

of α . This means we may not have $\mu(\rho_\mu(f(\alpha))) =_E \mu'(\rho_{\mu'}(f(\alpha)))$, whence f is not a c -map of $\langle S, S \rangle(G)$ to $\langle C, S \rangle(G)$. In the absence of blank nodes in C this situation cannot arise, ρ becomes redundant, and the proof becomes a straightforward generalisation of that for [Theorem 6](#). \square

As this proof-sketch is designed to show, extra care is required when the construct query C contains blank nodes—as it does for instance in [Example 1](#). However, the preceding lemmata and theorems lay out all the essential steps. More specifically, all that is needed in order to accommodate [Example 1](#) and similar ones, is to substitute equivalence classes of paths for paths throughout, where equivalence is equality up to relabelling of blank nodes. The verification of this claim is a rerun with minor modifications, and has therefore been left out.

6 Computational Properties

The problem of deciding whether there exists a homomorphism between two (standard) graphs is well-known to be NP-complete. Since p -maps are more restricted than generic graph homomorphisms, identifying p -maps between RDF graphs is an easier task. In fact it can be done in polynomial time, the verification of which is supported by the following lemmata:

Lemma 6. *Let h_1 and h_2 be p -maps of G_1 and G_2 respectively to H . Then $h_1 \cup h_2$ is a p -map of $G_1 \cup G_2$ to H if $h_1(u) = h_2(u)$ for all $u \in \text{dom}(h_1) \cap \text{dom}(h_2)$.*

Lemma 7. *If h_1, h_2 are bounded p -maps such that $h_1(u) = h_2(u)$ for all $u \in \text{dom}(h_1) \cap \text{dom}(h_2)$. Then $h_1 \cup h_2$ is a bounded p -map satisfying the weaker of the two bounds.*

According to [Lemma 6](#) the task of finding a p -map of G to H can be reduced to the task of finding a set of p -maps of *sub-graphs* of G into H that are compatible wrt. to shared domain elements. [Lemma 7](#) then tells us that to check whether the resulting p -map is bounded by some bound pn , it suffices to check whether each of the smaller maps is. This procedure, each step of which is clearly polynomial, does not require any backtracking, whence:

Theorem 9. *Given two RDF graphs G and H , finding a p -map $h : G \rightarrow H$, bounded or not, is a problem polynomial in the size of G and H .*

Proof (Sketch). For any RDF graphs G and H , fix the set V_G of nodes occurring as vertexes in G . Then for each $p \in E_G$ construct a p -map of $G_p := \{\langle a, p', b \rangle \in G \mid p' = p\}$ into H . This amounts to iterating through the edges of H and finding one, say q , such that i) $\langle a, p, b \rangle \in G_p \rightarrow \langle a, q, b \rangle \in H_q$ and ii) if $p \in V_G$ then $p = q$. [Lemma 6](#) tells us that the union of these maps is a p -map of G to H , i.e. no choice of q for p is a wrong choice. There is therefore no need for backtracking, whence a p -map can be computed in polynomial time. To check whether it satisfies a given bound pn , it suffices by [Lemma 7](#) to check that each of the maps h_p of G_p to H_q does. That is, for each element $\langle a, q, b \rangle \in H_q \setminus G_p$ check that the required triple is in G_p . This is clearly a polynomial check. \square

Input : RDF graphs G and H , bound pn .

Output: A p-map h bounded by pn , or \perp if none exists.

$h := \{\langle a, a \rangle\}$ for $a \in V_G$;

```

for  $p \in E_G$ 
  if  $p \in V_G$ 
    if  $\langle a, p, b \rangle \in G \rightarrow \langle a, p, b \rangle \in H$ 
      for  $\langle c, p, d \rangle \in H \setminus G$ 
        if not  $\text{CheckBound}(\langle c, p, d \rangle, pn)$  return  $\perp$ ;
      else return  $\perp$ ;
    else
      bool found := false;
      for  $q \in E_H$ 
        if not found and  $\langle a, p, b \rangle \in G \rightarrow \langle a, q, b \rangle \in H$ 
          for  $\langle c, q, d \rangle \in H \setminus G$ 
            if not  $\text{CheckBound}(\langle c, q, d \rangle, pn)$  break;
           $h := h \cup \{\langle p, q \rangle\}$ ;
          found := true;
        if not found return  $\perp$ ;
  return  $h$ ;

```

Algorithm 1. Computing a bounded p-map if one exists. The check for compliance with the bounds is encapsulated in a boolean subroutine **CheckBound**.

For c-maps the situation is more complex. Since the composition of an RDF graph may be exponentially larger than the graph itself, the problem is no longer polynomial. More precisely, if G is an RDF graph and κ a composition function for G , then $|\kappa(G^\dagger)| \leq \sum_{n=1}^{|V_G|} |E_G| \times n!$. Yet, this is not a problem for any realistically sized construct query. An experimental application is up and running at <http://sws.ifi.uio.no/MapperDan/>. Mapper Dan takes two RDF graphs or a construct query as input, lets the user specify which bounds to apply to which predicates, and checks whether there is a map under the given bound between the two graphs or between the **WHERE** and **CONSTRUCT** block of the construct query. In the cases where a bound is violated Mapper Dan offers guidance, if possible, as to how to obtain a stratified map which satisfies the bounds. A map can be used to translate the source RDF data to the target vocabulary, produce a construct query which reflects the map, or to rewrite SPARQL queries.

7 Conclusion and Future Work

This paper provides a structural criterion that separates conservative from non-conservative uses of SPARQL construct queries. Conservativity is here measured against the ‘asserted content’ of the underlying source, which is required to be preserved by the possible change of vocabulary induced by the construct clause. Our problem led us to consider a class of RDF homomorphisms (p-maps) the existence of which guarantees that the source and target interlock in a reciprocal simulation. Viewed as functions from triples to triples, p-maps are computable in

polynomial time. The complexity increases with more complex graph patterns. The class of p-maps has other applications besides that described here, e.g as the basis for a more refined notion of RDF merging. As of today merging is based on the method of taking unions modulo the standardising apart of blank nodes. If one also wants a uniform representation of the data thus collected this method is too crude. What one would want, rather, is a way of transforming the data by swapping vocabulary elements whilst, as far as it goes, preserving the information content of all the involved sources (this is not easily achieved by subsuming a set of properties or types under a common super-type in an ontology). Such a merge procedure may turn out to be an important prerequisite for truly RESTful write operations on the web of linked data.

Acknowledgements. This research was partially funded by the Norwegian Research Council through the Semicolon II project (<http://www.semicolon.no/>).

References

1. Arenas, M., Gutierrez, C., Pérez, J.: Foundations of RDF Databases. In: Tessaris, S., Franconi, E., Eiter, T., Gutierrez, C., Handschuh, S., Rousset, M.-C., Schmidt, R.A. (eds.) Reasoning Web. LNCS, vol. 5689, pp. 158–204. Springer, Heidelberg (2009)
2. Baget, J.-F.: RDF Entailment as a Graph Homomorphism. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 82–96. Springer, Heidelberg (2005)
3. Bleiholder, J., Naumann, F.: Data fusion. ACM Computing Surveys 41(1) (2008)
4. Fagin, R., Kolaitis, P.G., Miller, R.J., Popa, L.: Data exchange: semantics and query answering. Theoretical Computer Science 336, 89–124 (2005)
5. Ghilardi, S., Lutz, C., Wolter, F.: Did I Damage my Ontology? A Case for Conservative Extensions in Description Logics. In: Proc. of the 10th Int. Conference on Principles of Knowledge Representation and Reasoning, KR 2006 (2006)
6. Hayes, P.: RDF Semantics. W3C Recommendation, W3C (2004), <http://www.w3.org/TR/rdf-mt/>
7. Hutter, D.: Some Remarks on the Annotation %cons (1999)
8. Makinson, D.C.: Logical Friendliness and Sympathy in Logic. In: Logica Universalis. Birkhäuser, Basel (2005)
9. Pérez, J., Arenas, M., Gutierrez, C.: Semantics of SPARQL. Technical report, Universidad de Chile, TR/DCC-2006-17 (2006)
10. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. W3C Recommendation, W3C (2008), <http://www.w3.org/TR/rdf-sparql-query/>
11. Stolpe, A., Skjæveland, M.G.: From Spreadsheets to 5-star Linked Data in the Cultural Heritage Domain: A Case Study of the Yellow List. In: Norsk Informatikkonferanse (NIK 2011), Tapir, pp. 13–24 (2011)

Assessing Linked Data Mappings Using Network Measures

Christophe Guéret¹, Paul Groth¹, Claus Stadler², and Jens Lehmann²

¹ Free University Amsterdam, De Boelelaan 1105, 1081HV Amsterdam
{c.d.m.gueret,p.t.groth}@vu.nl

² University of Leipzig, Johannisgasse 26, 04103 Leipzig
{cstadler,lehmann}@informatik.uni-leipzig.de

Abstract. Linked Data is at its core about the setting of links between resources. Links provide enriched semantics, pointers to extra information and enable the merging of data sets. However, as the amount of Linked Data has grown, there has been the need to automate the creation of links and such automated approaches can create low-quality links or unsuitable network structures. In particular, it is difficult to know whether the links introduced improve or diminish the quality of Linked Data. In this paper, we present LINK-QA, an extensible framework that allows for the assessment of Linked Data mappings using network metrics. We test five metrics using this framework on a set of known good and bad links generated by a common mapping system, and show the behaviour of those metrics.

Keywords: linked data, quality assurance, network analysis.

1 Introduction

Linked Data features a distributed publication model that allows for any data publisher to semantically link to other resources on the Web. Because of this open nature, several mechanisms have been introduced to semi-automatically link resources on the Web of Data to improve its connectivity and increase its semantic richness. This partially automated introduction of links begs the question as to which links are improving the *quality* of the Web of Data or are just adding clutter. This notion of quality is particularly important because unlike the regular Web, there is not a human deciding based on context whether a link is useful or not. Instead, automated agents (with currently less capabilities) must be able to make these decisions.

There are a number of possible ways to measure the quality of links. In this work, we explore the use of network measures as one avenue of determining the quality. These statistical techniques provide summaries of the network along different dimensions, for example, by detecting how interlinked a node is within in a network [3]. The application of these measures for use in quality measurement is motivated by recent work applying networks measures to the Web of Data [11].

Concretely, we pose the question of whether network measures can be used to detect changes in quality with the introduction of new links (*i.e.* mappings)

between datasets. We test 5 network measures, three classic network measures (degree, centrality, clustering coefficient) and two network measures designed specifically for Linked Data (Open SameAs chains, and Description Richness). The experiments are performed on link sets produced by Silk [23], a state-of-the-art mapping tool. We show that at this time such network measures are only partially able to detect quality links. We discuss reasons for this and sketch a possible path forward.

In addition to these experiments, we present an extensible framework, LINK-QA, for performing such network analysis based quality assessment. The framework allows for both the execution and reporting of quality measurements. Our contributions are twofold:

1. a framework, LINK-QA, for measuring quality of topological modifications to Linked Data; and
2. analysis of five network measures for the applicability in testing link quality.

The rest of this paper is organized as follows. We begin with some preliminary definitions of the the networks we analyze. The metrics we test are defined in Section 3. This is followed 4 by a description of the framework for quality assessment. This includes a discussion of a reference implementation. Experimental results on a set of automatically generated links are reported on in Section 5. Finally, we discuss related work and conclude.

2 Network Definitions

We now introduce the definitions used throughout this paper. The graph we want to study will be referred to as the *Data Network*. It is the network of facts provided by the graph of the Web of Data, excluding the blank nodes.

Definition 1 (Data Network). *The data network is defined as a directed, labelled, graph $\mathcal{G} = \{V, E, L\}$ with V a set of nodes, E a set of edges and L a set of labels. An edge e_{ij} connects a node $v_i \in V$ to the node $v_j \in V$ with a label $l(e_{ij}) \in L$. The edges and labels correspond to the triples and predicates of the Web of Data.*

In this paper, we sample the Data Network by collecting information about the neighbourhood of selected sets of resources within it. A resource’s neighbourhood consists of a direct neighbourhood and an extended neighbourhood:

Definition 2 (Resource Neighbourhood). *The direct neighbourhood of a node i is defined as the set of nodes directly connected to it through either an incoming edge (N_i^-) or outgoing edge (N_i^+). That is, $N_i = N_i^+ \cup N_i^- = \{v_j \mid e_{ij} \in E\} \cup \{v_j \mid e_{ji} \in E\}$. Its extended neighbourhood N_i^* also include neighbours’ neighbours which are not i : $N_i^* = N_i \cup \bigcup_{v_j \in N_i} N_j$.*

A resource neighbourhood is used to build a local network around a particular resource from the Data Network.

Definition 3 (Local Network). The local network $\mathcal{G}_i = \{V_i, E_i, L_i\}$ of a node $v_i \in V$ is a directed, labeled graph of the extended neighbourhood of v_i . The set of nodes is defined as $V_i = N_i^*$, the edges are $E_i = \{e_{jk} \in E \mid (v_j, v_k) \in N_i^* \times N_i^*\}$ and the labels $L_i = \{l(e_{jk}) \mid e_{jk} \in E_i\}$.

Figure 1 shows an example of a local network for which \bullet is v_i , \bullet are the nodes in N_i and \bullet are the nodes in N_i^* . Local networks created around nodes from \mathcal{G} are the focus of the analysis performed by our framework. It is worth noting that the union of all local neighbourhoods of every node in \mathcal{G} is equivalent to this graph. That is, $\mathcal{G} \equiv \bigcup_{v_i \in N} \mathcal{G}_i$.

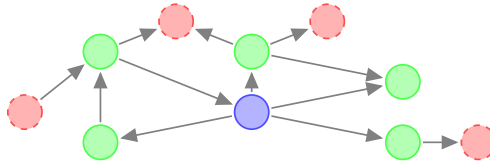


Fig. 1. Example of direct and extended neighbourhood around a source node graphics

3 Network Metrics

Based on the above definitions, we now detail a set of 5 network metrics to use in quality detection. Relevant prior work on network analysis was the key criteria to establish these metrics. Although Linked Data networks are different to social networks, we used them as starting point. The degree, clustering coefficient and centrality measures are justified as measures of network robustness [10]. The other two metrics are based on studies of the Web of Data as a network that show that fragmentation of the SameAs network is common and thus may be a sign of low quality [2].

In specifying these metrics, one must not only define the measure itself but also what constitutes quality with respect to that measure. Defining such a “quality goal” is difficult as we are only beginning to obtain empirical evidence about what network topologies map to qualitative notions of quality [10]. To address this problem, for each metric, we define an ideal and justify it with respect to some well-known quality notions from both network science and Linked Data publication practice. We consider the following to be broad quality goals that should be reached by the creation of links: 1. modifications should bring the topology of the network closer to that of a power law network to make the network more robust against random failure; 2. modifications should lower the differences between the centrality of the hubs in the network to make the network more robust against targeted failure of these critical nodes; 3. modifications should increase the clustering within topical groups of resources and also lower the average path length between groups (*i.e.* foster a small world network).

We now discuss the 5 metrics (degree, clustering coefficient, open sameAs chains, centrality, description richness). We describe the measure itself as well as the ideal (*i.e.* goal) distribution associated to it. Each metric is designed to fulfill the following criteria:

1. be computable using the local network of a resource;
2. be representative of a global network property;
3. be able to identify particular parts of the network that relate to an ideal distribution of that metric;
4. have a domain within positive real values (the metrics described here produce values between 0 and a factor of N).

We note that local networks can have variable size and thus may not be independent of each other as there may be overlaps. The network measures themselves are aggregations of those at the local level. In the development of these metrics, we attempt to ensure that the metrics are not sensitive to local level effects.

3.1 Degree

This measures how many hubs there are in a network. The aim is to have a network which allows for fast connectivity between different parts of the network. Thus making it easier for automated agents to find a variety of information through traversal. Power-law networks are known to be robust against random failure and are a characteristic of small world networks [1].

Measure. The degree of a node is given by its number of incoming and outgoing edges.

$$m_i^{degree} = \|\{e_{ij} \mid v_j \in N_i, e_{ij} \in E_i\}\| + \|\{e_{ji} \mid v_j \in N_i, e_{ji} \in E_i\}\|$$

Ideal. We aim at a degree distribution that follows some power-law $P(k) \sim ck^{-\gamma}$ where $P(k)$ is the probability of finding a node with a degree k and c, γ are two distribution parameters. Power-law degree distributions are a sign of robustness against random failure and one of the characteristics of small world networks. The distance between the degree distribution and its ideal is defined as the absolute difference between the observed distribution and its closest power-law equivalent obtained through fitting.

$$d^{degree} = \sum_k abs\left(\frac{\|\{v_i \mid m_i^{degree} = k\}\|}{\|N_i\| + 1} - ck^{-\gamma}\right)$$

3.2 Clustering Coefficient

The clustering coefficient is a measure of the denseness of the network. The metric measures the density of the neighbourhood around a particular node.

Measure. The local clustering coefficient of a node n_i is given by the ratio between the number of links among its neighbours and the number of possible links.

$$m_i^{clustering} = \frac{\|\{e_{jk} \mid v_j, v_k \in N_i, e_{jk} \in E_i\}\|}{\|N_i\|(\|N_i\| - 1)}$$

Ideal. The highest average clustering coefficient a network can have is 1, meaning that every node is connected to every other node (the network is said to be “complete”). Although this is a result the Web of Data should not aim at, as most links would then be meaningless, an increase of the clustering coefficient is a sign of cohesiveness among local clusters. The emergence of such topic oriented clusters are common in the Web of Data and are in line with having a small world. We thus set an average clustering coefficient of 1 as a goal and define the distance accordingly. S being the set of all resources, the distance to the ideal is 1 minus the average clustering coefficient of the nodes in S .

$$d^{clustering} = 1 - \frac{1}{\|S\|} \sum_{v_i \in S} m_i^{clustering}$$

3.3 Centrality

Commonly used estimates of the centrality of a node in a graph are *betweenness centrality*, *closeness centrality*, and *degree centrality*. All these values indicate the critical position of a node in a topology. For this metric, we focus on betweenness centrality, which indicates the likelihood of a node being on the shortest path between two other nodes. The computation of betweenness centrality requires knowing the complete topology of the studied graph. Because our metrics are node-centric and we only have access to the local neighbourhood, we use the ratio of incoming and outgoing edges as a proxy.

Measure. The centrality of a node v_i is given by the number of connections it takes part in. This value is obtained by the product between the number of nodes reaching v_i through its incoming neighbours, and the number of nodes reachable through the outgoing neighbours.

$$m_i^{centrality} = \frac{\|\{v_k \mid e_{kj} \in E_i, v_j \in N_i^+\}\|}{\|\{v_k \mid e_{jk} \in E_i, v_j \in N_i^-\}\|}$$

Ideal. A network dominated by highly central points is prone to critical failure in case those central points cease to operate or are being renamed [10]. Ideally, the creation of new links would reduce the overall discrepancy among the centrality values of the nodes. This means decreasing the centrality index of the graph:

$$d^{centrality} = \sum_{i \in V} \frac{\max_{j \in V} (m_j^{centrality}) - m_i^{centrality}}{\|V\| - 1}$$

3.4 SameAs Chains

The very common `owl:sameAs` property can be improperly asserted. One way to confirm a given sameAs relation is correct is to find closed chains of sameAs relations between the linking resource and the resource linked. This metric detects whether there are open sameAs chains in the network.

Measure. The metric counts the number of sameAs chains that are not closed. Let $p_{ik} = \{e_{ij_1}, \dots, e_{j_y k}\}$ be a path of length y defined as a sequence of edges with the same label $l(p_{ik})$. The number of open chains is defined as

$$m_i^{paths} = \|\{p_{ik} \mid l(p_{ik}) = \text{"owl:sameAs"}, k \neq i\}\|$$

As highlighted earlier, metrics should not be sensitive on scaling effects when going from the local definition to their global value. This metric is not sensitive under the assumption that there are few long sameAs chains in the global network [12].

Ideal. Ideally, we would like to have no open sameAs chains in the WoD. If the new links contribute to closing the open paths, their impact is considered positive.

$$d^{paths} = \sum_{v_i \in V} m_i^{paths}$$

3.5 Descriptive Richness through SameAs

This metric measures how much to the description of a resource is added through the use of sameAs edges. If a sameAs edge is introduced, we can measure whether or not that edge adds to the description.

Measure. The measure counts the number of new edges brought to a resource through the sameAs relation(s). This initial set of edges is defined as $A_i = \{e_{ij} \mid l(e_{ij}) \neq \text{"owl:sameAs"}, j \in N_i^+\}$ the number of edges brought to by the neighbours connected through a sameAs relation defined as $B_i = \{e_{jl} \mid v_l \in N_j^+, l \neq i, e_{ij} \in N_i^+, l(e_{ij}) = \text{"owl:sameAs"}\}$ Finally, the gain is the difference between the two sets

$$m_i^{description} = B_i \setminus A_i$$

Ideal. A resource's outgoing sameAs relations ideally link to resources that have a complementary description to the original one. Therefore, the richer the resulting description, the lower the distance to our ideal.

$$d^{description} = \sum_{i \in V} \frac{1}{1 + m_i^{description}}$$

4 LINK-QA Analysis Framework

The above metrics were tested using LINK-QA, a framework for assessing the quality of Linked Data using network metrics. The framework is scalable and extensible: the metrics applied are generic and share a common set of basic requirements, making it easy to incorporate new metrics. Additionally, metrics are computed using only the local network of a resource and are thus parallelisable by design. The framework consists of five components, “**Select**”, “**Construct**”, “**Extend**”, “**Analyse**” and “**Compare**”. These components are assembled together in the form of a workflow (see Figure 2).

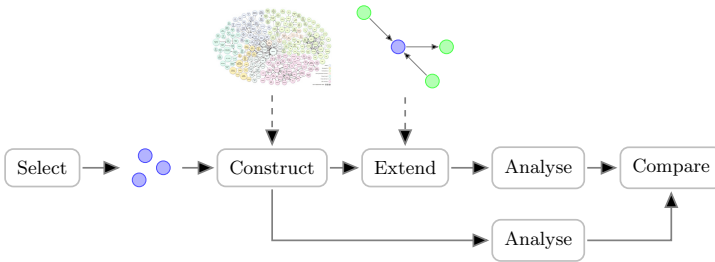


Fig. 2. Interaction between the different components of LINK-QA. The external inputs are indicated in dashed lines pointing towards the processes (rounded box) using them.

4.1 Components

Select. This component is responsible for selecting the set of resources to be evaluated. This can be done through a variety of mechanisms including sampling the Web of Data, using a user specified set of resources, or looking at the set of resources to be linked by a link discovery algorithm. It is left to the user to decide whether the set of resources is a reasonable sample of the Data Network.

Construct. Once a set of resources is selected, the local network, as defined in Definition 2, is constructed for each resource. The local networks are created by querying the Web of Data. Practically, LINK-QA makes use of either SPARQL endpoints or data files to create the graph surrounding a resource. In particular, sampling is achieved by first sending a SPARQL query to a list of endpoints. If no data is found, LINK-QA falls back on de-referencing the resource.

Extend. The “Extend” component adds new edges that are provided as input to the framework. These input edges are added to each local network where they apply. Once these edges are added, we compute a set of new local networks around the original set of selected resources. The aim here is to measure the impact of these new edges on the overall Data Network. This impact assessment is done by the Compare component.

Analyse. Once the original local network and its extended local networks have been created, an analysis consisting of two parts is performed:

1. A set of metrics m is performed on each node v_i within each local network. This produces a set of metric results $m_i^{metric\ name}$ for each node v_i .
2. Metric results obtained per node are aggregated into a distribution. Note, that this distribution converges to the distribution of the overall Data Network as more resources are considered.

Compare. The result coming from both analyses (before and after adding the new edges) are compared to ideal distributions for the different metrics. The comparison is provided to the user.

4.2 Implementation

The implementation is available as free software at <http://bit.ly/Linked-QA>, and takes as input a set of resources, information from the Web of Data (*i.e.* SPARQL endpoints and/or de-referencable resources) and a set of new triples to perform quality assessment on. The implementation is written in Java and uses Jena for interacting with RDF data. In particular, Jena TDB is used to cache resource descriptions. Any23 is used for dereferencing data in order to get a good coverage of possible publication formats.

The implementation generates HTML reports for the results of the quality assessment. These reports are divided in three sections:

1. An overview of the status of the different metrics based on the change of distance to the ideal distribution when the new links are added. The status is “green” if the distance to the ideal decreased and “red” otherwise. The relative change is also indicated. These statuses are derived from the change in $d^{metric\ name}$ observed when adding new links.
2. One graph per metric showing the distribution of the values for the different $m^{metric\ name}$ values obtained before and after adding the new set of links. The rendering of these graphs is done by the Google Graph API.
3. A table reporting for all of the metrics the resources for which the score $m_i^{metric\ name}$ has changed most after the introduction of the new links.

It is important to note that LINK-QA is aimed at analysing a set of links and providing insights to aid manual verification. There is no automated repair of the links nor an exact listing of the faulty links. Outliers - resources that rank farthest from the ideal distribution for a metric - are pointed out, but the final assessment is left to the user.

5 Metric Analysis

The framework is designed to analyse the potential impact of a set of link candidates prior to their publication on the Web of Data. To evaluate this, we test the links produced by a project using state of the art link generation tools: The European project LOD Around the Clock (LATC) aims to enable the use of the Linked Open Data cloud for research and business purposes. One goal of

the project is the publication of new high quality links. LATC created a set of linking specifications (link specs) for the Silk engine, whose output are link sets. In order to assess the correctness of link specs, samples taken from the generated links are manually checked. This results in two reference sets containing all the positive (correct, good) and negative (incorrect, bad) links of the sample. The link specs along with the link sets they produce, and the corresponding manually created reference sets are publicly available¹. Based on these link sets we performed experiments to answer the following questions:

1. Do positive linksets decrease the distance to a metric’s defined ideal, whereas negative ones increase it? If that is the case, it would allow us to distinguish between link sets having high and low ratios of bad links.
2. Is there a correlation between outliers and bad links? If so, resources that rank farthest from the ideal distribution of a metric would relate to incorrect links from/to them.

5.1 Impact of Good and Bad Links

To try and answer the first question, we performed the following experiment: out of 160 link specifications created by LATC, we selected the 6 link sets (*i.e.* mappings) for which the manual verification of the links led to at least 50 correct and incorrect links. For each specification, we took separate random samples of 50 links from the positive and negative reference sets and ran LINK-QA. This was repeated ten times. Table 1 shows the aggregated results for each metric on positive and negative reference sets. The LATC link specification used to create the links are used as identifiers in the tables. The outcome of the ten runs is aggregated into three categories as follows: if no changes were detected in the results distributions, the category is “blank (B)”; a “C” is granted to link specification for which all the (in)correct were detected in at least half (5 in this case) of the runs. Least successful experiments are classified as “I”.

Table 1. Detection result for each metric for both good and bad links. Blank - no detection, I - Incorrect detection, C - correct detection. (lgd = linkedgeodata).

	Centrality		Clustering		Degree		Description		SameAs	
	Good	Bad	Good	Bad	Good	Bad	Good	Bad	Good	Bad
linkedct-pubmed-disease			I	C	I	C	C	I	I	C
gho-linkedct-disease					C	I	C	I		
gho-linkedct-country			I	C	I	C				
geonames-lgd-island	C	C			I	I	C	C	C	C
gho-pubmed-country			I	I	I	I	C	I	I	C
geonames-lgd-mountain	C	I			I	C	C	I	I	C

¹ <https://github.com/LATC/24-7-platform/tree/master/link-specifications>

A global success rate can be quickly drawn from Table [1](#) by considering the cumulative number of “C” and “I” to compute a recall score.

$$recall = \frac{\|I\| + \|C\|}{\|B\| + \|I\| + \|C\|} = \frac{21 + 20}{19 + 21 + 20} = 0.68$$

A precision index is given by the ratio between “C” and “I”.

$$precision = \frac{\|C\|}{\|I\| + \|C\|} = \frac{20}{21 + 20} = 0.49$$

These two values indicate a mediocre success of our metrics on these data sets. From the table and these values, we conclude that common metrics such as centrality, clustering, and degree are insufficient for detecting quality. Additionally, while the Description Richness and Open SameAs Chain metrics look more promising, especially at detecting good and bad links, respectively, they report too many false positives for reference sets of the opposite polarity.

We now present a more in-depth analysis of the results found in the table focusing on the sensitivity of the metrics, their detection accuracy and their agreement.

Sensitivity of Metrics. The presence in Table [1](#) of blank fields indicates that the metric was not able to detect any change in the topology of the neighbourhood of resources, meaning that it fails at the first goal. We realise that the Degree metric is the only one to always detect changes. A behaviour that can be explained by the fact that adding a new link almost always yields new connections and thus alters the degree distribution.

The low performance of other metrics in detecting change can be explained by either a lack of information in the local neighbourhood or a stable change. The first may happen in the case of metrics such as the sameAs chains. If no sameAs relations are present in the local network of the two resources linked, there will be no chain modified and, thus, the metric will not detect any positive or negative effect for this new link. A stable change can happen if the link created does not impact the global distribution of the metric. The results found in Table [1](#) report changes in distributions with respect to the ideals defined, if the distribution does not change with the addition of the links the metrics is are ineffective.

Accuracy of Detection. With 21 “I” and 20 “C” in Table [1](#), we found as many incorrect results as correct ones. This result is unfortunately not good enough base decisions upon. There are a number of possible reasons for this low performance. It may be the case that network measures are not applicable at this level of network size. Indeed, a much larger network may be necessary for summarization effects to actually be applicable. Furthermore, the selected metrics may be inappropriate for Linked Data. Here, we enumerate 3 possible reasons.

1. Definition of ideals: The ideals are some target distribution we set as a universal goal Linked Data should aim for. It is however unclear whether such a unique goal can be set for Linked Data. Our inspiration from social networks

led us to aim at a small world topology, which does not correlate with the results found in our experiments; 2. **Coverage of sample:** The use of a sample of the studied network forces us to consider a proxy for the actual metrics we would have had computed on the actual network. Most noticeably, the centrality measure featured in our prototype is a rough approximation. For this metric in particular, a wider local neighbourhood around a resource would lead to better estimates. The same applies to the detection of sameAs chains which may span outside of the local neighbourhood we currently define; 3. **Validity of metrics:** The somewhat better performance of Linked Data specific network measures suggests that such tailored metrics may be a more effective than “class” metrics. The degree, clustering and centrality metrics look at the topology of the network without considering its semantics. However, as it is confirmed by our experiments, the creation of links is very much driven by these semantics and the eventual changes in topology do not provide us with enough insights alone. Our intuition, to be verified, is that effective metrics will leverage both the topological and semantic aspect of the network.

We believe a future path forward is to gain more empirical evidence for particular topologies and their connection to quality. The sampling of the Web of Data will also have to be reconsidered and may need to be defined with respect to a particular metric.

5.2 Detection of Bad Links

Our second research question is, whether LINK-QA can detect bad links in link sets with only a few bad links. Here, we are seeking a correlation between the ranking of outliers and the resources that are subjects of bad links. For this experiment, we took all LATC link specs with at least 50 positive and 10 negative reference links, and created samples of 45 positive and 5 negative links. LINK-QA was then run, and the process was repeated 5 times. Figure 3 shows the number of correct detections of outliers. With the exception of the cluster coefficient, the metrics show a bias for negative resources to be identified as outliers. Although the remaining distributions do not seem directly suitable

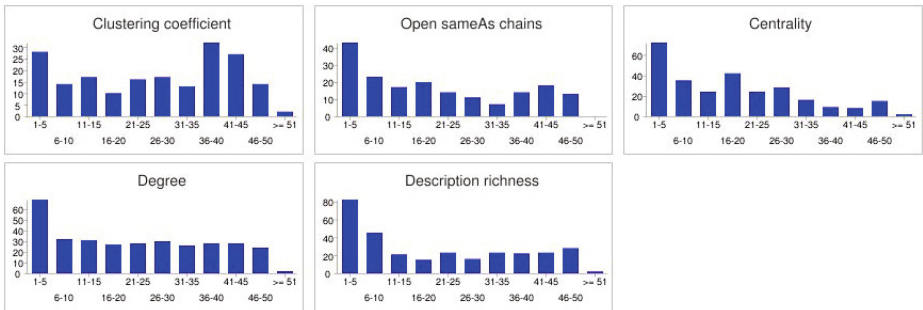


Fig. 3. Summary of outlier analysis. x-axis: rank of resources grouped in buckets of 5 (low values indicate outliers). y-axis: resource count.

for detecting bad links, they show a trend in this direction, indicating that the predictions could be improved with the combination of multiple metrics. We exclude the cluster coefficient from the following analysis. Given a ranking of resources for each metric, we can assign each resource a sorted list of its ranks, e.g. *Fabulous_Disaster* \rightarrow (3, 3, 10, 17). A resource’s n -th rank, considering $n = 1 \dots 4$ metrics, is then determined by taking the $n - 1$ -th element of this list. Ideally, we would like to see negative resources receiving smaller n -th ranks than positive ones. The distributions of the n -th ranks for all n s are shown in Figure 4. These charts indicate that a combination indeed improves the results: For example when combining 2 metrics, the probability of finding a negative resource on one of the first 5 ranks increases from about 20 to 30 percent, whereas an equi-distribution would only yield 10 percent (5 negative resources in 50 links). This effect increases, as can be observed in the right column: The positive-to-negative ratio is 0.6 for $n = 4$, which shows that a combination of metrics is effective in detecting incorrect links.

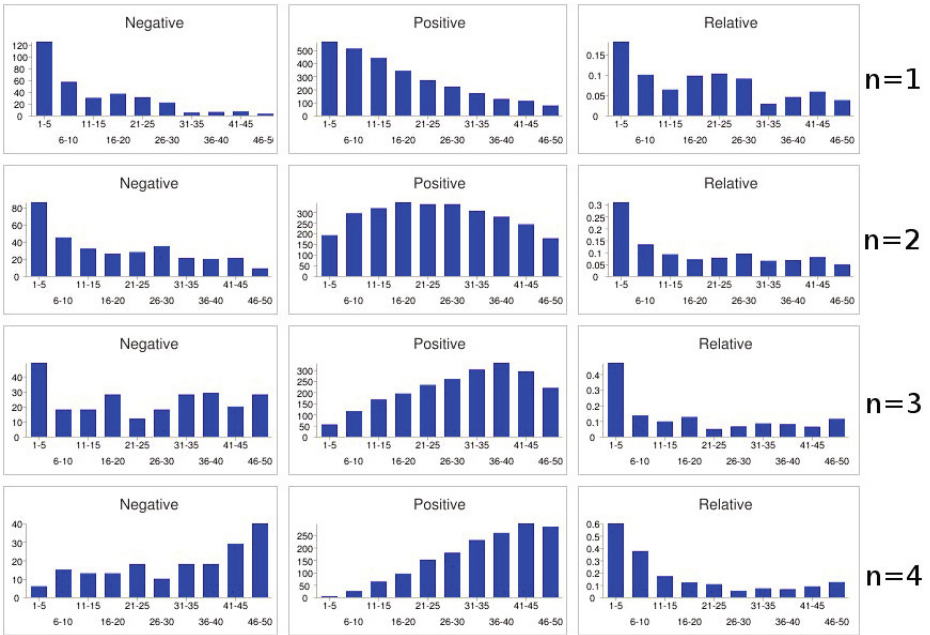


Fig. 4. Distribution of negative and positive resources by its n -th rank. For “negative” and “positive”, the y-Axis shows the absolute number of resources detected for every bucket of five ranks. For “relative” it shows the ratio of negative to positive links.

6 Related Work

In this section, we provide a review of related work touching on this paper. We particularly focus on quality with respect to the Semantic Web but also briefly touch on Network Analysis and the automated creation of links.

6.1 Quality

Improving data quality has become an increasingly pressing issue as the Web of Data grows. For example, the Pedantic Web group has encouraged data providers to follow best practices [15]. Much of the work related to quality has been on the application information quality assessment on the Semantic Web. In the WIQA framework [4], policies can be expressed to determine whether to trust a given information item based on both provenance and background information expressed as Named Graphs [5]. Hartig and Zhao follow a similar approach using annotated provenance graphs to perform quality assessment [19]. Harth *et al.* [13] introduce the notion of naming authority to rank data expressed in RDF based on network relationships and PageRank.

Trust is often thought as being synonymous with quality and has been widely studied including in artificial intelligence, the web and the Semantic Web. For a readable overview of trust research in artificial intelligence, we refer readers to Sabater and Sierra [20]. For a more specialized review of trust research as it pertains to the Web see [8]. Artz and Gil provide a review of trust tailored particularly to the Semantic Web [2]. Specific works include the IWTrust algorithm for question answering systems [24] and tSPARQL for querying trust values using SPARQL [14]. Our approach differs from these approaches in that it focuses on using network measures to determine quality.

Closer to our work, is the early work by Golbeck investigating trust networks in the Semantic Web [9]. This work introduced the notion of using network analysis type algorithms for determining trust or quality. However, this work focuses on trust from the point of view of social networks, not on networks in general. In some more recent work [11], network analysis has been used to study the robustness of the Web of Data. Our work differs in that it takes a wider view of quality beyond just robustness. The closest work is most likely the work by Bonatti *et al.*, which uses a variety of techniques for determining trust to perform robust reasoning [16]. In particular, they use a PageRank style algorithm to rank the quality of various sources while performing reasoning. Their work focuses on using these inputs for reasoning whereas LINK-QA specifically focuses on providing a quality analysis tool. Additionally, we provide for multiple measures for quality. Indeed, we see our work as complementary as it could provide input into the reasoning process.

6.2 Network Analysis on the Web of Data

There are only a few studies so far about network analysis on the Web of Data, most of the significant existing work is focused on semantic schemas, paying a particular attention to either the schema relations [21] or the documents instantiating them [7]. Both studies show, on various datasets, that schemas tend to follow power-law distributions. Network analysis has also been used to rank results when searching for datasets on the Web of Data [22]. Our work applies these techniques to quality of the data published.

6.3 Automated Creation of Links

There is a large body of literature over the creation of links between data sets on the Web of Data. As a cornerstone of semantic interoperability, ontologies have attracted most of the attention over the last decade. Several ontologies mapping/integration/merging techniques, tools and platforms allows for the connection of different datasets on the schema level [6]. The Silk Link discovery framework [23] offers a more versatile approach allowing configurable decisions on semantic relationships between two entities. More recently, the LIMES [17] framework offers an efficient implementation of similar functionality. Driven by those approaches, there has been increasing interest in new ways to measure the quality of automated links. For example, Niu et al. propose confidence and stability as metrics for measuring link creation based on notions from the information retrieval literature [18].

Overall, our work sits at the convergence of the need for the quality assessment of the links automatically created and the use of network measures to perform that assessment.

7 Conclusion

In this paper, we described LINK-QA, an extensible framework for performing quality assessment on the Web of Data. We described five metrics that might be useful to determine quality of Linked Data. These metrics were analysed using a set of known good and bad quality links created using the mapping tool Silk. The metrics were shown to be partially effective at detecting such links. From these results, we conclude that more tailored network measures need to be developed or that such a network based approach may need a bigger sample than the one we introduced. We are currently looking at finding more semantics-based measures, such as the sameAs chains. We are also looking at the interplay of different measures and the combined interpretation of their results.

Acknowledgements. This work was supported by the European Union's 7th Framework Programme projects LOD2 (GA no. 257943) and LATC (GA no. 256975). The authors would like to thank Peter Mika for his input.

References

1. Adamic, L.A.: The Small World Web. In: Abiteboul, S., Vercoustre, A.-M. (eds.) ECDL 1999. LNCS, vol. 1696, pp. 443–452. Springer, Heidelberg (1999)
2. Artz, D., Gil, Y.: A survey of trust in computer science and the Semantic Web. *J. Web Sem.* 5(2), 58–71 (2007)
3. Barabási, A.L.: *Linked* (Perseus, Cambridge, Massachusetts) (2002)
4. Bizer, C., Cyganiak, R.: Quality-driven information filtering using the WIQA policy framework. *Journal of Web Semantics* 7(1), 1–10 (2009)
5. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. In: *International World Wide Web Conference* (2005)

6. Choi, N., Song, I.Y., Han, H.: A survey on ontology mapping. *ACM SIGMOD Record* 35(3), 34–41 (2006)
7. Ding, L., Finin, T.: Characterizing the Semantic Web on the Web. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) *ISWC 2006. LNCS*, vol. 4273, pp. 242–257. Springer, Heidelberg (2006)
8. Golbeck, J.: Trust on the world wide web: a survey. *Foundations and Trends in Web Science* 1(2), 131–197 (2006)
9. Golbeck, J., Parsia, B., Hendler, J.: Trust Networks on the Semantic Web. In: Klusch, M., Omicini, A., Ossowski, S., Laamanen, H. (eds.) *CIA 2003. LNCS (LNAI)*, vol. 2782, pp. 238–249. Springer, Heidelberg (2003)
10. Guéret, C., Groth, P., van Harmelen, F., Schlobach, S.: Finding the Achilles Heel of the Web of Data: Using Network Analysis for Link-Recommendation. 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. 289–304. Springer, Heidelberg (2010)
11. Guéret, C., Wang, S., Schlobach, S.: The web of data is a complex system - first insight into its multi-scale network properties. In: *Proc. of the European Conference on Complex Systems*, pp. 1–12 (2010)
12. Guéret, C., Wang, S., Groth, P., Schlobach, S.: Multi-scale analysis of the web of data: A challenge to the complex system’s community. *Advances in Complex Systems* 14(04), 587 (2011)
13. Harth, A., Kinsella, S., Decker, S.: Using Naming Authority to Rank Data and Ontologies for Web Search. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) *ISWC 2009. LNCS*, vol. 5823, pp. 277–292. Springer, Heidelberg (2009)
14. Hartig, O.: Querying Trust in RDF Data with tSPARQL. 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. 5–20. Springer, Heidelberg (2009)
15. Hogan, A., Harth, A., Passant, A., Decker, S., Polleres, A.: Weaving the Pedantic Web. In: *Linked Data on the Web Workshop (LDOW 2010) at WWW 2010* (2010)
16. Hogan, A., Bonatti, P., Polleres, A., Sauro, L.: Robust and scalable linked data reasoning incorporating provenance and trust annotations. *Journal of Web Semantics* (2011) (to appear) (accepted)
17. Ngonga Ngomo, A.-C., Auer, S.: Limes - a time-efficient approach for large-scale link discovery on the web of data. In: *Proc. of IJCAI* (2011)
18. Niu, X., Wang, H., Wu, G., Qi, G., Yu, Y.: Evaluating the Stability and Credibility of Ontology Matching Methods. 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. 275–289. Springer, Heidelberg (2011)
19. Olaf, H., Zhao, J.: Using Web Data Provenance for Quality Assessment. In: *Proc. of the 1st Int. Workshop on the Role of Semantic Web in Provenance Management (SWPM) at ISWC, Washington, USA* (2009)
20. Sabater, J., Sierra, C.: Review on Computational Trust and Reputation Models. *Artificial Intelligence Review* 24(1), 33 (2005)
21. Theoharis, Y., Tzitzikas, Y., Kotzinos, D., Christophides, V.: On graph features of semantic web schemas. *IEEE Transactions on Knowledge and Data Engineering* 20, 692–702 (2007)

22. Toupikov, N., Umbrich, J., Delbru, R., Hausenblas, M., Tummarello, G.: DING! Dataset Ranking using Formal Descriptions. In: WWW 2009 Workshop: Linked Data on the Web (LDOW 2009), Madrid, Spain (2009)
23. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Silk: A link discovery framework for the web of data. In: 2nd Linked Data on the Web Workshop LDOW 2009, pp. 1–6. CEUR-WS (2009)
24. Zaihrayeu, I., da Silva, P.P., McGuinness, D.L.: IWTrust: Improving User Trust in Answers from the Web. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) *iTrust 2005*. LNCS, vol. 3477, pp. 384–392. Springer, Heidelberg (2005)

A Novel Concept-Based Search for the Web of Data Using UMBEL and a Fuzzy Retrieval Model

Melike Sah and Vincent Wade

Knowledge and Data Engineering Group, Trinity College Dublin, Dublin, Ireland
{Melike.Sah,Vincent.Wade}@scss.tcd.ie

Abstract. As the size of Linked Open Data (LOD) increases, the search and access to the relevant LOD resources becomes more challenging. To overcome search difficulties, we propose a novel concept-based search mechanism for the Web of Data (WoD) based on UMBEL concept hierarchy and fuzzy-based retrieval model. The proposed search mechanism groups LOD resources with the same concepts to form categories, which is called *concept lenses*, for more efficient access to the WoD. To achieve concept-based search, we use UMBEL concept hierarchy for representing context of LOD resources. A semantic indexing model is applied for efficient representation of UMBEL concept descriptions and a novel fuzzy-based categorization algorithm is introduced for classification of LOD resources to UMBEL concepts. The proposed fuzzy-based model was evaluated on a particular benchmark (~10,000 mappings). The evaluation results show that we can achieve highly acceptable categorization accuracy and perform better than the vector space model.

Keywords: Categorization, concept-based search, data mining, semantic indexing, fuzzy retrieval model, linked open data, UMBEL concept hierarchy.

1 Introduction

A key research focus in Web technology community is Linked Data. The term Linked Data describes best practices for creating typed links between data from different sources using a set of Linked Data principles. This ensures that published data becomes part of a single global data space, which is known as “Web of Data” (WoD) or “Linked Open Data” (LOD). Since the data is structured and relationships to other data resources are explicitly explained, LOD allows discovery of new knowledge by traversing links. However, as the number of datasets and data on the LOD is increasing, current LOD search engines are becoming more important to find relevant data for further exploration. This is analogous to the problem of the original Web [1]. However, current LOD search mechanisms are more focused on providing automated information access to services and simple search result lists for users [2, 3]. For example, they present search results in decreasing relevance order based on some criterion (i.e. relevance to class names). However, result list based presentations of retrieved links/resources do not provide efficient means to access LOD resources since URIs or titles of LOD resources are not very informative. More efficient access and discovery mechanisms on the WoD are crucial for finding starting points for

browsing and exploring potential data/datasets by Web developers and data engineers. Currently, there are few approaches, which investigate this problem [1, 4, 11].

Our objective is to improve current search mechanisms on the WoD with a novel concept-based search method. The dictionary definition of *concept* is “a general notion or idea”. Concept-based search systems provide search results based on the meaning, general notion of information objects so that search results can be presented in more meaningful and coherent ways. Key challenges in supporting concept-based search are: (1) the availability of a broad conceptual structure, which comprises good concept descriptions, (2) extraction of high-quality terms from LOD resources for the representation of resource context and categorization under the conceptual structure, and (3) a robust categorization algorithm. In this paper, we focus on these issues in order to introduce a novel concept-based search mechanism for the WoD.

1.1 Our Approach and Contributions

We introduce a novel concept-based search mechanism for the WoD based on the UMBEL concept hierarchy (<http://umbel.org/>), a fuzzy-based retrieval model and a categorical result list based presentation. The proposed search mechanism groups LOD resources with the same concepts to form categories, which we call *concept lenses*. In this way, search results are presented using categories and concept lenses, which can support more efficient access to the WoD. Such categorization enables the generation of much more human intuitive presentations, aggregations and concept-based browsing of retrieved links aligned to the users’ intent or interests. It can offer a considerable improvement over a single ranked list of results. Such presentations of links can allow more effective personalized browsing and higher user satisfaction [9].

There are three unique contributions of our approach: (1) For the first time, UMBEL is used for concept-based Information Retrieval (IR). UMBEL provides a rich concept vocabulary that is linked to DBpedia and designed to enable reasoning and browsing. (2) A second contribution is in novel semantic indexing and fuzzy retrieval model, which provides efficient categorization of search results in UMBEL concepts. This approach extends the traditional vector space model, which uses term frequency (*tf*) and inverse document frequency (*idf*) ($tf \times idf$) approach in indexing and retrieval to enable fuzzy relevancy score calculation according to relevancy of a term to semantic elements (structure) of concept(s). This significantly extends traditional $tf \times idf$ calculations. (3) A minor contribution is the realization of a concept-based search approach to WoD exploration. Concept-based search has only traditionally occurred in Web IR rather than in the realm of linked data.

The remainder of the paper is organized as follows: Section 2 discusses the related work. Section 3 discusses conceptual vocabularies for concept-based IR and explains why UMBEL was chosen rather than other well-known conceptual structures. Section 4 introduces the proposed concept-based search. In particular, term extraction from LOD resources, a novel semantic indexing and a novel fuzzy retrieval model is introduced for the categorization of LOD resources in UMBEL concepts. Section 5 presents evaluations prior to conclusions. Specifically, the proposed fuzzy retrieval model was evaluated on a particular benchmark from DBpedia to UMBEL mappings (~10,000), which achieved high categorization accuracy (~89%) and outperformed the vector space model (~33%), which is crucial for the correct formation of concept lenses. Moreover, time evaluations were performed to measure system performance.

2 Related Work

2.1 Concept-Based and Clustering-Based Information Retrieval (IR) Systems

Most of the current search engines utilize keyword-based search algorithms (i.e. full-text search) for information retrieval [2, 3]. Although this method is simple and fast, it often produces the problem of high recall and low precision, because it mainly finds documents that contain query keywords. Concept-based IR aims to improve retrieval effectiveness by searching information objects based on their meaning rather than on the presence of the keywords in the object. In concept based search, query context and context of information objects are represented with a reference concept hierarchy. Thus relevant information objects can be retrieved and results can be re-ranked (personalized) based on user's query context [5, 6]. Different than existing concept-based IR systems, which is based on results re-ranking [5, 6], our objective is to use concepts of LOD resources for categorical links presentation (concept lenses), which also has a key benefit of supporting concept-based browsing and discovery. On the other hand, since it is expensive and difficult to create broad conceptual structures, concept-based search approaches use existing conceptual structures, such as Yahoo Directory [7] and Open Directory Project (ODP) [5, 6]. With the increasing number of LOD ontologies and metadata, interests in using these taxonomies are decreasing.

Other relevant work is clustering search engines that group pages into topics (e.g. Carrot² – <http://search.carrot2.org/stable/search>) or hierarchical topics (e.g. Clusty – <http://search.yippy.com/>) for efficient information access/discovery. In these methods, challenge is the creation of useful topics that is achieved by clustering the retrieved pages using complex clustering algorithms [10]. Whereas in our work, a taxonomy is used for topic labels (which solve vocabulary problem) and our focus is categorization of individual resources to the known concepts (i.e. reverse of clustering techniques).

2.2 Search Mechanisms on the WoD

Current WoD search engines and mechanisms, such as Sindice [2] and Watson [3], utilize full-text retrieval, where they present a list of search results in decreasing relevance. However, users cannot understand “what the resource is about” without opening and investigating the LOD resource itself, since the resource title or example triples about the resource are not informative enough. More efficient search mechanisms on the WoD are crucial for finding starting points and exploration of potential data/datasets by Web developers and data engineers. Sig.ma attempts to solve this problem by combining the use of Semantic Web querying, rules, machine learning and user interaction [1]. The user can query the WoD and Sig.ma presents rich aggregated mashup information about a particular entity. Our approach is however focused on a novel concept-based presentation of search results using categories, which is different than mashup-based presentation of a particular resource.

Another related work is faceted search/browsing systems [4, 11], which is also known as exploratory search systems [14]. A key important difference between our “concept lenses” and faceted search is that we generate concept lenses based on a reference concept hierarchy rather than particular datatype or object properties of LOD resources. Thus our approach can be applied to heterogeneous LOD resources

that use different schemas. In contrast, faceted search systems are generally bound to specific schema properties and it can be difficult to generate useful facets for large and heterogeneous data of the WoD [12]. In addition, scalability and system performance is another issue for faceted systems over LOD [13].

3 Conceptual Structures and Vocabularies for Concept-Based IR

In concept-based IR, existence of a conceptual structure for representing context of an information object and query is crucial. The conceptual structures can range from simple thesaurus, dictionaries to more complex semantically rich ontologies. Yahoo Directory (<http://dir.yahoo.com/>), ODP (<http://www.dmoz.org/>), Proton (<http://proton.semanticweb.org/>), SUMO (<http://www.ontologyportal.org/>) and Sensus (<http://www.isi.edu/natural-language/projects/ONTOLOGIES.html>) are examples that have been utilized for concept-based search. SUMO and Proton have relatively sparse subject concepts and their penetration into general Web is quite limited. Sensus is a concept ontology derived from WordNet. However, Sensus does not include much semantic information, which can be very useful for concept-based IR. Yahoo and ODP are by far the most commonly used taxonomies for concept-based IR [5-8]. However, with the increasing number of LOD ontologies and metadata, usage of Yahoo and ODP has significantly decreased. Therefore, we looked for candidates in LOD ontologies such as DBpedia (<http://dbpedia.org/>), Yago (<http://www.mpi-inf.mpg.de/yago-naga/yago/>), OpenCyc (<http://www.opencyc.org/>) and UMBEL.

DBpedia and Yago's named entity coverage is good but their content does not have a consistent backbone structure¹, which makes it difficult to use and reason. OpenCyc is another upper ontology generated manually over the last twenty years. It captures common sense knowledge and a broad number of concepts. In addition, OpenCyc is purposefully created to support inferencing such as it uses WordNet for concept disambiguation and it captures subject relationships between concepts to enable reasoning. However, OpenCyc's top ontology concepts are obscure and contain many domain specific concepts that are developed for project purposes.

UMBEL is a cleaner and simpler sub-set of OpenCyc with the specific aim of promoting interoperability with external linked datasets. UMBEL provides a coherent framework of broad subjects and topics (around 28,000 concepts) with useful relationships and properties drawn from OpenCyc (i.e. broader, narrower, external, equivalent classes and preferred, alternative, hidden labels). In addition, UMBEL concepts are organized into 32 super type classes, which make it easier to reason, search and browse. Moreover, UMBEL is connected to/from OpenCyc and DBpedia (which is also linked to Yago through DBpedia).

On the other hand, recently Google, Yahoo and Bing announced schema.org, which is a markup vocabulary for annotating Web pages, so that search engines can improve presentation of search results. The vocabulary contains broad concepts, which can be useful if they publish the training data in future.

¹ DBpedia and Yago provide rich structures for linking instance data. However they do not have a consistent framework of concepts (topics) for representing those instances.

Based on current data, UMBEL’s broad concept coverage, rich representation of concept descriptions and powerful reasoning capabilities stand out among other LOD conceptual ontologies. Thus, we propose to use UMBEL for concept-based search.

4 Proposed Concept-Based Search on the Web of Data

The proposed concept-based search mechanism is fully implemented² and its system architecture is shown in Figure 1. Users can provide keyword or URI based queries to the system. Using these input queries, our system search the WoD by utilizing Sindice search API [2] and initial search results from the Sindice search are presented to users with no categorization. Then, for each search result (LOD URI), parallel requests are sent to the server for categorization of LOD resources under UMBEL concepts. First, for each LOD resource, its RDF description is cached to a Jena model using the Sindice Cache (i.e. <http://any23.org/>) at the server. Using the RDF description, terms from different semantic parts of the LOD resource are mined and the extracted terms are weighted for matching to UMBEL concept representations. In particular, UMBEL concepts are represented by a semantic indexing model (see Section 4.2). The obtained terms from the LOD resource are matched to the inverted concept index by a fuzzy-based retrieval algorithm (see Section 4.3) and categorized LOD resources are sent back to the client. Since dynamic categorization can be time consuming, search results are shown incrementally by using Asynchronous JavaScript (AJAX) to enhance user experiences with the system. Finally, LOD resources with the same concepts (i.e. resources with same categories, e.g. Organization) are grouped together to form concept lenses for coherent presentation of search results. LOD resources with no categorization are presented without a category in the result lists.

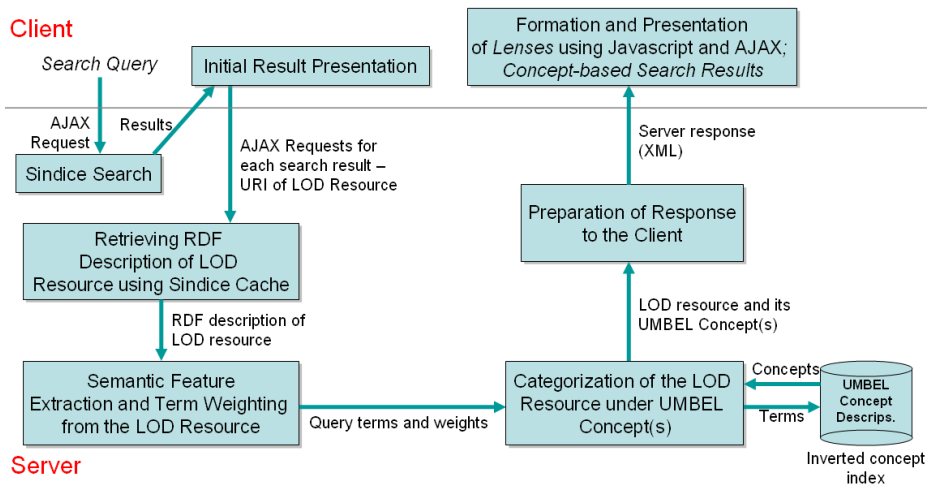


Fig. 1. System Architecture

² It will be made public. A video demo is available at http://www.scss.tcd.ie/melike.sah/concept_lenses.swf

Given that indexing and caching of WoD is very expensive, our approach is based on existing 3rd party services. In particular, we use Sindice search for querying the WoD and Sindice Cache for retrieving RDF descriptions of LOD resources [2]. Lucene IR framework is utilized for indexing of concepts and at the implementation of the fuzzy retrieval model. The server side is implemented with Java Servlets and uses Jena for processing RDF. The client side is written using Javascript and AJAX.

In Figure 2, a screen shot of the concept-based search interface is presented. The user interface presents the list of categories (concepts) at the left of the screen for quick navigation access to concept lenses (LOD resources grouped based on context).

4.1 Recognizing Context of Linked Open Data Resources

In order to generate concept-based search results, first the retrieved LOD resources from the Sindice search need to be categorized under UMBEL concepts. To achieve this, the concepts of LOD resources should be understood, where lexical information about LOD resources can be used to mine such knowledge. One option is to extract all lexical information from the URI, labels, properties and property values of the LOD resources that are retrieved by Sindice search. However, in such a process, many misleading words may also be extracted. For example, a LOD resource about a TV broadcasting organization may include information about broadcasting network but it may also include other information about programs, coverage, etc., which may lead to an incorrect categorization. Thus, the challenge is to identify important features of LOD resources for correct categorization.

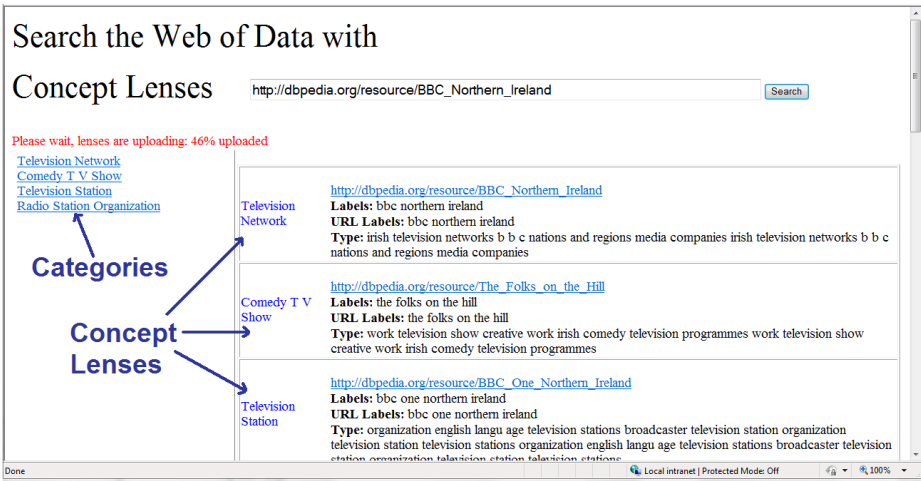


Fig. 2. The user interface of the concept-based search mechanism – categories (concepts) at the left of the screen and concept lenses in the main panel of the screen

Term Extraction and Enrichment. We chose the following common features of LOD resources that may be used to represent context of the resource – *URI (u)*, *label (l)*, *type (t)*, *subject (s)* and *property names (p)*. We chose these features for the

following reasons: the *URI* of a resource may contain keywords relevant to the context of the resource. Titles (dc:title), names (foaf:name) and labels (rdfs:label), so called *label* features usually include informative information about the resource. *property names* typically provide further knowledge about “what type of the resource is”. For example, birth place, father, mother, spouse are properties associated with persons. However, some property names are generic (i.e. rdfs:label, rdf:type, foaf:name, etc.) and do not provide information about the context. To overcome this, we compiled a list of generic property names and if a property name matches any of these, it is not accepted. On the other hand, *type* (rdf:type and dc:type) and *subject* (dc:subject) provides the most discriminative features about the context of a LOD resource. For instance, *type* and *subject* values can provide useful knowledge about concepts (general notion or idea) of the resource for correct categorization. For instance, for label “ocean” the context is not clear. But if we know *type* is album, then we can understand that “ocean” is a label of an “album”.

From each LOD resource, keywords are extracted from the features as explained above. Then, qualifiers and propositions are removed from the extracted keywords, to enhance categorization accuracy. For instance, for the context “hockey games in Canada”, the main concept is “hockey games” and not the “country” Canada. Thus, we remove qualifiers/ propositions and the words after them for better term extraction. Qualifier removal is based on keyword matching of *from*, *in*, *of* and *has*. To ensure correct matching, there must be a space before/after the qualifiers, e.g. words in italic are removed after the qualifiers: people *from Alaska*, reservoirs *in Idaho*, mountains *of Tibet*, chair *has four legs*. This has the effect of generalizing the concepts, which is perfectly reasonable for our purpose of categorizing and browsing based on higher level of concepts.

On the other hand, after initial experiments, we observed that many LOD resources do not have information about *label*, *type* and *subject*. To improve lexical data mining, we also apply a semantic enrichment technique, where more lexical data is gathered from the linked data graph of the resource by traversing owl:sameAs and dbpedia:WikiPageRedirect links. If a resource has such links, first we obtain RDF description of these resources and apply the feature extraction techniques explained above. Finally, from the obtained and enriched terms, stop words are removed and stemming is applied, where we obtain the final terms, which we call *LOD terms*.

LOD Term Weights Based on Features. Since different *LOD terms* have comparative importance on the context of the LOD resource, terms are weighted. For example, *type* and *subject* features provide more discriminative terms. Therefore terms which appear in these features should be weighted higher. To achieve this, we divided LOD resource features into two groups: Important features (*I*) and Other features (*O*). For important features terms from *type* and *subject* features are combined to form a vector. For other features terms from *URI*, *label* and *property name* are combined to form a vector. We use the normalized term frequency (*tf*) of these features for term weighting as given below,

$$\text{if } t \in I, w(t) = 0.5 + 0.5 \times \left(\frac{tf(t)_I}{\max(tf_I)} \right), \quad \text{if } t \in O, w(t) = \frac{tf(t)_O}{\max(tf_O)} \quad (1)$$

where $w(t)$ is weight of term t , and $tf(t)_I$ and $tf(t)_O$ is term frequency of term t in important and other features respectively. $\max(tf(t)_I)$ and $\max(tf(t)_O)$ represent maximum term frequency in those features. For terms that appear in important features (I), a minimum weight threshold of 0.5 is used to encourage syntactic matches to these terms within UMBEL concept descriptions for categorization. On the other hand, inverse document frequency (idf) can also be used together with term frequency. However, idf calculation for each LOD term is expensive. For dynamic idf calculations, dynamic search on the LOD is required for each term, which is computationally expensive. For offline calculations of idf , we need to continuously index the LOD, which is a resource-intensive task. Thus, we did not use idf .

4.2 Representation of Umbel Concept Descriptions

In UMBEL version 1.0, there are 28,000 UMBEL concepts, which are categorized under 32 top-level *Supertype* classes (i.e. Events, Places, etc.) and classified into a taxonomy using super and sub-concepts. Each UMBEL concept description contains preferred label and alternative labels. Alternative labels usually include synonyms, quasi-synonyms, lexical variations, plural, verb derivations and semantic related words. In summary, UMBEL provides highly structured descriptions of a broad number of concepts, which can be used to represent the context of LOD.

Semantic Indexing Model. The formal representation of concept descriptions plays a crucial role in the effectiveness of the IR system. In general, concepts' representations are based on extraction of keywords and the usage of a weighting scheme in a vector space model ($tf \times idf$). This provides a simple but robust statistical model for quick retrieval [8]. For indexing of UMBEL concept descriptions, $tf \times idf$ weighting scheme is used similar to other concept-based IR models [5-7]. Typically these IR models utilize vector space representations of categories (concepts), where the terms inside the category description and the terms inside sub-categories are indexed and retrieved using $tf \times idf$ scheme. However, such formal representations are extremely simple and do not discriminate the terms that are semantically more important to the concept based on the semantic structure of concept hierarchy.

In our opinion, structured concept descriptions of UMBEL can be indexed more efficiently by exploiting the semantic structure of the concept descriptions. For instance, where the term appears in a structured concept description (i.e. in a URI label, preferred labels, alternative labels, sub-concepts labels or super-concepts labels), should have a certain impact on the associated weight of the term to the concept. Therefore, to produce more effective concept representations, we propose a semantic indexing model based on the different semantic parts of the concept.

In our approach, we divided concept descriptions into different parts: *concept URI labels (uri)*, *concept labels (cl)*, *sub-concept labels (subl)*, *super concept labels (supl)* and *all labels (al)*. A *uri* contains terms that appear in the URI of the concept, where terms that appear in a *uri* can be particularly important to the concept. *cl* contain terms that appear in the preferred and alternative labels of the concept. Hence most lexical variations of the concept description are captured by *cl*. In concept-based IR, typically sub-concept labels are also accepted as a part of the concept [5-7]. For instance, for the concept "sports", sub-concepts baseball, basketball, football, etc.

may provide further relevant lexical terms about “sports”. Instead of accepting sub-concepts as a part of the concept, we separately index sub-concepts labels as *subl*. The *subl* include terms that occur in all inferred sub-concepts’ URIs, preferred and alternative labels. In addition, we observed that many LOD resources contain links to super-concepts. For example, a resource about a writer also contains information that writer is a person. Thus, we index super-concept labels for more robust descriptors, where *supl* contain terms that occur in all inferred super-concepts’ URIs, preferred and alternative labels. Finally, *al* contain all the terms that appear in all parts.

UMBEL is formatted in RDF N-triple format and we load the triples into a triple store (Jena persistent storage using Mysql DB) to extract the terms from UMBEL concepts. Each concept is divided into semantic parts of *uri*, *cl*, *subl*, *supl* and *al* using SPARQL queries, where concept descriptions are extracted from each semantic part. From the concept descriptions, stop words are removed, as they have no semantic importance to the description, and words are stemmed into their roots using the Porter stemmer. The resultant words are accepted as *concept terms*. Finally, the extracted concept terms from the semantic parts are indexed. To do this, we consider each concept as a unique document and each semantic part is separately indexed as a term vector under the document (concept) using Lucene IR framework. In addition, the maximum normalized term frequency and inverse document frequency term value of each semantic part is calculated (which is subsequently used by the fuzzy retrieval model) and indexed together with the concept for quick retrieval. The inverted concept index is used for categorization of LOD resources.

It should be also noted that concept descriptions can be enhanced with lexical variations using WordNet. However, typically UMBEL descriptions include such word variations in alternative labels. This is the advantage of UMBEL being built upon on OpenCyc since OpenCyc contains rich lexical concept descriptions using WordNet. For the UMBEL concept <<http://umbel.org/umbel/rc/Automobile>> for instance, preferred label is *car* and alternative labels are *auto*, *automobile*, *automobiles*, *autos*, *cars*, *motorcar* and *motorcars*. This demonstrates a rich set of apparent lexical variations. Since these rich lexical descriptions are available in UMBEL, we did not use other lexical enhancement techniques because accuracy of the automated enhancements may affect categorization performance significantly.

4.3 Categorization of LOD Resources Using a Novel Fuzzy Retrieval Model

In this step, we match the extracted *LOD terms* to UMBEL concept descriptions. In traditional IR, $tf \times idf$ is used to retrieve relevant concepts, i.e. each term in a concept has an associated value of importance (i.e. weight) to that concept and important terms are those that are frequently occur inside a text but infrequent in the whole collection ($tf \times idf$) [8]. Since we represent each UMBEL concept as a combination of different semantic parts (rather than one document), we need to calculate term relevancy to each semantic part. Then individual part relevancies can be used for a final relevancy score calculation. For this purpose, we propose a fuzzy retrieval model, where relevancy of a term, t , on concept, c , is calculated by a fuzzy function, $\mu(t,c) \in [0,1]$, using semantic parts of the concept c and an extended $tf \times idf$ model.

Fuzzy Retrieval Model. First, UMBEL concept candidates are retrieved by searching *LOD terms* in all labels (*al*) of concepts. Then, for each LOD term, *t*, a relevancy score to every found UMBEL concept, *c*, is calculated by using a fuzzy function, $\mu(t,c) \in [0,1]$, on *uri*, *cl*, *subl* and *supl* semantic parts (since different parts have relative importance on the context of the concept *c*). Thus, $\mu(t,c)$ shows the degree of membership of the term *t* to all semantic parts of the concept *c*; where high values of $\mu(t,c)$ show that *t* is a good descriptor for the concept *c* and $\mu(t,c) = 0$, means that the term *t* is not relevant for *c*. For membership degree calculation of $\mu(t,c)$, first membership degree of the term, *t*, to each part, *p*, should to be computed:

Definition 1: The membership degree of the term, *t*, to each part, $p = [cl, subl, supl, uri]$, is a fuzzy function, $\mu(t,c,p) \in [0,1]$, which is based on *tf* \times *idf* model. First, we calculate normalized term frequency (*ntf*) of *t* in *cl*, *subl* and *supl* of the concept *c*,

$$ntf(t,c,cl) = 0.5 + 0.5 \times \left(\frac{tf(t,c,cl)}{\max(tf(c,cl))} \right), \forall subl, supl \in p, ntf(t,c,p) = \frac{tf(t,c,p)}{\max(tf(c,p))} \quad (2)$$

where *tf*(*t,c,p*) represents term frequency of the term *t* in the part *p* of the concept *c* and $\max(tf(c,p))$ represents maximum term frequency in the part *p* of the concept *c*. We calculate local normalized term frequencies for each semantic part, rather than calculating normalized term frequency using all terms of the concept in all semantic parts. In this way, term importance for a particular semantic part is obtained and frequent terms have higher value. For *cl* a minimum threshold value of 0.5 is set, since *cl* contains preferred/alternative terms of the concept, which is important for the context of the concept *c*. Then, for each part, *p*, we calculate the *idf* value of *t* in *p*,

$$\forall cl, subl, supl \in p, idf(t,c,p) = \log \left(\frac{C}{(n:t \in p) + 1} \right) \quad (3)$$

here, $n:t \in p$ is the number of semantic parts that contain the term *t* in *p* (i.e. $n:t \in cl$) and *C* is the total number of concepts in the collection. Again *idf* of a term in a particular semantic part is calculated instead of *idf* of a term in the whole corpus. In this way, rare terms that occur in a particular semantic part are assigned with higher values, which mean that rare terms are more important for the semantic part *p*. Next, *tf* \times *idf* value of the term *t* in the semantic part *p* is computed,

$$\forall cl, subl, supl \in p, tf \times idf(t,c,p) = ntf(t,c,p) \times idf(t,c,p) \quad (4)$$

Finally, the membership degree of the term *t* to each part *p* is a fuzzy value,

$$\forall cl, subl, supl \in p, \mu(t,c,p) = \frac{tf \times idf(t,c,p)}{\max(tf \times idf(c,p))} \quad (5)$$

where $\mu(t,c,p) \in [0,1]$ equals to normalized *tf* \times *idf* value of the term *t* in the part *p*. In this way, a fuzzy relevancy score is generated, where the term that has the maximum *tf* \times *idf* value in the part *p*, $\mu(t,c,p) = 1$ and $\mu(t,c,p)$ reduces as the term importance decreases. As we discussed earlier, the maximum *tf* \times *idf* value for each

semantic part is calculated and indexed during the semantic indexing for better algorithm performance. Last, $\mu(t, c, uri) \in [0,1]$ equals to normalized term frequency,

$$\mu(t, c, uri) = 0.5 + 0.5 \times \left(\frac{tf(t, c, uri)}{\sum_{i=1}^n tf(t_i, c, uri)} \right) \quad (6)$$

here, term frequency of t in uri is divided by total number of terms in the uri . A minimum threshold value of 0.5 is set, since uri terms are important. If uri contains one term, $\mu(t, c, uri) = 1$, means the term is important for uri . The term importance decreases as the number of terms in the uri increases.

Definition 2: Relevancy of the term t to the concept c , is calculated by, $\mu(t, c)$, where membership degrees of the term t to the parts uri , cl , $subl$ and $supl$ are combined,

$$\mu(t, c) = \frac{w_{uri} \times \mu(t, c, uri) + w_{cl} \times \mu(t, c, cl) + w_{subl} \times \mu(t, c, subl) + w_{supl} \times \mu(t, c, supl)}{w_{uri} + w_{cl} + w_{subl} + w_{supl}} \quad (7)$$

where, $\mu(t, c) \in [0,1]$ and, w_{uri} , w_{cl} , w_{subl} and w_{supl} are constant coefficients that aid to discriminate features obtained from different parts. For example, the terms obtained from the uri and cl can be weighted higher than $subl$ and $supl$. The parameter values were experimentally determined as we discuss later in the evaluations section.

Definition 3: Finally, relevancy of all *LOD terms*, $T = \{t_1, \dots, t_m\}$, to the concept c is,

$$\mu(T, c) = \frac{\sum_{i=1}^m (\mu(t_i, c) \times w(t_i))}{\sum_{i=1}^m w(t_i)} \quad (8)$$

where, $\mu(T, c) \in [0,1]$ and $w(t_i)$ is term importance of LOD term t_i , which is calculated by Equation 1. Using term weights, $w(t_i)$, term matches to the important LOD terms are encouraged (i.e. terms with higher weights). In addition, concepts that have a good coverage of *LOD terms* especially in important semantic parts (which is determined by coefficients in Equation 7), will have a higher relevancy score.

Definition 4: Finally, the concept with the maximum $\mu(T, c)$ is selected as the categorization of the LOD resource. In cases where there are two or more concepts that have the maximum value, categorization is decided based on an ontological relationships driven voting algorithm. The algorithm is as follows: For each concept with the maximum $\mu(T, c)$; (1) we find the number of sub-concepts (n), (2) sub-concepts (sc) count, k , that have non-zero relevancy value to *LOD terms*, $\mu(T, sc) > 0$, (3) total membership degree, ψ , of all sub-concepts that $\mu(T, sc) > 0$. The voting score (v) of the concept is computed as, $v = \psi \times (k/n)$, which means we encourage the concept whose sub-concepts have higher scores as well as the concept with a greater number of sub-concept matches. After the voting, the concept with the maximum v is accepted as the categorization. If there is still more than one maximum, we accept all concepts as categorizations, since a LOD resource may belong to one or more concept, e.g. Kyoto is a city as well as a district.

5 Evaluations

This section discusses the evaluation setup and the experiments undertaken to test the performance of our approach. A particular benchmark was created to evaluate: (1) performance of different LOD features, (2) categorization accuracy of the fuzzy retrieval model against the vector space model, (3) efficiency of system performance.

5.1 Setup

The performance of the proposed concept-based search depends on the accuracy of the fuzzy-based categorization algorithm, because incorrect categorizations can degrade user experience with the search mechanism. Thus, categorization accuracy is crucially important and it needs to be evaluated. Fortunately, many DBpedia LOD resources (~900,000) have mappings to UMBEL concepts and DBpedia publishes these links in RDF N-Triple format (<http://dbpedia.org/Downloads>). Since our aim is to test performance of various features and different algorithms, the use of whole mappings is computationally expensive. Instead, we created a particular benchmark of ~10,000 mappings from the provided DBpedia links. The selection procedure for the benchmark was as follows: We randomly selected from different types of UMBEL concept mappings (e.g. `umbel:HockeyPlayer`, `umbel:Plant`, etc.) and those DBpedia resources that are cached by the Sindice Cache. This resulted with 10227 benchmark resources. Then, RDF descriptions of 10,227 benchmark resources and resources that are linked from those resources using `owl:sameAs` and `dbpedia:WikiPageRedirect` links (~2 per resource) were cached to a local disk for context extraction. In addition, during the feature extraction, links to UMBEL concepts were ignored.

In the experiments, the parameter values of $w_{uri} = 2$, $w_{cl} = 2$, $w_{subl} = 1$ and $w_{super} = 1$ which were experimentally determined to achieve the best results. We randomly selected 500 mappings from the benchmark and chose the values that gave the best precision/recall. With these parameter values, URI and concept labels have the highest impact and, sub-concept and super-concept labels contribute moderately.

5.2 Categorization Accuracy – Precision and Recall

By accepting the DBpedia to UMBEL mappings as ground truth, precision (P) and Recall (R) of the automatically generated categorizations are calculated. Precision equals to the number of correctly predicted (CPr) divided by total number of predictions (Pr), $P = CPr / Pr$. Recall equals to the number of unique correct predictions (UPr) per resource (since our algorithm may predict one or more categorizations for each resource) divided by total number of mappings ($T = 10227$), $R = UPr / T$. In evaluations, if the predicted categorization directly matches to any of UMBEL concept mappings or the predicted categorization is a super-concept of a UMBEL concept mapping, then the categorization is accepted as correct. Super-concept mappings are also accepted as correct since it is intuitively and logically true. For example, if x is a `umbel:Cyclist`, it is also true that x is a `umbel:Athlete` or if x is a `umbel:Bird`, it is also true that x is a `umbel:Animal`.

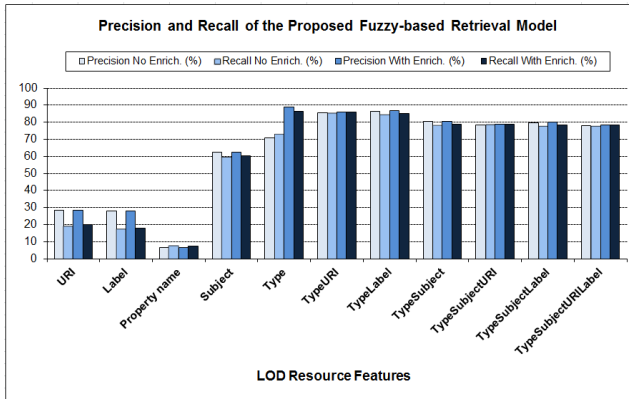


Fig. 3. Categorization accuracy of the proposed fuzzy retrieval model with respect to different LOD resource features and the semantic enrichment technique

Proposed Fuzzy Retrieval Model. Figure 3 shows precision and recall of the proposed fuzzy retrieval model: (1) with different LOD resource features and (2) with and without the semantic enrichment technique. The results show that among all LOD resource features, *type* feature alone gave the best precision of 70.98% and 88.74% without and with the enrichment respectively. This is because most resources contain *type* feature, which provides knowledge about the context of resources. *subject* feature performed a precision of ~62%, *uri* and *label* features alone did not perform well (~28%) and *property names* performed the worse accuracy. Among combinations of different LOD resource features, *type+uri* and *type+label* provided the best accuracy without the semantic enrichment with a precision of 85.55% and 86.38% respectively. Other combinations did not improve the overall accuracy despite more *LOD terms* being used in the categorization. Another interesting outcome is that the semantic enrichment technique did not have a significant impact on the categorization accuracy (~1% improvement) except the *type* feature. In the *type* feature, the enrichment technique improved the precision and recall ~18%. In addition, we noticed that in some cases all possible mappings from DBpedia to UMBEL are not included, e.g. a volcano mountain is mapped as `umbel:Mountain`, but not as `umbel:Volcano`. Besides, DBpedia uses more general mappings, for example, a science fiction writer is mapped as `umbel:Writer`, despite the existence of `umbel:ScienceFictionWriter`. This could be because of human error since manual mapping process³ is involved, which can be error-prone. Although these particular cases affected categorization accuracy, the proposed fuzzy retrieval model achieved high accuracy on the benchmark. Especially high performance is achieved by using the *type* feature and the *type+uri* and *type+label* features (with and without the enrichment). The results are promising because typically LOD resources contain data about type and labels of the resource, which can be used to provide high quality categorization.

³ <http://umbel.googlecode.com/svn/trunk/v100/External%20Ontologies/dbpediaOntology.n3>

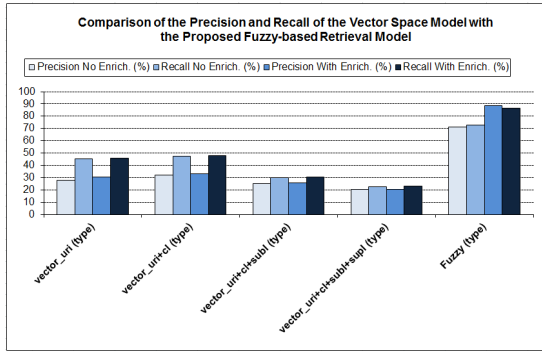


Fig. 4. Comparison of the vector space model with the proposed fuzzy retrieval model

Vector Space Model. Since the proposed fuzzy retrieval model extends $tf \times idf$ with a fuzzy relevancy score calculation using semantic structure of concepts, we compared the categorization accuracy against the $tf \times idf$ model. In vector space model, concept descriptions can be represented in a number of ways; using (1) only *uri*, (2) *uri+cl*, (3) *uri+cl+ subl*, and (4) *uri+cl+subl+supl*. On the concept representation alternatives, we applied $tf \times idf$ retrieval model on the benchmark. For fair comparison, the same clean-up steps are applied to the vector space model (i.e. stemming, stop word and qualifier removal) and the same voting algorithm is used if there is more than one maximum categorization. In contrast, concept weights to all *LOD terms* are calculated using the $tf \times idf$ scheme. In Figure 4, the best results are shown, which is achieved by the *type* feature. Results show that the vector space model did not perform well. The best precision and recall is obtained by using *uri+cl* with a precision and recall of 31.77% and 47.42 without the semantic enrichment and with a precision and recall of 33.09% and 47.75 with the semantic enrichment. When using all semantic parts, the precision of the vector space is decreased to 20.37% compared to 88.74% precision of the proposed fuzzy retrieval model.

Discussion of Results. Our fuzzy retrieval model performs outstandingly better than the vector space model for the following reason. $tf \times idf$ is a robust statistical model, which works well with good training data. Traditional concept-based IR systems [5,6,7] use the top 2-3 levels of a concept hierarchy (few hundred concepts) with hundreds of training documents. In contrast, we use the whole 28,000 UMBEL concepts. Moreover each concept contains few lexical information in different semantic parts of the concept, such as in URI, preferred/alternative labels and super/sub-concept(s) labels to describe that concept. $tf \times idf$ cannot discriminate terms only using combined terms and often few LOD terms are matched to many concepts (sometimes hundreds) with the same $tf \times idf$ scores. We propose a more intuitive approach, where our fuzzy retrieval model extends $tf \times idf$ with a fuzzy relevancy score calculation based on semantic structure of concepts, i.e. terms from the concept, sub-concept(s) and super-concept(s) have certain importance in retrieval. Besides, relevancy scores are combined according to their importance to the concept. Hence, this more intuitive approach performs astoundingly better than $tf \times idf$, which do not discriminate term importance based on semantic structure of a concept hierarchy.

5.3 Computational Efficiency of Dynamic Categorization

In addition to high accuracy, dynamic categorization performance is an important factor for the proposed concept-based search. To provide fast categorizations, each search result (resource) is processed in parallel using AJAX. In addition, to give an idea of dynamic (online) categorization times, we measured average processing times based on number of *LOD terms* per resource using a laptop with Windows 7 operating system, 4 GB RAM, Intel Core 2 Duo CPU (2.53 GHz) and 54Mbps Internet connection. Without the enrichment, average processing times vary between 1-1.5 secs for the proposed approach compared to 0.1-0.5 secs of the vector space model. With the enrichment, processing times increase for both model, because of dynamic caching from LOD graphs. We found that an average of twelve *LOD terms* are extracted from the benchmark resources, which means we can perform categorization within ~1 secs and ~1.5 secs with and without the enrichment respectively.

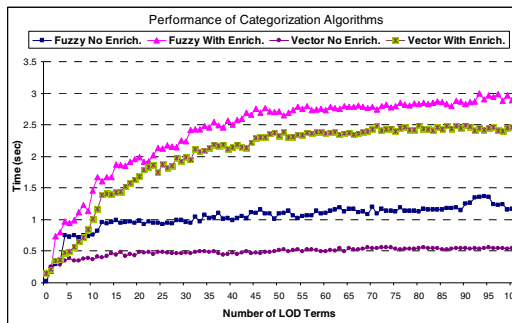


Fig. 5. Dynamic algorithm performance with respect to number of LOD terms

6 Conclusions and Future Work

We have presented a novel approach for concept-based search on the Web of Data. The proposed innovative search mechanism is based on UMBEL concept hierarchy, fuzzy-based retrieval model and categorical result list presentation. Our approach groups LOD resources with same concepts to generate concept lenses that can provide efficient access to the WoD and enables concept-based browsing. The concept-based search is achieved using UMBEL for representing context of LOD resources. Then, a semantic indexing model is applied for efficient representation of UMBEL concept descriptions. Finally a fuzzy-based retrieval algorithm is introduced for categorization of LOD resources to UMBEL concepts. Evaluations show that the proposed fuzzy-based model achieves highly acceptable results on a particular benchmark and outperforms the vector space model in categorization accuracy, which is crucial for correct formation of concept lenses.

The introduced semantic indexing and fuzzy retrieval model are not inherently dependent on UMBEL vocabulary and should be applicable to multiple vocabularies.

Moreover, in UMBEL, we are using all sub-concepts and super-concepts of a concept, but other vocabularies can be explored for relating concepts with different semantic relationships other than hierarchical structures. In future work, we will incorporate personalization into our concept-based search methodology in order to personalize results to the context and individual needs of the user, and perform user-based studies.

References

1. Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., Decker, S.: Sig.ma: live views on the Web of Data. *Journal of Web Semantics* 8(4), 355–364 (2010)
2. Delbru, R., Campinas, S., Tummarello, G.: Searching Web Data: an Entity Retrieval and High-Performance Indexing Model. *Journal of Web Semantics* 10, 33–58 (2012)
3. D’Aquin, M., Motta, E., Sabou, M., Angeletou, S., Gridinoc, L., Lopez, V., Guidi, D.: Toward a New Generation of Semantic Web Applications. *IEEE Intelligent Systems* (2008)
4. Heim, P., Ertl, T., Ziegler, J.: Facet Graphs: Complex Semantic Querying Made Easy. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) *ESWC 2010. LNCS*, vol. 6088, pp. 288–302. Springer, Heidelberg (2010)
5. Chirita, P.A., Nejd, W., Paiu, R., Kohlschütter, C.: Using ODP metadata to personalize search. In: *International ACM SIGIR Conference* (2005)
6. Sieg, A., Mobasher, B., Burke, R.: Web Search Personalization with Ontological User Profiles. In: *International Conference on Information and Knowledge Management* (2007)
7. Labrou, Y., Finin, T.: Yahoo! As An Ontology – Using Yahoo! Categories to Describe Documents. In: *International Conference on Information and Knowledge Management* (1999)
8. Salton, G., McGill, M.J.: *Introduction to Modern Information Retrieval* (1983)
9. Steichen, B., O’Connor, A., Wade, V.: Personalisation in the Wild – Providing Personalisation across Semantic, Social and Open-Web Resources. *ACM Hypertext* (2011)
10. Carpineto, C., Romano, G.: Optimal Meta Search Results Clustering. In: *SIGIR* (2010)
11. Erling, O.: Faceted Views over Large-Scale Linked Data. In: *Linked Data on the Web (LDOW) Workshop*, co-located with *International World Wide Web Conference* (2009)
12. Teevan, J., Dumais, S.T., Gutt, Z.: Challenges for Supporting Faceted Search in Large, Heterogeneous Corpora like the Web. In: *Workshop on HCIR* (2008)
13. Shangquan, Z., McGuinness, D.L.: Towards Faceted Browsing over Linked Data. In: *AAAI Spring Symposium: Linked Data Meets Artificial Intelligence* (2010)
14. White, R.W., Kules, B., Drucker, S.M., Schraefel, M.C.: Supporting Exploratory Search. *Introduction to Special Section of Communications of the ACM* 49(4), 36–39 (2006)

Unsupervised Learning of Link Discovery Configuration

Andriy Nikolov, Mathieu d’Aquin, and Enrico Motta

Knowledge Media Institute, The Open University, Milton Keynes, UK
{a.nikolov,m.daquin,e.motta}@open.ac.uk

Abstract. Discovering links between overlapping datasets on the Web is generally realised through the use of fuzzy similarity measures. Configuring such measures is often a non-trivial task that depends on the domain, ontological schemas, and formatting conventions in data. Existing solutions either rely on the user’s knowledge of the data and the domain or on the use of machine learning to discover these parameters based on training data. In this paper, we present a novel approach to tackle the issue of data linking which relies on the unsupervised discovery of the required similarity parameters. Instead of using labeled data, the method takes into account several desired properties which the distribution of output similarity values should satisfy. The method includes these features into a fitness criterion used in a genetic algorithm to establish similarity parameters that maximise the quality of the resulting linkset according to the considered properties. We show in experiments using benchmarks as well as real-world datasets that such an unsupervised method can reach the same levels of performance as manually engineered methods, and how the different parameters of the genetic algorithm and the fitness criterion affect the results for different datasets.

1 Introduction

Identity links between data instances described in different sources provide major added value of linked data. In order to facilitate data integration, newly published data sources are commonly linked to reference repositories: popular datasets which provide good coverage of their domains and are considered reliable. Such reference repositories (e.g., DBpedia or Geonames) serve as hubs: other repositories either link their individuals to them or directly reuse their URIs. However, establishing links between datasets still represents one of the most important challenges to achieve the vision of the Web of Data. Indeed, such a task is made difficult by the fact that different datasets do not share commonly accepted identifiers (such as ISBN codes), do not rely on the same schemas and ontologies (therefore using different properties to represent the same information) and often implement different formatting conventions for attributes.

Automatic data linking often relies on fuzzy similarity functions comparing relevant characteristics of objects in the considered datasets. More precisely, a data linking task can be specified as the evaluation of a *decision rule* establishing

whether two individuals should be considered equivalent, based on the value of a function aggregating the similarity comparisons of some properties of these individuals. In most systems, establishing the appropriate decision rule is left to the user, who needs to rely on his/her knowledge of the domain, of the data in both datasets, and on his/her intuition regarding the performance of various similarity functions in the considered linking situation. Other systems try to alleviate the issue of establishing the decision rule for linking by using machine learning techniques. They however require a substantial set of training data in the form of pre-established links within a subset of the considered datasets.

In this paper, we investigate the question: can a suitable decision rule for linking two datasets be learned without possessing labelled training data, based only on the characteristics of the datasets and on the distribution of similarity values amongst their instances? Our hypothesis is that in a scenario which involves establishing links to reference datasets, available information (e.g., knowledge that the datasets do not contain duplicates and have high degree of overlap) can provide sufficient evidence to learn a decision rule which would determine identity mappings between instances in two datasets with high accuracy. To learn such rules, we propose an approach based on a genetic algorithm, which evolves a set of initially random solutions to a problem according to a fitness criterion. Following research in the area of record linkage in databases, we devise an applicable fitness criterion which relies on the distribution of links and similarity values generated by applying a particular decision rule.

To test our assumptions, we apply this approach to the benchmark datasets from the OAEI 2010 and 2011 instance matching contests. We show that applying the learned decision rule for data linking achieves results at the level of the best state-of-the-art tools, without the need to configure linking parameters for each task. We also experiment with subsets of real-world linked datasets to demonstrate the robustness of the approach to different types of datasets in different domains and discuss the effects of some of the parameters of the genetic algorithm on its behaviour in data linking tasks. The remainder of this paper is structured as follows. In section 2, we provide an overview of the basic notions of the link discovery problem and relevant work in both Semantic Web and database research communities. Section 3 describes our algorithm in detail. Section 4 describes the experiments we performed in order to validate our approach. Section 5 concludes the paper and discusses directions for future work.

2 Problem Definition and Related Work

In this section, we specify the tasks of link discovery and of establishing the necessary decision rule, together with a brief description of the relevant existing work.

2.1 Link Discovery Problem

The problem of reconciliation was originally studied in the database community where it is known as record linkage or object identification [3]. With the

development of the linked data initiative, it gains importance in the Semantic Web community where it is studied under the name of link discovery [14]. The link discovery task takes as inputs two datasets \mathcal{D}_1 and \mathcal{D}_2 and tries to discover all pairs of individuals (I_{1i}, I_{2j}) belonging to these datasets such that they describe the same entity ω according to a chosen identity criterion. In the context of linked data, datasets \mathcal{D}_1 and \mathcal{D}_2 represent RDF graphs and their individuals are identified by URIs.

Existing techniques solving this task can be divided into two main categories: *individual matching* and *dataset matching*. We essentially focus on individual matching in this paper. *Dataset matching* techniques are built on top of individual matching ones: they take as input two datasets as a whole together with the initial set of mappings produced by individual matching and further refine them. These techniques take into account additional available information such as relations between individuals, axioms defined in the ontological schema, and mutual impact of different mappings. The individual matching task can be defined as follows.

Definition 1: Let $I_{1i} \in \mathcal{I}_1$ and $I_{2j} \in \mathcal{I}_2$ represent two individuals in instance sets \mathcal{I}_1 and \mathcal{I}_2 . The individual matching task takes I_{1i} and I_{2j} as input and makes a decision whether $I_{1i} \equiv I_{2j}$ (in which case they are said to be matching) or not. This decision is made based on the comparison of the profiles of two individuals. A **profile** $P(I)$ is defined as a set of pairs $\{(a_i, V_i)\}$, where a_i represent attributes describing an individual (e.g., name, age, colour, etc.), each of which has a set of values V_i . The output of individual matching is a set of mappings $M = \{(I_{1i}, I_{2j})\}$ believed to represent equivalent individuals $I_{1i} \equiv I_{2j}$.

Most individual matching techniques follow the approach proposed in a seminal paper by Fellegi & Sunter [6], in which the decision is based on a **similarity function** $sim(P(I_1), P(I_2))$ which returns a degree of confidence that $I_1 \equiv I_2$. The similarity function commonly takes the form of aggregated similarity over attributes $sim(P(I_1), P(I_2)) = f_{agg}(\{sim_i(V_{1i}, V_{2i})\})$, where f_{agg} is an aggregation function and sim_i is a comparison function, which returns a degree of similarity between two values of the attribute a_i . The decision rule then takes the form of applying a **filtering criterion** which determines whether the confidence degree returned by the similarity function is sufficient to consider a pair of individuals as identical. The threshold-based criterion is commonly used: a mapping (I_1, I_2) is returned if $sim(P(I_1), P(I_2)) \geq t$, where t is a threshold.

2.2 Establishing a Decision Rule for Individual Matching

As can be seen from the description above, the key component of an individual matching method is the decision rule. For a given pair of datasets to link, a decision rule has to be established that incorporates comparisons between relevant pairs of properties using appropriate similarity functions, weights, and thresholds to obtain an adequate discriminative ability.

Some systems assume that a pre-established, generic similarity measure can be employed across domains. This approach is often followed by systems targeted for the global scale link discovery (e.g., OKKAM [13]), generic ontology

matching systems (e.g., RiMOM [9]), or systems which primarily rely on the dataset matching stage (e.g., CODI [12]). However, in most other cases, a dedicated decision rule has to be established for each link discovery task (i.e., each pair of datasets to link). Existing systems in the Semantic Web area take two different approaches to realise this:

Manual configuration. where the decision rule is specified by the user. Besides requiring user effort, the clear disadvantage of such an approach is that it relies on extensive knowledge from the user of the structure and content of the two datasets to link, as well as on a reasonable level of intuition regarding the performance of (often complex) similarity functions in a particular situation.

Learning from training data. where the appropriate decision rule is produced by analyzing the available labeled data. This method is followed, for example, by the ObjectCoref system [7]. This alleviates the need for user input to establish the decision rule, but requires the availability of a substantial set of robust training data (although some methods, like active learning [10] can reduce the required amount of data).

Here we investigate a third category of approaches that relies on the characteristics of the datasets and of the similarity distributions resulting from comparing them to establish high performing decision rules in an unsupervised way. Several solutions in the database research community proposed to use the distribution features of similarity functions. For example, in [2] individuals are clustered into matching and non-matching classes based on the structure of their neighbourhood rather than on simple threshold filtering. Zardetto et al [15] proposed to use prior knowledge about the features of the similarity distribution – namely, that correct mappings are dominant in the area of high similarity values and that matches are very rare in comparison with non-matches. These features are used to build a mixture model, which is later used for classifying candidate mappings into matching and non-matching.

Considering the task of linking to a reference repository, we can make several assumptions about the datasets and the desired instance matching output:

- **Assumption 1:** While different URIs are often used to denote the same entity in different repositories, distinct URIs within one dataset can be expected to denote distinct entities.
- **Assumption 2:** Datasets \mathcal{D}_1 and \mathcal{D}_2 have a strong degree of overlap.
- **Assumption 3:** A meaningful similarity function produces results in the interval $\{0..1\}$ and returns values close to 1.0 for pairs of matching individuals.

The method described in this paper proposes to use a genetic algorithm guided by a fitness criterion using these assumptions to assess the expected quality of a decision rule, and of the derived set of links. Our method goes a step further than existing methods, as it chooses an appropriate similarity function for a given matching task as well as a suitable filtering criterion, rather than relying

on given similarity functions. Hence, producing a solution requires selecting multiple parameters of the decision rule simultaneously, such as similarity functions, comparable attributes, and weights.

For such problems where a suitable complex function has to be found based on its desired output, genetic algorithms are known to perform well on many practical tasks, and have already been applied to the instance matching problem in the context of supervised learning [1], [8]. The idea here is to use such an approach to evolve a population of candidate solutions (i.e., decision rules) using selection and variation mechanisms to favour the “fittest” solutions in each generation, therefore presumably converging to decision rules that can be optimally applied to link the two given datasets.

3 Algorithm

Applying a genetic algorithm to the problem of optimizing a decision rule requires solving three issues: how relevant parameters of a decision rule are encoded as a set of genes, what fitness measure to use to evaluate candidate solutions, and how to use selection and variation operators to converge on a good solution.

3.1 Representing Individual Matching in Terms of a Genetic Algorithm

Definition 2: Let C_i represent a candidate solution to a given optimization task \mathcal{T} . Assume that C_i can be encoded as a set of numeric parameters. Then, the term **gene** g_{ij} denotes the j th parameter of the candidate solution C_i , **genotype** or **chromosome** $G(C_i) = \langle g_{i1}, \dots, g_{in} \rangle$ denotes a set of genes representing a candidate solution C_i , and **population** $\mathcal{G} = \{G_1, \dots, G_N\}$ represents a set of N chromosomes encoding candidate solutions for the task. A **fitness function** $F_{fit}(C_i)$ is a function which provides an estimation of the quality of a solution.

An initial population is used as a pool of candidates, from which the algorithm selects the best chromosomes according to the fitness function. In order to find a solution which optimizes the fitness function, the algorithm updates the initial population by using *selection* and *variation* operators:

- *Selection* chooses a subset of chromosomes in the original population to be used in the creation of the new one.
- *Variation* changes the genes of the selected chromosomes to generate new candidate solutions from the old ones. Commonly used variation operators include *crossover*, which recombines elements of several “parent” chromosomes to produce several new chromosomes (or “children”), and *mutation*, which produces a new chromosome by randomly tweaking the genes of the original one.

¹ The term “individual” is used both in the Semantic Web domain to denote ontological instances and in the evolutionary computation area, where it refers to candidate solutions. To avoid confusion, we use it only in its first sense, while using the term “candidate solution” when talking about the output of the genetic algorithm.

The updated population is created by applying these operators to selected chromosomes from the original one. Then, the same steps are performed for the updated population, and the algorithm continues iterating until the optimal solution (or one sufficiently close to the optimum) is produced or a termination condition is satisfied: e.g. maximal number of iterations is reached or the fitness of the population does not improve for a long time. The candidate solution $C_{best} = \operatorname{argmax}(F_{fit}(C_i))$ is returned by the algorithm as its output.

To apply a genetic algorithm to the individual matching problem, we need to represent candidate decision rules as a set of genes. Similarly to many existing approaches (see section 2), we represent a decision rule using an aggregated attribute similarity function.

Definition 3: A *decision rule* for an individual matching task is defined as: $\operatorname{filt}(\operatorname{sim}(P(I_1), P(I_2)))$ where $\operatorname{sim}(P(I_1), P(I_2))$ is the **similarity function** comparing profiles of two individuals, and $\operatorname{filt}(\operatorname{sim}(P(I_1), P(I_2)))$ is a **boolean filtering function**. The similarity function takes the form

$$\operatorname{sim}(P(I_1), P(I_2)) = f_{agg}(w_{11}\operatorname{sim}_{11}(V_{11}, V_{21}), \dots, w_{mn}\operatorname{sim}_{mn}(V_{1m}, V_{2n}))$$

- sim_{ij} is the function which measures similarity between the values of the attributes a_{1i} of $P(I_1)$ and a_{2j} of $P(I_2)$,
- w_{ij} is a numeric weight ($0 \leq w_{ij} \leq 1$),
- f_{agg} is an aggregation function.

We considered two alternative filtering criteria: the *threshold-based* one and the *nearest neighbour* one. The former requires that $\operatorname{sim}(P(I_1), P(I_2)) \geq t$, where t is a threshold value. The latter chooses for each instance I_1 in the source dataset such I_2 that $\operatorname{sim}(P(I_1), P(I_2)) = \max(\operatorname{sim}(P(I_1), P(I_j)))$. This criterion is applicable in cases where we expect each I_1 to have a matching I_2 .

Each of these parameters is represented by a gene in the following way:

- sim_{ij} are encoded as nominal values representing corresponding attribute similarity functions (or *nil*, if a_{1i} and a_{2j} are not compared). We included a number of character-based functions (edit distance, Jaro, I-Sub, etc., and the corresponding token-based similarity metrics. The latter divide both string values into sets of tokens, then compare each pair of tokens using a character-based similarity function and try to find the best match between them.
- Weights of each attribute comparison pair w_{ij} and the threshold t are encoded using their real values.
- f_{agg} is encoded as a nominal value representing one of two types of aggregation functions: weighted average $\operatorname{avg}(P(I_1), P(I_2)) = \frac{\sum w_{ij}\operatorname{sim}_{ij}(a_{1i}, a_{2j})}{\sum w_{ij}}$ and maximum $\max(P(I_1), P(I_2)) = \max(\{w_{ij}\operatorname{sim}_{ij}(a_{1i}, a_{2j})\})$. In the latter case the weights w_{ij} can only take values 0 or 1.

These genotypes are evaluated by applying the decision rule to the matching task and calculating the fitness function.

3.2 Fitness Functions: Pseudo-F-measure and Neighbourhood Growth

In the absence of labelled data it is not possible to estimate the quality of a set of mappings accurately. However, there are indirect indicators corresponding to “good characteristics” of sets of links which can be used to assess the fitness of a given decision rule. To establish such indicators, we rely on the assumptions we made about the matching task. Traditionally, the quality of the matching output is evaluated by comparing it with the set of true mappings M^t and calculating the precision p and recall r metrics. Precision is defined as $p = \frac{|tp|}{|tp|+|fp|}$, where tp is a set of true positives (mappings $m = (I_1, I_2)$ such that both $m \in M$ and $m \in M^t$) and fp is a set of false positives ($m \in M$, but $m \notin M^t$). Recall is calculated as $r = \frac{|tp|}{|tp|+|fn|}$, where fn is a set of false negatives ($m \notin M$, but $m \in M^t$). In the absence of gold standard mappings, we use Assumption 1 to formulate the pseudo-precision and pseudo-recall measures in the following way:

Definition 4: Let M represent a set of mappings (I_i, I_j) between two sets of individuals $\mathcal{I}_1, \mathcal{I}_2$ such that $I_i \in \mathcal{I}_1, I_j \in \mathcal{I}_2$. Then, **pseudo-precision** is the value $p^\sim = \frac{|\{I_i|\exists I_j:(I_i,I_j)\in M\}|}{\sum_i |\{I_j|(I_i,I_j)\in M\}|}$, and **pseudo-recall** is the value $r^\sim = \frac{|M|}{\min(|\mathcal{I}_1|, |\mathcal{I}_2|)}$.

In an ideal case where $p = 1$, if Assumption 1 holds, then $p^\sim = 1$: of two mappings from the same individual one is necessarily an error. Similarly, in case where $r = 1$, the number of returned mappings will be equal to the size of the overlap between two instance sets $|M| = n_o = |\mathcal{I}_1 \cap \mathcal{I}_2|$, and the *pseudo-recall* value $r^\sim = \frac{|M|}{n_o} = 1$. However, estimating the true recall is problematic since n_o is not known in advance. From Assumption 1 it follows that $n_o \leq \min(|\mathcal{I}_1|, |\mathcal{I}_2|)$, while $n_o = \min(|\mathcal{I}_1|, |\mathcal{I}_2|)$ if one instance set is a subset of another. Incorrect estimation of n_o can be misleading for the genetic algorithm: it can result in “lenient” decision rules being favored and, in consequence, to many false positives in the resulting solution. To deal with such cases, we reduce the impact of incorrect recall estimations in the final fitness function.

A standard metric combining precision and recall is the F-measure $F_\beta = \frac{(1+\beta^2) \cdot p \cdot r}{\beta^2 \cdot p + r}$, where β characterizes the preference of recall over precision, and $\beta = 1$ means equal importance of both. To reduce the impact of recall, we used $\beta = 0.1$ and the pseudo-F-measure $F_{0.1}^\sim = \frac{1.01 p^\sim \cdot r^\sim}{0.01 \cdot p^\sim + r^\sim}$. In this way, solutions which increase precision are favored, while recall is only used to discriminate between solutions with similar estimated precision. This “cautious” approach is also consistent with the requirements of many real-world data linking scenarios, as the cost of an erroneous mapping is often higher than the cost of a missed correct mapping.

In order to incorporate Assumption 3, the final fitness function gives a preference to the solutions which accept mappings with similarity degrees close to 1: $F_{fit}^\sim = F_{0.1}^\sim \cdot (1 - (1 - sim_{avg})^2)$. In this way, the fitness function is able to discriminate between such decision rules as $avg(0.5 \cdot jaro(name, label), 0.5 \cdot edit(birthYear, yearOfBirth)) \geq 0.98$ and $avg(0.05 \cdot jaro(name, label), 0.05 \cdot edit(birthYear, yearOfBirth), 0.9 \cdot edit(name, yearOfBirth)) \geq 0.098$. While

these two rules would produce the same output in most cases, comparing irrelevant attributes (like *name* and *yearOfBirth*) is not desirable, because it increases a possibility of spurious mappings without adding any value.

While we used F_{fit}^{\sim} as the main fitness criterion, to test the effect of the choice of a fitness function on the performance of the genetic algorithm, we implemented an alternative fitness function: the *neighbourhood growth* function F_{fit}^{NG} . While the pseudo F-Measure tries to estimate the quality of resulting mappings to guide the evolution of candidate solutions, F_{fit}^{NG} tries to exploit the desired property of a “good” similarity function: namely, that it should be able to discriminate well between different possible candidate mappings. To measure this property, we adapt the neighbourhood growth indicator defined in [2] to achieve an optimal clustering of instance matching results for a pre-defined similarity function, as an alternative to the threshold-based filtering criterion. We adapt this indicator as an alternative fitness criterion for selecting the most appropriate similarity functions.

Definition 5: Let M_x represent a set of mappings (I_x, I_{x_j}) between an individual $I_i \in \mathcal{I}_1$ and a set of individuals $I_{x_j} \in \mathcal{I}_{\S} \subseteq \mathcal{I}_{\in}$. Let $sim_{max} = \max(sim(P(I_x), P(I_{x_j})))$. Then, **neighbourhood growth** $NG(I_x)$ is defined as the number of mappings in M_x such that their similarity values are higher than $1 - c \cdot (1 - sim_{max})$, where c is a constant.

Intuitively, high values of $NG(I_x)$ indicate that the neighbourhood of an instance is “cluttered”, and the similarity measure cannot adequately distinguish between different matching candidates. Then the fitness function for a set of compared instance pairs M is defined as $F_{fit}^{NG} = 1/avg_x(NG(I_x))$. As this function does not require applying the filtering criterion, it only learns the similarity function, but not the optimal threshold. However, the threshold can be determined after the optimal similarity function has been derived: t is selected in such a way that it maximises the F_{fit}^{\sim} function over a set of compared pairs.

3.3 Obtaining the Optimal Solution: Genetic Algorithm

The algorithm takes as input two instance sets \mathcal{I}_1 and \mathcal{I}_2 and two sets of potential attributes A_1 and A_2 . Each set of attributes A_i includes all literal property values at a distance l from individuals in \mathcal{I}_i . In our experiments we used $l = 1$, however, also including the paths of length 2 if an individual was connected to a literal through a blank node. In order to filter out rarely defined properties, we also remove all attributes a_{ij} for which $\frac{|P(I_i)|_{a_{ij} \in P(I_i), I_i \in \mathcal{I}_i}|}{|\mathcal{I}_i|} < 0.5$.

As the first step, the algorithm initializes the population of size N . For the initial population, all values of the genotype are set in the following way:

- A set of k pairs of attributes (a_{1i}, a_{2j}) is selected randomly from the corresponding sets A_1 and A_2 .
- For these pairs of attributes the similarity functions sim_{ij} and the corresponding weights w_{ij} are assigned randomly while for all others are set to *nil*.

- The aggregation function and the threshold are initialized with random values, and the weights are normalized so that $\sum w_{ij} = 1$.

All initial solutions only compare a single pair of attributes ($k = 1$): this is done to identify highly discriminative pairs of attributes at the early iterations, and then improve these solutions incrementally.

Each iteration of the algorithm consists of two stages: selection and reproduction. At the selection stage, each candidate solution is applied to produce mappings between individuals from \mathcal{I}_1 and \mathcal{I}_2 . In case of large-scale datasets, random sampling can be applied, so that the solutions are only applied to a subset $\mathcal{I}_1^S \subseteq \mathcal{I}_1$. The calculated F_{fit} fitness measure is used for the selection of candidate solutions for reproduction. Our algorithm uses the standard roulette wheel selection operator: the probability of a chromosome being selected is proportionate to its F_{fit} fitness. At the reproduction stage, a new population of chromosomes is generated by three different operators: elitist selection, crossover, and mutation. In the new population, the proportion of chromosomes produced by each operator is proportional to its rate: elitist selection rate r_{el} , crossover rate r_c , and mutation rate r_m ($r_{el} + r_c + r_m = 1$). Elitist selection copies the best subset of chromosomes from the previous population. The crossover operator takes two parent chromosomes and forms a pair of “children”: each gene of the parent is passed to a randomly chosen child, while another child inherits a corresponding gene of the second parent. Finally, mutation modifies one of the genes of the original chromosome in one of the following ways:

- Adding or removing a comparison between attributes with a probability p_{att}^m . The operator either changes the similarity function for a pair of attributes to *nil* or selects a random similarity function and weight for a pair of attributes not compared in the original chromosome. The probability of adding a component (versus removing one) is calculated as $p_{add} = \frac{1}{n^+}$, where n^+ is the number of non-*nil* similarity comparisons in the original solution.
- Changing one of the weights w_{ij} for a pair of attributes where $sim_{ij} \neq nil$, with a probability p_{wgt}^m . The value of the change is calculated as $\frac{0.8 \cdot rnd + 0.2}{n^+}$, where *rnd* is a random number between 0 and 1.
- Changing a non-*nil* similarity function for a pair of attributes into a randomly selected one with a probability p_{sym}^m .
- Modifying the threshold value with the probability p_t^m : the algorithm decides whether the current threshold should be increased or decreased with the probability 0.5. The new threshold is set as $t_{new} = t_{old} \pm \Delta t$, where $\Delta t = rnd \cdot (1 - p^{\sim})(1 - t_{old})$ for increase and $rnd \cdot (1 - r^{\sim})t_{old}$ for decrease. The rationale behind this is to make bigger steps if precision/recall values are far from desired.
- Changing the aggregation function with p_{agg}^m .

At the new iteration, chromosomes in the updated population are again evaluated using the F_{fit} fitness function, and the process is repeated. The algorithm stops if the pre-defined number of iterations n_{iter} is reached or the algorithm

converges before this: i.e., the average fitness does not increase for n_{conv} generations. The phenotype with the best fitness in the final population is returned by the algorithm as its result.

4 Evaluation

To validate our method, we performed experiments with two types of datasets. First, we tested our approach on the benchmark datasets used in the instance matching tracks of the OAEI 2010 and OAEI 2011 ontology matching competitions², to compare our approach with state-of-the-art systems. Second, we used several datasets extracted from the linked data cloud to investigate the effect of different parameter settings on the results.

4.1 Settings

As discussed above, a genetic algorithm starts with an initial population of random solutions, and iteratively create new generations through selection, mutation and crossover. In our experiments, we used the following default parameters:

- rates for different recombination operators: $r_{el} = 0.1$, $r_m = 0.6$, and $r_c = 0.3$.
- rates for different mutation options: $p_{att}^m = 0.3$, $p_{wgt}^m = 0.15$, $p_{sym}^m = 0.15$, $p_t^m = 0.3$, $p_{agg}^m = 0.1$ (ensuring equivalent probabilities for modifying the list of compared properties, comparison parameters, and the threshold).
- termination criterion: $n_{iter} = 20$ (found to be sufficient for convergence in most cases).
- fitness function: F_{fit}^{\sim} , except when comparing F_{fit}^{\sim} with F_{fit}^{NG}

The genetic algorithm is implemented as a method in the KnoFuss architecture [11]. Relevant subsets of two datasets are selected using SPARQL queries. Each candidate decision rule is used as an input of the KnoFuss tool to create the corresponding set of links. To reduce the computation time, an inverted Lucene³ index was used to perform *blocking* and pre-select candidate pairs. Each individual in the larger dataset was indexed by all its literal properties. Each individual in the smaller dataset was only compared to individuals returned by the index when searching on all its literal properties, and pairs of compared individuals were cached in memory. Common pre-processing techniques (such as removing stopwords and unifying synonyms) were applied to the literal properties.

4.2 Benchmark Test

The OAEI 2010 benchmark contains three test cases: *Person1* and *Person2*, which contain artificially distorted records of people, and *Restaurants*, which includes data about restaurants from the RIDDLE repository⁴. Two versions

² <http://oaei.ontologymatching.org/>

³ <http://lucene.apache.org>

⁴ <http://www.cs.utexas.edu/users/ml/riddle/data.html>

Table 1. Comparison of F1-measure with other tools on the OAEI 2010 benchmark [4]

Dataset	KnowFuss+GA	ObjectCoref	ASMOV	CODI	LN2R	RiMOM	FBEM
Person1	1.00	1.00	1.00	0.91	1.00	1.00	N/A
Person2	0.99	0.95	0.35	0.36	0.94	0.97	0.79
Restaurant (OAEI)	0.78	0.73	0.70	0.72	0.75	0.81	N/A
Restaurant (fixed)	0.98	0.89	N/A	N/A	N/A	N/A	0.96

of the *Restaurants* dataset exist: the version originally used in the OAEI 2010 evaluation which contained a bug (some individuals included in the gold standard were not present in the data), and the fixed version, which was used in other tests (e.g. [13], [7]). To be able to compare with systems which used both variants of the dataset, we also used both variants in our experiments. The OAEI 2011 benchmark includes seven test cases, which involve matching three subsets of the New York Times linked data (people, organisations, and locations) with DBpedia, Freebase, and Geonames datasets.

We compared our algorithm with the systems participating in the OAEI 2010 tracks as well as with the FBEM system [13], whose authors provided the benchmark datasets for the competition. We report in Table 1 on the performance of the KnowFuss system using decision rules learned through our genetic algorithm (noted KnowFuss+GA) as the average F1-Measure obtained over 5 runs of the algorithm with a population size $N = 1000$. The solution produced by the

Table 2. Example decision rules found by the algorithm with $N = 1000$

Test case	Similarity function	Threshold
Person1	$\max(\text{tokenized-jaro-winkler}(\text{soc_sec_id};\text{soc_sec_id});$ $\text{monge-elkan}(\text{phone_number};\text{phone_number}))$	≥ 0.87
Person2	$\max(\text{jaro}(\text{phone_number};\text{phone_number});$ $\text{jaro-winkler}(\text{soc_sec_id};\text{soc_sec_id}))$	≥ 0.88
Restaurants (OAEI)	$\text{avg}(0.22*\text{tokenized-smith-waterman}(\text{phone_number};\text{phone_number});$ $0.78*\text{tokenized-smith-waterman}(\text{name};\text{name}))$	≥ 0.91
Restaurants (fixed)	$\text{avg}(0.35*\text{tokenized-monge-elkan}(\text{phone_number};\text{phone_number});$ $0.65*\text{tokenized-smith-waterman}(\text{name};\text{name}))$	≥ 0.88

genetic algorithm managed to achieve the highest F1-measure on 3 out of 4 datasets and the second highest F1-measure on 1 out of 4. Examples of produced decision rules are provided in Table 2. We observed that the algorithm took less time on identifying discriminative pairs of properties and the aggregation function and more on tuning weights and attribute similarity functions. To test the robustness of the results achieved by the algorithm with different settings, we performed tests on the benchmark datasets varying the crossover rates r_c , and mutation rate r_m . Surprisingly, varying the crossover rate and the mutation rate did not lead to significant changes in the results, except for extreme values. These parameters mostly affected the number of generations needed to

Table 3. Comparison of F1-measure with other tools on the OAEI 2011 benchmark [5](#)

Dataset	KnoFuss+GA	AgreementMaker	SERIMI	Zhishi.links
DBpedia (locations)	0.89	0.69	0.68	0.92
DBpedia (organisations)	0.92	0.74	0.88	0.91
DBpedia (people)	0.97	0.88	0.94	0.97
Freebase (locations)	0.93	0.85	0.91	0.88
Freebase (organisations)	0.92	0.80	0.91	0.87
Freebase (people)	0.95	0.96	0.92	0.93
Geonames	0.90	0.85	0.80	0.91
Average	0.93	0.85	0.89	0.92

converge to the optimal solution, and the algorithm usually converged well before 20 generations⁵.

Given the larger scale of the OAEI 2011 benchmark, to speed up the algorithm we used random sampling with the sample size $s = 100$ and reduced the population size to $N = 100$. To improve the performance, a post-processing step was applied: the 1-to-1 rule was re-enforced, and for a source individual only 1 mapping was retained. As shown in Table [3](#), these settings were still sufficient to achieve high performance: the algorithm achieved the highest $F1$ measure on 4 test cases out of 7 and the highest average $F1$ measure. These results verify our original assumptions that (a) the fitness function based on the pseudo- F -measure can be used as an estimation of the actual accuracy of a decision rule and (b) the genetic algorithm provides a suitable search strategy for obtaining a decision rule for individual matching.

4.3 LOD Datasets

To test the reusability of our method in different real-world scenarios, we have defined the following three matching tasks:

Music Contributors. As a source dataset, we selected a list of music contributors from the LinkedMDB dataset⁶. This dataset of 3995 individuals was matched against the set of all people from DBpedia⁷ (363751 individuals).

The gold standard was constructed manually and included 1182 mappings.

Book Authors. To construct this dataset, we extracted a set of 1000 individuals describing book authors from the BNB dataset⁸ (from the first part of the dump, we selected 1000 authors with the highest number of published books). This dataset was also matched against the set of all people from DBpedia. The gold standard was constructed manually and included 219 correct mappings.

Research Papers. To generate a matching task with a larger number of reliable gold standard mappings, we used a subset of 10000 research publications

⁵ The datasets and test results are available for download from our website: <http://kmi.open.ac.uk/technologies/knofuss/knofuss-GA-tests.zip>

⁶ <http://www.linkedmdb.org/>

⁷ <http://dbpedia.org>

⁸ <http://www.archive.org/details/Bibliographica.orgBnbDataset>

represented in the L3S-DBLP dataset⁹ (out of the snapshot of 366113 publications included in the BTC 2010 dataset¹⁰). For these publications, we extracted their RDF descriptions from the DOI web-site¹¹. We used equivalent DOI codes to create the gold standard and then removed corresponding properties from respective datasets to prevent the algorithm from using them as an easy solution.

On each of these datasets, we applied the algorithm with the same default settings as used in the benchmark tests. We performed the experiments using two different fitness functions: the unsupervised F_{fit}^{\sim} fitness function and the actual $F1$ -measure produced using the gold standard dataset. The latter case represents an ideal scenario, in which a complete set of labeled data is available in advance, and the algorithm only has to produce an optimal decision rule which would approximate this data. For *Music contributors* and *Book authors*, we varied the population size N in order to estimate the necessary number of candidate solutions which the algorithm has to test before achieving stable performance. The results for these datasets are summarised in Table 4, which shows average precision, recall, and $F1$ -measure achieved using two different fitness functions, as well as the standard deviation of $F1$ measure $\sigma F1$ over 5 runs and the time of a single run for the unsupervised case¹². In both cases, F_{fit}^{\sim} allowed reaching

Table 4. Results with different population size

Dataset	Pop. size N	F1-fitness (ideal case)			F_{fit}^{\sim} -fitness (unsupervised)				
		Precision	Recall	F1	Precision	Recall	F1	$\sigma F1$	Time (s)
Music contributors	50	0.92	0.92	0.92	0.90	0.90	0.90	0.021	520
	100	0.91	0.93	0.92	0.92	0.91	0.92	0.003	931
	500	0.91	0.93	0.92	0.92	0.92	0.92	0.003	4197
Book authors	50	0.90	0.93	0.91	0.66	0.69	0.68	0.022	753
	100	0.98	0.95	0.97	0.78	0.89	0.82	0.13	1222
	500	0.99	0.98	0.98	0.91	0.91	0.91	0.009	7281

high performance ($F1$ above 0.9), and increasing the population size N led to improvement in performance as well as more robust results (lower $\sigma F1$). In fact, for the *Music contributors* test case, the results produced using F_{fit}^{\sim} and the ideal case $F1$ were almost equivalent. For the *Research papers* dataset (Table 5), we trained the algorithm on several samples taken from the DOI dataset and then applied the resulting decision rules to the complete test case (10000 individuals in the DOI dataset). This was done to emulate use cases involving large-scale repositories, in which running many iterations of the genetic algorithm over complete datasets is not feasible. From Table 5 we can see that starting from 100 sample individuals the algorithm achieved stable performance, which is consistent

⁹ <http://dblp.l3s.de/>

¹⁰ <http://km.aifb.kit.edu/projects/btc-2010/>

¹¹ <http://dx.doi.org/>

¹² Experiments were performed on a Linux desktop with two Intel Core 2 Duo processors and 3GB of RAM.

Table 5. Results obtained for the *Research papers* dataset (for all sample sizes, population size $N = 100$ was used)

Sample size	F1-fitness (ideal case)			F_{fit}^{\sim} -fitness (unsupervised)				Complete set			
	Precision	Recall	F1	Precision	Recall	F1	$\sigma F1$	Time (s)	Precision	Recall	F1
50	0.50	0.76	0.60	0.58	0.36	0.44	0.063	162	0.68	0.22	0.33
100	0.95	0.88	0.91	0.998	0.72	0.83	0.068	255	0.995	0.68	0.81
500	0.96	0.85	0.90	0.99	0.73	0.84	0.046	842	0.98	0.75	0.85
1000	0.95	0.88	0.91	0.99	0.67	0.79	0.065	3667	0.997	0.71	0.83

with the results achieved for the OAEI 2011 benchmark. Applying the resulting decision rules to the complete dataset also produced results with precision and recall values similar to the ones achieved on the partial sample. Finally, to test

Table 6. Comparing the F_{fit}^{\sim} and F_{fit}^{NG} fitness functions

Dataset	F_{fit}^{\sim} -fitness			NG -fitness		
	Precision	Recall	F1	Precision	Recall	F1
Music contributors	0.92	0.91	0.92	0.90	0.91	0.91
Book authors	0.78	0.89	0.82	0.97	0.78	0.85
NYT-Geonames	0.88	0.82	0.84	0.87	0.92	0.89
NYT-Freebase (people)	0.60	0.97	0.74	0.47	0.66	0.55

the effect of the chosen fitness function on the performance, we compared the pseudo-F-measure F_{fit}^{\sim} and neighbourhood growth F_{fit}^{NG} fitness functions. We applied the algorithm to the *Music contributors* and *Book authors* datasets, as well as to the NYT-Geonames and NYT-Freebase (people) test cases from the OAEI 2011 benchmark (without applying post-processing). The results reported in Table 6 show that both functions are able to achieve high accuracy with F_{fit}^{\sim} providing more stable performance. This validates our initial choice of F_{fit}^{\sim} as a suitable fitness criterion and reinforces our assumption that features of the similarity distribution can indirectly serve to estimate the actual fitness.

5 Conclusion and Future Work

In this paper, we proposed a method which exploits expected characteristics of “good” sets of mappings to estimate the quality of results of the individual matching task. We formalised these characteristics to propose a fitness function for a genetic algorithm, which derives a suitable decision rule for a given matching task. Experiments, which we performed with both benchmark and real-world datasets, have validated our initial assumptions and have shown that the method is able to achieve accuracy at the level of the top-performing state-of-the-art data linking systems without requiring user configuration, training data, or external knowledge sources.

We plan to use the results presented in this paper to pursue several promising research directions, in particular, combining our approach with more knowledge-involving dataset matching methods. On the one hand, dataset matching systems

have to rely on individual matching techniques to provide initial sets of mappings for refining. For such systems, using initial mappings of better quality can be beneficial. On the other hand, domain knowledge can be used to improve the unsupervised fitness functions, for example to reduce the fitness of decision rules whose results violate ontological restrictions.

Acknowledgements. Part of this research has been funded under the EC 7th Framework Programme, in the context of the SmartProducts project (231204).

References

1. de Carvalho, M.G., Laender, A.H.F., Goncalves, M.A., da Silva, A.S.: A genetic programming approach to record deduplication. *IEEE Transactions on Knowledge and Data Engineering* 99(PrePrints) (2010)
2. Chaudhuri, S., Ganti, V., Motwani, R.: Robust identification of fuzzy duplicates. In: *ICDE 2005*, pp. 865–876 (2005)
3. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering* 19(1), 1–16 (2007)
4. Euzenat, J., et al.: Results of the ontology alignment evaluation initiative 2010. In: *Workshop on Ontology Matching (OM 2010), ISWC 2010* (2010)
5. Euzenat, J., et al.: Results of the ontology alignment evaluation initiative 2011. In: *Workshop on Ontology Matching (OM 2011), ISWC 2011* (2011)
6. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. *Journal of American Statistical Association* 64(328), 1183–1210 (1969)
7. Hu, W., Chen, J., Qu, Y.: A self-training approach for resolving object coreference on the semantic web. In: *WWW 2011*, pp. 87–96 (2011)
8. Isele, R., Bizer, C.: Learning linkage rules using genetic programming. In: *Workshop on Ontology Matching (OM 2011), ISWC 2011, Bonn, Germany* (2011)
9. Li, J., Tang, J., Li, Y., Luo, Q.: RiMOM: A dynamic multistrategy ontology alignment framework. *IEEE Transactions on Knowledge and Data Engineering* 21(8), 1218–1232 (2009)
10. Ngonga Ngomo, A.C., Lehmann, J., Auer, S., Höffner, K.: RAVEN - active learning of link specifications. In: *Workshop on Ontology Matching (OM 2011), ISWC 2011* (2011)
11. Nikolov, A., Uren, V., Motta, E., de Roeck, A.: Integration of Semantically Annotated Data by the KnoFuss Architecture. In: Gangemi, A., Euzenat, J. (eds.) *EKAW 2008. LNCS (LNAI)*, vol. 5268, pp. 265–274. Springer, Heidelberg (2008)
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. LNCS*, vol. 6089, pp. 334–348. Springer, Heidelberg (2010)
13. Stoermer, H., Rassadko, N., Vaidya, N.: Feature-Based Entity Matching: The FBEM Model, Implementation, Evaluation. In: Pernici, B. (ed.) *CAiSE 2010. LNCS*, vol. 6051, pp. 180–193. Springer, Heidelberg (2010)
14. 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)
15. Zardetto, D., Scannapietro, M., Catarci, T.: Effective automated object matching. In: *ICDE 2010*, pp. 757–768 (2010)

Graph Kernels for RDF Data

Uta L \ddot{o} sch¹, Stephan Bloehdorn², and Achim Rettinger¹

¹ Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany

{uta.loesch, rettinger}@kit.edu

² IBM Germany, 12277 Berlin, Germany

bloehdorn@de.ibm.com

Abstract. The increasing availability of structured data in Resource Description Framework (RDF) format poses new challenges and opportunities for data mining. Existing approaches to mining RDF have only focused on one specific data representation, one specific machine learning algorithm or one specific task. Kernels, however, promise a more flexible approach by providing a powerful framework for decoupling the data representation from the learning task. This paper focuses on how the well established family of kernel-based machine learning algorithms can be readily applied to instances represented as RDF graphs. We first review the problems that arise when conventional graph kernels are used for RDF graphs. We then introduce two versatile families of graph kernels specifically suited for RDF, based on intersection graphs and intersection trees. The flexibility of the approach is demonstrated on two common relational learning tasks: entity classification and link prediction. The results show that our novel RDF graph kernels used with Support Vector Machines (SVMs) achieve competitive predictive performance when compared to specialized techniques for both tasks.

1 Introduction

The RDF-formatted data on the World Wide Web leads to new types of distributed information systems and poses new challenges and opportunities to data mining research. As an official standard of the World Wide Web Consortium (W3C), the *Resource Description Framework (RDF)* establishes a universal graph-based data model not intuitively suited for standard machine learning (ML) algorithms. Recent efforts of research, industry and public institutions in the context of Linked Open Data (LOD) initiatives have led to considerable amounts of RDF data sets being made available and linked to each other on the web [1]. Besides that, there is progress in research on extracting RDF from text documents [2]. Consequently, the question of systematically exploiting the knowledge therein by data mining approaches becomes highly relevant.

This paper focuses on making instances represented by means of RDF graph structures available as input to existing ML algorithms, which can solve data mining tasks relevant to RDF, such as class-membership prediction, property value prediction, link prediction or clustering. As an example, consider the mining of the social network emerging from the linked user profiles available in FOAF, a popular RDF-based vocabulary [3]. Relevant tasks in this setting include the identification of similar users (e.g. for identity resolution) or the prediction of user interests (e.g. for recommendations).

Existing approaches to mining the Semantic Web (SW) have either focused on one specific semantic data representation [4, 5] based on RDF or on one of the specific tasks mentioned above, i.e. the data representation and the learning algorithm have been devised specifically for the problem at hand [6, 7, 8, 9, 10].

In contrast, *kernels* [11] promise a highly flexible approach by providing a powerful framework for decoupling the data representation from the learning task: Specific *kernel functions* can be deployed depending on the format of the input data and combined in a “plug-and-play”-style with readily available *kernel machines* for all standard learning tasks. In this context, the challenge of learning from RDF data can be reformulated as designing adequate kernel functions for this representation. In fact, various kernel functions for general graph structures have been proposed over the last years. However, the properties of RDF graphs are different from general graphs: e.g. graphs representing chemical compounds usually have few node labels which occur frequently in the graph and nodes in these graphs have a low degree. In contrast, RDF node labels are used as identifiers occurring only once per graph and nodes may have a high degree.

In this paper, we bridge the gap between the too general and too specialized approaches to mining RDF data. We first review existing approaches for mining from semantically annotated data on the one hand and from general graphs on the other. We discuss the problems of these approaches with respect to their flexibility (e.g. applicability in various tasks, combination with different learning algorithms, different data representations etc.) and their suitability as data representations for RDF-type data. Improving on that, we introduce two versatile families of graph kernels based on intersection graphs and intersection trees. We discuss why our approach can be more easily applied by non-experts to solve a wider range of data mining tasks using a wider range of ML techniques on a wider range of RDF data sets compared to the more specialized approaches found in the related work. To the best of our knowledge, these are the first kernels which can interface between any RDF graph and any kernel machine, while exploiting the specifics of RDF. By applying standard Support Vector Machines (SVM) to RDF and RDF(S) data sets we show how our kernels can be used for solving two different learning problems on RDF graphs: property value prediction and link prediction. Despite its broader applicability we achieve comparable performance to methods which are either more specialized on a specific data format and data mining task or too general to exploit the specifics of RDF graphs efficiently.

In Section 2, we introduce foundations on kernel methods and discuss related work. In Section 3 we describe a new family of kernel functions and their applicability to learning from RDF data. In Section 4, we report experiments which demonstrate their flexibility and performance. We conclude with a discussion and outlook in Section 5.

2 Preliminaries and Related Work

In this section, we briefly introduce the required formalisms for kernel methods and present related work with respect to mining Semantic Web (SW) data. After introducing kernel methods and the learning problems arising from SW data, we discuss the applicability of general graph kernels to semantic data representations, before detailing on the existing methods for specific learning problems in the context of the SW.

2.1 Preliminaries: Kernels

The core idea behind the use of kernel functions is the decoupling of the employed learning algorithms from the representations of the data instances under investigation. Kernel-based machine learning algorithms abandon the explicit vector representations of data items by means of the *kernel function*, a similarity function which maintains a geometric interpretation as the inner product of two vectors in some, potentially even unknown, feature space. While SVMs for classification can safely be regarded as the best known kernel machine, kernel machines have been devised for many more supervised and unsupervised learning problems: kernel-k-means, SVM regression, one-class SVM, etc. The algorithms are formalized such that it is sufficient to access the evaluations of the inner product $\langle x, x' \rangle$ of two vectors x, x' . As a consequence, it is possible to replace the inner products $\langle \cdot, \cdot \rangle$ in the unkernelized algorithms by any valid *kernel function* which yields the same result as the inner product without the explicit representation of the training instances as vectors.

Definition 1 (Kernel Function). Any function $\kappa : X \times X \rightarrow \mathbb{R}$ on objects x, x' from some input domain X that satisfies $\kappa(x, x') = \langle \phi(x), \phi(x') \rangle$, is a valid kernel, whereby ϕ is a mapping function (feature representation) from \mathcal{X} to a feature space \mathcal{H} .

Technically, the set of valid kernel functions exactly corresponds to the set of so-called *positive semi-definite functions* [11]. It can be shown that kernel functions are closed under sum, product, multiplication by a positive scalar and combination with well-known kernel modifiers. The interested reader is pointed to [11] for a more comprehensive introduction to kernel methods.

2.2 Preliminaries: Data Mining Tasks for RDF

The goal in this work is to make kernel machines available to learning tasks which use SW data as input. On RDF instances, variants of classical ML tasks, like classification, can be performed. This could be the assignment of individuals to classes, like `person`-instances to `foaf:person`-classes if this information is only partially known. The prediction of features of instances, like `foaf:topic_interest` of a person, is another classification task which we will call *property value prediction*. An important task that has been of increasing interest is *relation or link prediction*. In our example this could be `foaf:knows` recommendations to persons. Another important conventional data mining task applicable to RDF data is the *clustering* of instances like similar persons (sometimes also called *group detection* [12]).

Two kinds of existing approaches to mining RDF data exist: either, general graph kernels are used (see Section 2.3) or very specific approaches are devised (see Section 2.4). We argue that the first category does not exploit the properties of RDF data adequately and that approaches of the second category are, from a practical perspective, too specific with respect to the learning problem, data representation, or dataset.

2.3 Related Work: General Graph Kernels

Graphs present a very generic model for representing various kinds of data. Therefore, the problem of making graphs amenable for kernel machines has received considerable

interest. In the spirit of the *Convolution Kernel* by Haussler [13], which represents a generic way of defining kernels, kernel functions for general graphs have been devised by counting common subgraphs of two graphs. While the subgraph isomorphism problem, i.e. the problem of identifying whether a given input graph contains another input graph as its subgraph, is known to be NP-complete, the search for common subgraphs with specific properties can often be performed more efficiently. Thus, kernel functions have been defined for various substructures such as walks [14], cycles [15] and trees [16]. Any of these kernels is applicable to RDF data as long as it is defined on directed, labeled graphs. While these kernel functions are applicable in a wide range of applications, they do not exploit the specific properties of RDF data. RDF graphs are directed, labeled graphs in which nodes may be identified by their label, their URI.

2.4 Related Work: Specialized Methods for Mining Semantic Data

Besides the general graph kernels, specific approaches for mining from Semantic Data have been developed. Huang et al. [8] have proposed a link prediction method which is based on the matrix completion of the relation matrix. Thor et al. [10] have proposed a link prediction method which is based on finding frequent patterns in the data graph. Rettinger et al. [7] have proposed a method for integrating ontology-based constraints with statistical relational learning. These approaches are tailored towards specific applications and thus do not expose the general applicability of kernel methods. Fanizzi and d'Amato [4] propose a declarative kernel for semantic concept descriptions in the description logic (DL) \mathcal{ALC} . Structurally, these kernels are based on a representation of \mathcal{ALC} concepts in normal form. Fanizzi et al. [5] have proposed a different set of kernels, which can be applied directly to individuals. These kernel functions are based on the analysis of the membership of the individuals in a set of classes. These kernel functions are tailored towards specific data representations and rely on clean, DL-based formal ontologies which, in contrast to lightweight RDF-based data, only constitute a very small part of the “semantic” data sources published. Bicer et al. [9] have proposed a very general family of kernel functions which is based on a set of logical clauses that are generated and selected automatically. However, this approach can not be integrated “plug-and-play”-style with existing kernel machines as these have to be adapted in order to select a relevant subset of the defined features. Finally, Bloehdorn and Sure [6] have proposed a set of kernel functions for individuals based on a manual feature definition. This means that relevant kernels and corresponding features for the learning task and the dataset at hand have to be defined upfront.

The goal of our work is to define kernel methods for RDF which can be used with any existing kernel machine, which is not restricted to specific kinds of data, which can readily be applied to any RDF dataset, but which still exploits the specific properties of RDF-type data (in contrast to general graph data).

3 Kernel Functions for RDF Graphs – Design and Implementation

In this section, we present our core contribution: two classes of kernel functions based on *intersection graphs* and *intersection trees*, specifically tailored to the properties of RDF. We thereby rely on feature sets based on the graph structure underlying RDF data.

Definition 2 (RDF Graph). An RDF graph is defined as a set of triples of the form $G = \{(s, p, o)\} = (V, E)$, where the subject $s \in V$ is an entity, the predicate p denotes a property, and the object $o \in V$ is either another entity or, in case of a relation whose values are data-typed, a literal. The vertices $v \in V$ of G are defined by all elements that occur either as subject or object of a triple. Each edge $e \in E$ in the graph is defined by a triple (s, p, o) : the edge that connects s to o and has label p .¹

Generally, we will look at RDF entities as the instances for learning. For example, two sets of entities, identified by their Uniform Resource Identifiers (URIs) could be positive and negative classes in a classification scenario. The argument entities' neighborhood in the overall RDF graph forms the basis for their kernel-induced feature representations. Essentially, all proposed kernel functions are thus based on a neighborhood graph which is obtained by a breadth-first search *up to depth* k starting from the entity of interest. We have defined two versions of the neighborhood graph: the *intersection graph* (see Section 3.1) and the *intersection tree* (see Section 3.2).

We define RDF kernels in a similar manner to other graph kernels by adopting the idea of counting subgraphs with a specific structure in the input graphs. The essential difference is that, as RDF builds on unique node labels, each RDF subgraph can occur at most once in the input graph. This is not the case in general graphs, where it is common that several nodes carry the same label – thus yielding potentially several equivalent subgraphs. Therefore, when calculating the kernel function between two RDF graphs, it is not necessary to identify the interesting structures and their frequencies in the two graphs separately. Instead, it is sufficient to analyze a single structure which contains the features of interest *common* in both input graphs.² For each of the definitions of the neighborhood graphs sketched above, we have defined a way of representing their *common* structures, which are used as basis for the two families of kernel functions we define: In Section 3.1 we will present kernel functions which are based on *intersection graphs* (obtained from two instance graphs), in Section 3.2 kernel functions based on *intersection trees* (on the basis of instance trees) will be presented.

3.1 Kernel Functions Based on Intersection Graphs

The intersection graph of two graphs is a graph containing all the elements the two graphs have in common.

Definition 3 (Intersection Graph). The intersection graph $G_1 \cap G_2$ of two graphs G_1 and G_2 is defined as: $V(G_1 \cap G_2) = V_1 \cap V_2$ and $E(G_1 \cap G_2) = \{(v_1, p, v_2) | (v_1, p, v_2) \in E_1 \wedge (v_1, p, v_2) \in E_2\}$.

Note that if the intersection graph contains a given subgraph, this subgraph is also a subgraph of each of the two input graphs. Inversely, if a subgraph is contained in both

¹ Note that this defines a multigraph which allows for multiple edges between the same two nodes, as long as the type of the relation is different.

² Gärtner et al. [14] have proposed kernel functions which are based on counting common structures in the *direct product graph*. In the case of graphs with unique node labels, like RDF, this is equivalent to what we call the *Intersection Graph* which we define in Section 3.1.

instance graphs, it is part of the intersection graph. Thus, calculating a kernel function based on a set of subgraphs can be reduced to constructing the intersection graph in the first step and then counting the substructures of interest therein. We have defined kernels based on implicit feature sets based on walks, paths, and (connected) subgraphs:

Edge-Induced Subgraphs. The set of edge-induced subgraphs qualifies as a candidate feature set.

Definition 4 (Edge-Induced Subgraph). *An edge-induced subgraph of $G = (V, E)$ is defined as $G' = (V', E')$ with $E' \subseteq E$ and $V' = \{v \mid \exists u, p : (u, p, v) \in E' \vee (v, p, u) \in E'\}$. We denote the edge-induced subgraph relation by $G' \subseteq G$.*

The set of edge-induced subgraphs is a valid feature set, as counting edge-induced subgraphs in the intersection graph is equivalent to performing the feature mapping explicitly and calculating the dot product of the two feature vectors.

Walks and Paths. Connected elements within the intersection graphs are likely to yield more interesting results than a set of arbitrary relations taken from the intersection graph. We have therefore defined additional kernels whose features are restricted to subsets of all edge-induced subgraphs. We have focused on *walks* and *paths* as interesting subsets, as they represent property chains in RDF.

Definition 5 (Walk, Path). *A walk in a graph $G = (V, E)$ is defined as a sequence of vertices and edges $v_1, e_1, v_2, e_2, \dots, v_n, e_n, v_{n+1}$ with $e_i = (v_i, p_i, v_{i+1}) \in E$. The length of a walk denotes the number of edges it contains.*

A path is a walk which does not contain any cycles i.e. a walk for which the additional condition $v_i \neq v_j \forall i \neq j$ holds. We denote the set of walks of length l in a graph G by $walks_l(G)$, the paths up to length l by $paths_l(G)$.

Definition 6 (Walk Kernel, Path Kernel). *The Walk Kernel for maximum path length l and discount factor $\lambda > 0$ is defined by $\kappa_{l,\lambda}(G_1, G_2) = \sum_{i=1}^l \lambda^i |\{w \mid w \in walks_i(G_1 \cap G_2)\}|$. Analog the Path Kernel $\kappa_{l,\lambda}(G_1, G_2) = \sum_{i=1}^l \lambda^i |\{p \mid p \in paths_i(G_1 \cap G_2)\}|$.*

The corresponding feature space consists of one feature per walk (resp. path). In the definition, the parameter $\lambda > 0$ serves as a discount factor and allows to weight longer walks (paths) different from shorter ones. If $\lambda > 1$ then longer walks (paths) receive more weight, in case of $\lambda < 1$ shorter ones contribute more weight.

As paths and walks are edge-induced substructures of a graph, the validity of the proposed kernel functions can be shown using the same argument as for edge-induced subgraphs. The kernel function can be calculated iteratively by constructing walks of length i as extension of walks of length $i - 1$. In each iteration an edge is appended at the end of the walks found in the previous iteration. For counting paths, the condition that $v_i \notin \{v_1, \dots, v_{i-1}\}$ has to be added when constructing the paths of length i .

A different approach for calculating these kernel functions is based on the powers of the intersection graph's adjacency matrix. The adjacency matrix M is a representation of a graph in the form of a matrix with one row and one column per node in the graph and entries $x_{ij} = 1$ if the graph contains an edge from node i to node j , 0 otherwise.

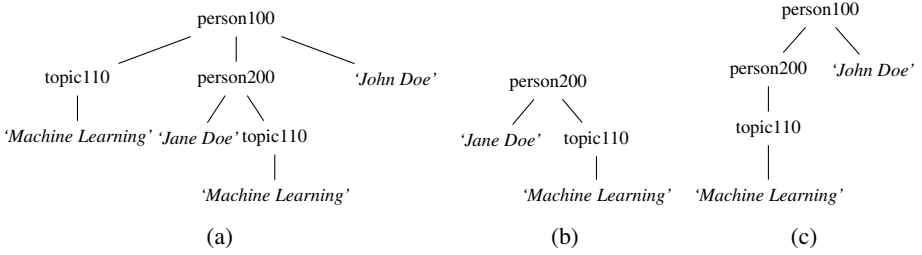


Fig. 1. (a) Example of an instance tree, (b) example of a complete subtree and (c) example of a partial subtree of the instance tree

Each entry x_{ij} of the k^{th} power of M can be interpreted as the number of walks of length k existing from node i to node j . Therefore, the number of walks up to length k in the graph can be obtained as $\sum_{i=1}^k \sum_{j=1}^n \sum_{l=1}^n (M^i)_{jl}$. By setting the elements x_{jj} of M^i to 0, this formula can also be used for the path kernel [\[3\]](#)

3.2 Kernel Functions Based on Intersection Trees

The kernel functions presented in the previous section are based on the calculation of the intersection graph. The use of the intersection graph may become problematic as its calculation is potentially expensive: the whole instance graph for each entity has to be extracted and the two graphs have to be intersected explicitly. However, the size of the instance graph grows exponentially with the number of hops crawled from the entity to build up the instance graph. Merging the two steps of extracting the instance graphs and intersecting them is not directly feasible: Consider an entity e which can be reached within k hops from both entities e_1 and e_2 , but through different paths. In this case, e would be part of the intersection graph, but it would not be reachable from either e_1 or e_2 in this graph. In the following, we present a different way of extracting common neighborhoods of two entities, which enables a direct construction of the common properties, without building the instance graphs. This alternative method is based on the use of instance trees instead of instance graphs. Instance trees are obtained based on the graph expansion with respect to an entity of interest e (as for example defined in [\[17\]](#)).

Definition 7 (Graph Expansion). *The expansion $X(e)$ of a graph G with respect to entity e is a tree defined as follows: (i) If e does not have any successors, then $X(e)$ is the tree consisting only of the node e . (ii) If v_1, \dots, v_n are the successors of e , then $X(e)$ is the tree $(e, X(v_1), \dots, X(v_n))$ (in prefix notation).*

³ Gärtner et al. [\[14\]](#) use this approach for counting walks up to infinite length by calculating the limes $k \rightarrow \infty$ of the matrix power series. They also use a weight factor to give different weight to walks of different length. However, for the matrix power series to converge, their weight factor has to be smaller than 1. Thus, in their kernel it is only possible to give smaller weight to larger structures. In the case of RDF, it may however be preferable to give more weight to larger structures as those convey more meaning.

The graph expansion can grow infinitely if the graph contains cycles. To avoid this problem and to limit the size of the obtained trees the graph expansion is bound by depth d . While in the original RDF graph, node labels were used as identifiers, i.e. each node label occurred exactly once, this is not true in the expanded graph. If there is more than one path from entity e to another entity e' , then e' will occur more than once, and thus the label of e' is not unique anymore. An intersection of the instance trees leads to an intersection tree in the same spirit as the intersection graph. We introduce two changes to the instance trees as obtained by direct expansion from the data graph:

Definition 8 (Intersection Tree). *An intersection tree $IT(G, e_1, e_2, d)$ is the tree obtained through the following steps:*

1. *Intersect the instance trees of depth d for entities e_1 and e_2 .*
2. *Replace all occurrences of labels of e_1 and e_2 by a new dummy label not occurring elsewhere in the graph.*
3. *Retain only those parts of the intersection which are connected to the root element.*

The intersection tree can be extracted directly from the data graph without constructing the instance trees explicitly. Therefore, the intersection tree can be directly obtained using Algorithm [1](#). The algorithm directly extracts the intersection tree $it_d(e_1, e_2)$ from the data graph. Starting from the two entities e_1 and e_2 the intersection tree is built using breadth-first search. Two cases have to be distinguished. In cases where one of the entities e_1 or e_2 is found a new node is added to the tree with a dummy label. For nodes with this label, the common relations of e_1 and e_2 are added as children. The second case are all nodes which do not correspond to one of the entities: for them, a new node with the node's URI respective label is added to the tree, the children of these latter nodes are all relations of this node in the data graph.

As in the case of the intersection graphs, our proposed kernel functions are based on counting elements in the intersection trees. The features of interest are restricted to features which contain the root of the tree. This is because features which are not connected to the root element may be present in each instance tree, but may by construction not be part of the intersection tree.

Full Subtrees. The first kernel function we propose based on the intersection tree is the *full subtree kernel*, which counts the number of full subtrees of the intersection tree $it_d(e_1, e_2)$, i.e. of the intersection tree of depth d for the two entities e_1 and e_2 . A full subtree of a tree t rooted at a node v is the tree with root v and all descendants of v in t (see Figure [1](#) for an example).

Definition 9 (Full Subtree Kernel). *The Full Subtree kernel is defined as the number of full subtrees in the intersection tree. Subtrees of different height are weighted differently using a discount factor λ : $\kappa_{st}(e_1, e_2) = st(\text{root}(it_d(e_1, e_2)))$ where $st(v) = 1 + \lambda \sum_{c \in \text{children}(v)} st(c)$.*

The corresponding feature mapping consists of one feature per subtree. Counting the number of full subtrees in the kernel is equivalent to counting the walks starting at the root of the intersection tree. The full subtree kernel is valid due to this equivalence.

Algorithm 1. Extraction of an intersection tree of depth d for entities e_1 and e_2

Input: entities e_1, e_2 maximum tree depth d **Data:** RDF data graph $G = (V, E)$, where labels of entities e_1 and e_2 are replaced by source**Result:** *tree*: Graph expansion $X(e_1 \cap e_2)$ of depth d

```

1 tree  $\leftarrow$  new Node("source",0)
2 newLeaves  $\leftarrow$  {tree}
3 for  $i = 1 : d - 1$  do
4   | leaves  $\leftarrow$  newLeaves;
5   | for leaf  $\in$  leaves do
6     | if leaf.label="source" then
7       |   | ce  $\leftarrow$  { $e_i | (e_1, p, e_i) \in G \wedge (e_2, p, e_i) \in G$ }
8         |   | for  $p : (e_1, p, e_2) \in G \wedge (e_2, p, e_1) \in G$  do
9           |   |   | ce.add(new Node("source"))
10        |   | else
11          |   |   | ce  $\leftarrow$  { $e_i | (leaf, p, e_i) \in G$ }
12        |   |   | for  $c \in ce$  do
13          |   |   |   | if  $c \in \{e_1, e_2\}$  then
14            |   |   |   |   | label="source"
15          |   |   |   | else
16            |   |   |   |   | label=c.uri
17          |   |   |   |   | child  $\leftarrow$  new Node(label, leaf.depth + 1)
18          |   |   |   |   | leaf.addChild(child)
19          |   |   |   |   | newLeaves.add(child)

```

Partial Subtrees. Given a tree $T = (V, E)$, its partial subtrees are defined by subsets $V' \subset V$ and $E' \subset E$ such that $T' = (V', E')$ is a tree. We propose to define a kernel function which counts the number of partial subtrees in the intersection tree $it_d(e_1, e_2)$ who are rooted at the root of $it_d(e_1, e_2)$.

Definition 10 (Partial Subtree Kernel). *The Partial Subtree Kernel is defined as the number of partial trees that the intersection tree contains. A discount factor λ gives more or less weight to trees with greater depth: $\kappa_{pt}(e_1, e_2) = t(\text{root}(it_d(e_1, e_2)))$ where t is defined as: $t(v) = \prod_{c \in \text{children}(v)} (\lambda t(c) + 1)$. The function $t(v)$ returns the number of partial subtrees with root v that the tree rooted at v contains weighted by depth with a discount factor λ .*

The corresponding feature space consists of one feature per partial tree up to depth d with root e_i replaced by a dummy node in the data graph. The value of each feature is the number of times a partial tree occurs.

4 Evaluation

To show the flexibility of the proposed kernels, we have conducted evaluations on real world data sets in two learning tasks: a property value prediction task and a link

prediction task. The kernel functions are implemented in Java using JENA⁴ for processing RDF. For the Property Value Prediction, we used *SVMlight* [18] with the JNI Kernel Extension⁵. The Link Prediction problem we used the ν -SVM implementation of LIBSVM [19].

4.1 Property Value Prediction

In a first evaluation setting, we applied our kernels to property value prediction problems. The task consists in predicting which of the possible values a property should have for a specific entity. The task thus consists in classifying the entities into the class corresponding to the predicted property value. Our approach is compared to two kernels that have been devised for general graphs: the Weisfeiler-Lehman kernel [16] and the Gaertner-kernel [14]. The Weisfeiler-Lehman kernel has been chosen as a state-of-the-art graph kernel which can be computed very efficiently. It is based on counting common subtree-patterns in the input graphs. The Gaertner kernel is closely related to the kernel function we present here: it is based on the analysis of the direct product graph which is equivalent to the intersection graph in the case of RDF data. Additionally, the kernel functions presented by Bloehdorn and Sure [6] are used for comparison: these are kernel functions manually defined for a specific dataset.

Data Sets. The first evaluation set uses data from the SWRC ontology [20] and the metadata available in the Semantic Portal of the institute AIFB. The ontology models key concepts within a research community. The evaluation data consists of 2,547 entities of which 1,058 belong to the person class. 178 of these persons are affiliated with one of the research groups at AIFB. There are 1232 instances of type publication, 146 of type research topic and 146 of type project. The entities are connected by a total of 15,883 relations. Additionally, there are 8,705 datatype properties, i.e. properties linking an entity to a literal. The evaluation setting defined in [6] consists in classifying staff members with respect to their affiliation to research groups. All relations denoting the affiliation of a person with a research group were deleted from the training data.

In a second setting, a larger dataset consisting of Friend of a Friend (FOAF) descriptions of people from the community website LiveJournal.com was used for experiments. The dataset describes 22745 people and their social networks, as well as some personal information (location, school they attended, online accounts, their birthdays and the number of posts on their blog). Note that birth dates and number of blog posts are reduced to a small number of discrete states. For example, the precise age was replaced by one of four newly introduced age classes. For the evaluation, we removed all relations of type *dateOfBirth* from the dataset and tried to learn this relation afterwards, i.e. the goal was to predict for all 1567 people with available age information to which of the 4 age classes they should belong.

Compared Approaches. All results reported in Table 1 were obtained using leave-one-out cross-validation. The trade-off parameter c of the Support Vector Machine learning was set to 1 in all experiments. We compare the results obtained using the kernel functions defined in Sect. 3 to the Weisfeiler-Lehman kernel and the Gärtner kernel. We

⁴ <http://jena.sourceforge.net>

⁵ <http://people.aifb.kit.edu/sbl/software/jnikernel/>

Table 1. Evaluation results for Property Value Prediction

Kernel configuration				SWRC results		LiveJournal results	
Kernel	Instance Depth	Max. Size	Discount	Error	F1 measure	Error	F1 measure
Bloehdorn/Sure [6]				0.0449	0.7237	–	–
Weisfeiler-Lehman [16]	2	2		0.1096	0.0000	0.2215	0.4084
Gaertner [14]	2		0.5	0.0590	0.6625	0.3824	0.5344
Paths/Walks	2	1	1	0.0576	0.6318	0.2125	0.4096
Paths	2	2	1	0.0576	0.6318	0.1946	0.4192
Walks	2	2	1	0.0576	0.6318	0.1948	0.4175
Full Subtree	2		1	0.0548	0.6543	0.1902	0.4018
Partial Subtree	2		0.01	0.1025	0.3247	0.1866	0.3286
Edges (no schema)	2			0.0478	0.7488	–	–
Walks (no schema)	2	2	1	0.0478	0.7488	–	–
Paths (no schema)	2	2	1	0.0478	0.7488	–	–
Full Subtree	2		1	0.0548	0.6687	–	–

applied these graph kernels to the instance graphs of depth 2 which were extracted from the RDF data graph. We chose the parameters of the compared approaches according to the best setting reported in the literature: The maximum depth of trees in the Weisfeiler-Lehman kernel was set to 2, the discount factor for longer walks in the Gartner kernel was set to 0.5. On the SWRC dataset, the best configuration obtained by Bloehdorn and Sure [6] in the original paper was used for comparison. This kernel configuration, denoted by *sim-ctpp-pc* combines the common class similarity kernel described in their paper with object property kernels for the *workedOnBy*, *worksAtProject* and *publication*. Additionally, also on the SWRC dataset, we compared the performance of our kernels on the whole data set (including the schema) to a setting where all relations which are part of the schema were removed (lowest part in Table 1).

Discussion. Our results show that with specific parameters our kernels reach comparable error and higher F1-measure than the kernels proposed by Bloehdorn and Sure. Considering that our approach is generic and can be used off the shelf in many scenarios, while their kernel function was designed manually for the specific application scenario, this is a positive result. Our kernel functions also perform well with respect to other graph kernels: The Weisfeiler-Lehman kernel is not able to separate the training data in the SWRC dataset as it can match only very small structures. While the Gaertner kernel achieves results which are comparable to our results, its calculation is more expensive - due to the cubic time complexity of the matrix inversion. Our results show that the walk kernel and the path kernel perform better in terms of classification errors when the maximum size of the substructures are increased. As for the partial subtree kernel it turned out that parameter tuning is difficult and that the results strongly depend on the choice of the discount factor. Last but not least, a surprising result is that the kernels which are based on the intersection graph perform better on the reduced data set which does not contain the schema information. Our explanation for this is that

the intersection graph contains part of the schema and thus produces a similar overlap for many of the instances. Further results which we do not report here show that with increasing maximum size k of the considered substructures precision is increased and recall is decreased. This effect may be due to the effect that bigger structures describe instances in a more precise way, thus refining the description of elements belonging to the class but losing generality of the class description on the other hand.

4.2 Link Prediction

In a second scenario we evaluated our kernel functions on link prediction problems. We formalized the link prediction as a binary classification problem where the goal is to predict whether a link exists between two entities. The classifier is learned for a specific relation. In this problem, we require a kernel function for the comparison of relation instances. We propose a link kernel which is based on the comparison of entities:

$$\kappa((s_1, o_1), (s_2, o_2)) = \alpha\kappa_s(s_1, s_2) + \beta\kappa_o(o_1, o_2)$$

This is a valid kernel function for $\alpha, \beta \geq 0$ as the space of valid kernel function is closed under sum and multiplication with a positive scalar [11]. Any kernel functions for RDF entities may be used as subject and object kernel (κ_s and κ_o).

Datasets. We evaluated the link prediction approach on two datasets. The first evaluation setting uses the SWRC dataset as for property value description. In this scenario the goal is to predict the `affiliation` relation, i.e. to decide for a tuple of person and research group whether the given person works in the given research group. The second dataset is a reduced version of the LiveJournal dataset used for age prediction. The reduced dataset contains descriptions of 638 people and their social network. Overall, there are 8069 instances of the `foaf:knows` relation that is learned in this setting.

Evaluation Method. RDF is based on the open-world assumption, which means that if a fact is not stated, it may not be assumed to not hold. Rather, the truth value of this relation is unknown. This is problematic as no negative training and test data is available. As to the training phase, some initial experiments using one-class SVM [21] did not yield promising results. We therefore considered some negative training data to be necessary for the evaluation. Therefore, the positive training instances were complemented with an equal number of unknown instances. We consider this number of negative instances a good trade-off between the number of assumptions made and the gain in quality of the trained classifier. The obtained training set is a balanced dataset which does not present any bias towards one of the classes.

Evaluation may not be based on positive or negative classifications due to the absence of negative data. Thus, a ranking-based approach to the evaluation was chosen: positive instances should be ranked higher in the result set than negative ones. For evaluation, a modified version of 5-fold cross validation was used: the positive instances were split in five folds, each fold was complemented with an equal number of unknown instances and the unused unknown instances were used as additional test data. Each model was trained on four folds and evaluated on the fifth fold and the additional test data. NDCG [22] and bpref [23] were used as evaluation measures.

Table 2. Evaluation Results for Link Prediction

Kernel configuration				Results for SWRC		Results for LiveJournal	
Kernel	Inst. depth	α/β	Param. SUNS	NDCG	bpref	NDCG	bpref
SUNS-SVD			20	0.8022	0.3844	0.7871	0.2889
SUNS-Regularized			1	0.4038	0.0538	0.5446	0.0125
Weisfeiler-Lehman	2	1		0.9443	0.8303	0.9991	0.9963
Common Subtrees	2	0.5		0.9316	0.8363	0.9724	0.9056
Common Subtrees	2	1		0.9340	0.8438	0.9886	0.9680
Common Subtrees	2	2		0.9548	0.8538	0.9694	0.8948
Common Subtrees	2	3		0.9271	0.7831	0.9741	0.9006
Common Subtrees	2	5		0.8387	0.6313	0.9744	0.8945

Compared Approaches. We have compared our approach to the SUNS approach presented by Huang et al. [8] and to the link kernel with the Weisfeiler-Lehman kernel [16] as entity kernel. This approach is based on a singular value decomposition of the relation matrix and generates predictions based on the completed matrix. We experimented with different settings for the parameter of SUNS, where the range of the parameter was taken from Huang et al. [8]. However, our evaluation procedure is substantially different to theirs. We report the best results which we obtained. Results are reported in Table 2. The settings of the Weisfeiler-Lehman kernel are the same as in the property value prediction task. For all SVM-based evaluations presented here, the parameter ν was set to 0.5.

Discussion. Our results show that our Link Prediction approach can outperform the SUNS approach in terms of NDCG and bpref. Partly, this may be due to the chosen evaluation procedure: the SUNS approach can only deal with those relation instances which were present in the training phase: as only 80% of the available data were used for training, some instances of the domain or range may not occur in the training data and thus no prediction is obtained for those. To obtain a complete ranking, a minimum value was assigned to these instances.⁶ Additionally, a pruning step is part of the preprocessing steps of SUNS, which removes elements with few connections. In comparison with the Weisfeiler-Lehman kernel our kernel functions achieve comparable results on the SWRC dataset. As to the relation of α and β in the link kernel, best results are obtained for $\frac{\alpha}{\beta} = 2$ in the SWRC dataset and for $\frac{\alpha}{\beta} = 1$ in the LiveJournal dataset. We suppose that the higher importance of the subject in the SWRC dataset is due to the smaller number of objects in the range of the relation. In contrast, in the LiveJournal dataset there is an equal number of elements in the domain and the range of the relation. Another finding from additional experiments which we do not report here is that the quality of the model increases with growing ν . This means that a complexer model achieves better generalisation results on both datasets.

⁶ This explains the very low bpref achieved by the SUNS approach: all positive instances for which no prediction could be obtained are ranked lower than all negative instances.

5 Conclusion

With the advent of the Resource Description Framework (RDF) and its broad uptake, e.g. within the Linked Open Data (LOD) initiative, the problem of mining from semantic data has become an important issue. In this paper, we have introduced a principle approach for exploiting RDF graph structures within established machine learning algorithms by designing suitable kernel functions which exploit the specific properties of RDF data while remaining general enough to be applicable to a wide range of learning algorithms and tasks. We have introduced two versatile families of kernel functions for RDF entities based on intersection graphs and intersection trees and have shown that they have an intuitive, powerful interpretation while remaining computationally efficient. In an empirical evaluation, we demonstrated the flexibility of this approach and show that kernel functions within this family can compete with hand-crafted kernel functions and computationally more demanding approaches.

In the future, we plan to extend this framework to other substructures that can be shown to be present in intersection graphs resp. intersection trees and to apply it to new tasks and other Linked Open Data (LOD) data sets.

Acknowledgments. The work was funded by the EU project XLike under FP7.

References

- [1] Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. *Int. Journal on Semantic Web and Information Systems* 5(3), 1–22 (2009)
- [2] Augenstein, I., Padó, S., Rudolph, S.: Lodifier: Generating Linked Data from Unstructured Text. In: Simperl, E., et al. (eds.) *ESWC 2012*. LNCS, vol. 7295, pp. 210–224. Springer, Heidelberg (2012)
- [3] Brickley, D., Miller, L.: FOAF vocabulary specification. Technical report, FOAF project (2007), <http://xmlns.com/foaf/spec/20070524.html> (Published online on May 24, 2007)
- [4] Fanizzi, N., d’Amato, C.: A Declarative Kernel for *ALC* Concept Descriptions. In: Esposito, F., Raś, Z.W., Malerba, D., Semeraro, G. (eds.) *ISMIS 2006*. LNCS (LNAI), vol. 4203, pp. 322–331. Springer, Heidelberg (2006)
- [5] Fanizzi, N., d’Amato, C., Esposito, F.: Statistical Learning for Inductive Query Answering on OWL Ontologies. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) *ISWC 2008*. LNCS, vol. 5318, pp. 195–212. Springer, Heidelberg (2008)
- [6] Bloehdorn, S., Sure, Y.: Kernel Methods for Mining Instance Data in 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.) *ASWC 2007 and ISWC 2007*. LNCS, vol. 4825, pp. 58–71. Springer, Heidelberg (2007)
- [7] Rettinger, A., Nickles, M., Tresp, V.: Statistical Relational Learning with Formal Ontologies. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) *ECML PKDD 2009*. LNCS, vol. 5782, pp. 286–301. Springer, Heidelberg (2009)
- [8] Huang, Y., Tresp, V., Bundschuh, M., Rettinger, A., Kriegel, H.-P.: Multivariate Prediction for Learning on the Semantic Web. In: Frasconi, P., Lisi, F.A. (eds.) *ILP 2010*. LNCS, vol. 6489, pp. 92–104. Springer, Heidelberg (2011)

- [9] Bicer, V., Tran, T., Gossen, A.: Relational Kernel Machines for Learning from Graph-Structured RDF Data. 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. 47–62. Springer, Heidelberg (2011)
- [10] Thor, A., Anderson, P., Raschid, L., Navlakha, S., Saha, B., Khuller, S., Zhang, X.-N.: Link Prediction for Annotation Graphs Using Graph 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. 714–729. Springer, Heidelberg (2011)
- [11] Shawe-Taylor, J., Cristianini, N.: *Kernel Methods for Pattern Analysis*. Cambridge University Press (2004)
- [12] Getoor, L., Friedman, N., Koller, D., Pferrer, A., Taskar, B.: Probabilistic relational models. In: *Introduction to Statistical Relational Learning*. MIT Press (2007)
- [13] Haussler, D.: Convolution kernels on discrete structures. Technical Report UCS-CRL-99-10, University of California at Santa Cruz (1999)
- [14] Gärtner, T., Flach, P., Wrobel, S.: On graph kernels: Hardness results and efficient alternatives. In: Schölkopf, B., Warmuth, M.K. (eds.) *COLT/Kernel 2003*. LNCS (LNAI), vol. 2777, pp. 129–143. Springer, Heidelberg (2003)
- [15] Horváth, T., Gärtner, T., Wrobel, S.: Cyclic pattern kernels for predictive graph mining. In: *Proc. of the 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2004)*, pp. 158–167. ACM Press, New York (2004)
- [16] Shervashidze, N., Borgwardt, K.: Fast subtree kernels on graphs. In: *Advances in Neural Information Processing Systems*, vol. 22 (2009)
- [17] Güting, R.H.: *Datenstrukturen und Algorithmen*. B.G. Teubner, Stuttgart (1992)
- [18] Joachims, T.: Making large-scale SVM learning practical. In: *Advances in Kernel Methods - Support Vector Learning* (1999)
- [19] Chang, C.C., Lin, C.-J.: LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2, 27:1–27:27 (2011), Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
- [20] Sure, Y., Bloehdorn, S., Haase, P., Hartmann, J., Oberle, D.: The SWRC Ontology – Semantic Web for Research Communities. In: Bento, C., Cardoso, A., Dias, G. (eds.) *EPIA 2005*. LNCS (LNAI), vol. 3808, pp. 218–231. Springer, Heidelberg (2005)
- [21] Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Technical report (1999)
- [22] Järvelin, K., Kekäläinen, J.: IR evaluation methods for retrieving highly relevant documents. In: *Proc. of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 41–48. ACM (2000)
- [23] Buckley, C., Voorhees, E.M.: Retrieval evaluation with incomplete information. In: *Proc. of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2004)*, pp. 25–32. ACM (2004)

EAGLE: Efficient Active Learning of Link Specifications Using Genetic Programming

Axel-Cyrille Ngonga Ngomo and Klaus Lyko

Department of Computer Science
University of Leipzig Johannisgasse 26, 04103 Leipzig
ngonga@informatik.uni-leipzig.de, lyko.klaus@gmail.com
<http://limes.sf.net>

Abstract. With the growth of the Linked Data Web, time-efficient approaches for computing links between data sources have become indispensable. Most Link Discovery frameworks implement approaches that require two main computational steps. First, a link specification has to be explicated by the user. Then, this specification must be executed. While several approaches for the time-efficient execution of link specifications have been developed over the last few years, the discovery of accurate link specifications remains a tedious problem. In this paper, we present EAGLE, an active learning approach based on genetic programming. EAGLE generates highly accurate link specifications while reducing the annotation burden for the user. We evaluate EAGLE against batch learning on three different data sets and show that our algorithm can detect specifications with an F-measure superior to 90% while requiring a small number of questions.

1 Introduction

The growth of the Linked Data Web over the last years has led to a compendium of currently more than 30 billion triples [3]. Yet, it still contains a relatively low number of links between knowledge bases (less than 2% at the moment). Devising approaches that address this problem still remains a very demanding task. This is mainly due to the difficulty behind Link Discovery being twofold: First, the quadratic complexity of Link Discovery requires time-efficient approaches that can efficiently compute links when given a specification of the conditions under which a link is to be built [14,21] (i.e., when given a so-called *link specification*). Such specifications can be of arbitrary complexity, ranging from a simple comparison of labels (e.g., for finding links between countries) to the comparison of a large set of features of different types (e.g., using population, elevation and labels to link villages across the globe). In previous work, we have addressed this task by developing the LIMES¹ framework. LIMES provides time-efficient approaches for Link Discovery and has been shown to outperform other frameworks significantly [20].

¹ <http://limes.sf.net>

The second difficulty behind Link Discovery lies in the detection of accurate link specifications. Most state-of-the-art Link Discovery frameworks such as LIMES and SILK [14] adopt a property-based computation of links between entities. To ensure that links can be computed with a high accuracy, these frameworks provide (a) a large number of similarity measures (i.e., Levenshtein, Jaccard for strings) for comparing property values and (b) manifold means for combining the results of these measures to an overall similarity value for a given pair of entities. When faced with this overwhelming space of possible combinations, users often adapt a time-demanding trial-and-error approach to detect an accurate link specification for the task at hand. There is consequently a blatant need for approaches that support the user in the endeavor of finding accurate link specifications. From a user’s perspective, approaches for the semi-automatic generation of link specification must support the user by

1. reducing the time frame needed to detect a link specification (time efficiency),
2. generating link specifications that generate a small number of false positives and negatives (accuracy) and
3. providing the user with easily readable and modifiable specifications (readability).

In this paper, we present the EAGLE algorithm, a supervised machine-learning algorithm for the detection of link specifications that abides by the three criteria presented above. One of the main drawbacks of machine-learning approaches is that they usually require a large amount of training data to achieve a high accuracy. Yet, the generation of training data can be a very tedious process. EAGLE surmounts this problem by implementing an active learning approach [27]. Active learning allows the interactive annotation of highly informative training data. Therewith, active learning approaches can minimize the amount of training data necessary to compute highly accurate link specifications.

Overall, the contributions of this paper are as follows:

- We present a novel active learning approach to learning link specifications based on genetic programming.
- We evaluate our approach on three different data sets and show that we reach F-measures of above 90% by asking between 10 and 20 questions even on difficult data sets.
- We compare our approach with state-of-the-art approaches on the DBLP-ACM dataset and show that we outperform them with respect to runtime while reaching a comparable accuracy.

The advantages of our approach are manifold. In addition to its high accuracy, it generates readable link specifications which can be altered by the user at will. Furthermore, given the superior runtime of LIMES on string and numeric properties, our approach fulfills the requirements for use in an interactive setting. Finally, our approach only requires very small human effort to discover link specifications of high accuracy as shown by our evaluation.

The rest of this paper is organized as follows: First, we give a brief overview of the state of the art. Thereafter, we present the formal framework within which

EAGLE is defined. This framework is the basis for the subsequent specification of our approach. We then evaluate our approach with several parameters on three different data sets. We demonstrate the accuracy of our approach by computing its F-measure. Moreover, we show that EAGLE is time-efficient by comparing its runtime with that of other approaches on the ACM-DBLP dataset. We also compare our approach with its non-active counterpart and study when the use of active learning leads to better results.

2 Related Work

Over the last years, several approaches have been developed to address the time complexity of link discovery. Some of these approaches focus on particular domains of applications. For example, the approach implemented in RKB knowledge base (RKB-CRS) [11] focuses on computing links between universities and conferences while GNAT [25] discovers links between music data sets. Further simple or domain-specific approaches can be found in [9,23,13,28,24]. In addition, domain-independent approaches have been developed, that aim to facilitate link discovery all across the Web. For example, RDF-AI [26] implements a five-step approach that comprises the preprocessing, matching, fusion, interlinking and post-processing of data sets. SILK [14] is a time-optimized tool for link discovery. It implements a multi-dimensional blocking approach that is guaranteed to be lossless thanks to the overlapping blocks it generates. Another lossless Link Discovery framework is LIMES [20], which addresses the scalability problem by implementing time-efficient similarity computation approaches for different data types and combining those using set theory. Note that the task of discovering links between knowledge bases is closely related with record linkage [30,10,5,17]. To the best of our knowledge, the problem of discovering accurate link specifications has only been addressed in very recent literature by a small number of approaches: The SILK framework [14] now implements a batch learning approach to discovery link specifications based on genetic programming which is similar to the approach presented in [6]. The algorithm implemented by SILK also treats link specifications as trees but relies on a large amount of annotated data to discover high-accuracy link specifications. The RAVEN algorithm [22] on the other hand is an active learning approach that treats the discovery of specifications as a classification problem. It discovers link specifications by first finding class and property mappings between knowledge bases automatically. RAVEN then uses these mappings to compute linear and boolean classifiers that can be used as link specifications. A related approach that aims to detect discriminative properties for linking is that presented by [29]. In addition to these approaches, several machine-learning approaches have been developed to learn classifiers for record linkage. For example, machine-learning frameworks such as FEBRL [7] and MARLIN [4] rely on models such as Support Vector Machines [16,8], decision trees [31] and rule mining [1] to detect classifiers for record linkage. Our approach, EAGLE, goes beyond previous work in three main ways. First, it is an active learning approach. Thus, it does not require the large amount of training data required by batch learning approaches such as FEBRL, MARLIN and

SILK. Furthermore, it allows to use the full spectrum of operations implemented in LIMES. Thus, it is not limited to linear and boolean classifiers such as those generated by FeBRL and RAVEN. Finally, it can detect property and class mappings automatically. Thus, it does not need to be seeded to converge efficiently like previous approaches [14].

3 Preliminaries

In the following, we present the core of the formalization and notation necessary to implement EAGLE. We first formalize the Link Discovery problem. Then, we give an overview of the grammar that underlies links specifications in LIMES and show how the resulting specifications can be represented as trees. We show how the discovery of link specifications can consequently be modeled as a genetic programming problem. Subsequently, we give some insight in active learning and then present the active learning model that underlies our work.

3.1 Link Discovery

The link discovery problem, which is similar to the record linkage problem, is an ill-defined problem and is consequently difficult to model formally [2]. In general, link discovery aims to discover pairs $(s, t) \in S \times T$ related via a relation R .

Definition 1 (Link Discovery). *Given two sets S (source) and T (target) of entities, compute the set \mathcal{M} of pairs of instances $(s, t) \in S \times T$ such that $R(s, t)$.*

The sets S resp. T are usually (not necessarily disjoint) subsets of the instances contained in two (not necessarily disjoint) knowledge bases \mathcal{K}_S resp. \mathcal{K}_T . One way to automate this discovery is to compare the $s \in S$ and $t \in T$ based on their properties using a (complex) similarity measure σ . Two entities $s \in S$ and $t \in T$ are then considered to be linked via R if $\sigma(s, t) \geq \theta$ [21]. The specification of the sets S and T and of this similarity condition is usually carried out within a *link specification*.

Definition 2 (Link Specification). *A link specification consists of three parts: (1) two sets of restrictions $\mathcal{R}_1^S \dots \mathcal{R}_m^S$ resp. $\mathcal{R}_1^T \dots \mathcal{R}_k^T$ that specify the sets S resp. T , (2) a specification of a complex similarity metric σ via the combination of several atomic similarity measures $\sigma_1, \dots, \sigma_n$ and (3) a set of thresholds $\theta_1, \dots, \theta_n$ such that θ_i is the threshold for σ_i .*

A restriction \mathcal{R} is generally a logical predicate. Typically, restrictions in link specifications state the `rdf:type` of the elements of the set they describe, i.e., $\mathcal{R}(x) \leftrightarrow x \text{ rdf:type someClass}$ or the features that the elements of the set must have, e.g., $\mathcal{R}(x) \leftrightarrow (\exists y : x \text{ someProperty } y)$. Each $s \in S$ must abide by each of the restrictions $\mathcal{R}_1^S \dots \mathcal{R}_m^S$, while each $t \in T$ must abide by each of the restrictions $\mathcal{R}_1^T \dots \mathcal{R}_k^T$. Each similarity σ_i is used to compare pairs of property values of instances s and t . EAGLE relies on the approach presented in [20] to detect the class and property mappings. Consequently, in the remainder of this paper, the term *link specification* will be used to denote the complex similarity condition used to determine whether two entities should be linked.

3.2 Link Specifications as Trees

Our definition of a link specification relies on the definition of *atomic similarity measures* and *similarity measures*. Generally, a similarity measure m is a function such that $m : S \times T \rightarrow [0, 1]$. We call a measure atomic (dubbed *atomicMeasure*) when it relies on exactly one similarity measure σ (e.g., trigrams similarity for strings) to compute the similarity of a pair $(s, t) \in S \times T$. A similarity measure m is either an atomic similarity measure *atomicMeasure* or the combination of two similarity measures via a metric operator *metricOp* such as *MAX*, *MIN* and linear combinations *ADD*.

1. $m \rightsquigarrow \text{atomicMeasure}$
2. $m \rightsquigarrow \text{metricOp}(m_1, m_2)$

We call a link specification atomic (*atomicSpec*) if it compares the value of a measure m with a threshold θ , thus returning the pairs (s, t) that satisfy the condition $\sigma(s, t) \geq \theta$. A link specification $\text{spec}(m, \theta)$ is either an atomic link specification or the combination of two link specifications via specification operators *specOp* such as *AND* (intersection of the set of results of two specs), *OR* (union of the result sets), *XOR* (symmetric set difference), or *DIFF* (set difference). Thus, the following grammar for specifications holds :

1. $\text{spec}(m, \theta) \rightsquigarrow \text{atomicSpec}(m, \theta)$
2. $\text{spec}(m, \theta) \rightsquigarrow \text{specOp}(\text{spec}(m_1, \theta_1), \text{spec}(m_2, \theta_2))$

Each link specification that abides by the grammar specified above can be consistently transformed into a tree that contains the central constructs shown in Figures [1](#), [2](#), [3](#) and [4](#).

4 Approach

As we have formalized link specifications as trees, we can use Genetic Programming (GP) to solve the problem of finding the most appropriate complex link specification for a given pair of knowledge bases. Given a problem, the basic idea behind genetic programming [18](#) is to generate increasingly better solutions of the given problem by applying a number of genetic operators to the current population. In the following, we will denote the population at time t by g^t . Genetic operators simulate natural selection mechanisms such as mutation and reproduction to enable the creation of individuals that best abide by a given fitness function. One of the key problems of genetic programming is that it is a non-deterministic procedure. In addition, it usually requires a large training data set to detect accurate solutions. In this paper, we propose the combination of GP and active learning [27](#). Our intuition is that by merging these approaches, we can infuse some determinism in the GP procedure by allowing it to select the most informative data for the population. Thus, we can improve the convergence of GP approaches while reducing the labeling effort necessary to use them. In the following, we present our implementation of the different GP operators on link specifications and how we combine GP and active learning.



Fig. 1. Atomic measure

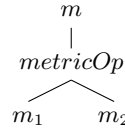


Fig. 2. Complex measure



Fig. 3. Atomic specification

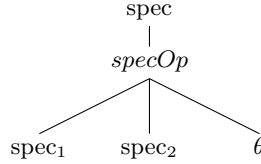


Fig. 4. Complex specification

4.1 Overview

Algorithm 1 gives an overview of the approach implemented by EAGLE. After the detection of matching classes and properties using the approach explicated in [22], we begin by generating a random population of individual link specifications. To evolve a population to the next one a number of steps is required: First, all existing individuals must be assigned a fitness score. This score reflects how well a link specification performs on the training data at hand. Subsequently, the genetic operators reproduction, mutation and crossover are applied to the individuals of the current population in order to let the individuals adapt to the problem. The *reproduction* determines which individual is carried into the next generation. Note that throughout this paper, we use a tournament setting of two randomly chosen individuals. The *mutation* operator can be applied to a single individual. It alters this individual by randomly changing parts of its tree (i.e., of his genome) with the aim of creating a new individual. The *crossover* operator also aims at generating new individuals. It achieves this goal by crossing over branches of the program trees of two parent individuals.

Algorithm 1. EAGLE

Require: Specification of the two knowledge bases KS and KT
 Get set S and set T of instances as specified in KS respectively KT .
 Get property mapping (KS, KT)
 Get reference mapping by asking user to label n random pairs $(s, t) \in S \times T$
repeat
 Evolve population(*population, size*) *generations* times.
 Compute n most informative link candidates and ask user to label them.
until stop condition reached

Algorithm 2. Evolves a population

```

if population is empty then
  create size random individuals
end if
Compute fitness of population
Apply genetic operators to population
return population

```

In the following, we will explicate each of the steps of our algorithm in more detail. Each of these steps will be exemplified by using the link specification shown in Figure 5.

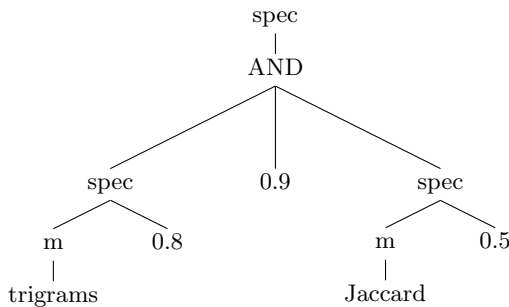


Fig. 5. Exemplary link specification

4.2 Evolution of Population

Evolution is the primary process which enables GP to solve problems and drives the development of efficient solutions for a given problem. At the beginning of our computation the population is empty and must be built by individuals generated randomly. This is carried out by generating random trees whose nodes are filled with functions or terminals as required. For this paper, we defined the operators (functions and terminals) in the genotype for the problem to generate link specifications as follows: all *metricOp* and *specOp* were set to be functions. Terminal symbols were thresholds and measures. Note that these operators can be extended at will. In addition, all operators were mapped to certain constraints so as to ensure that EAGLE only generates valid program trees. For example, the operator that compares numeric properties only accepts terminals representing numeric properties from the knowledge bases.

Let g^t be the population at the iteration t . To evolve a population to the generation g^{t+1} we first determine the fitness of all individuals of generation g^t (see Section 4.3). These fitness values build the basis for selecting individuals for the genetic operator reproduction. We use a tournament setting between two selected individuals to decide which one is copied to the next generation g^{t+1} .

On randomly selected individuals the operator mutation is applied according with a probability called the *mutation rate*. The mutation operator changes single nodes in the program tree of the individuals. A mutation can affect an individual in three different ways: First, it can alter the thresholds used by the individual. Second, a mutation can alter the properties contained in the individual’s genome. Finally, mutations can modify the measures included in the individuals (see Figure 6). The third genetic operator, *crossover*, operates on two parent individuals and builds a new offspring by swapping two random subtrees of the parent genotypes. Figure 7 exemplifies the functionality of the crossover operator.

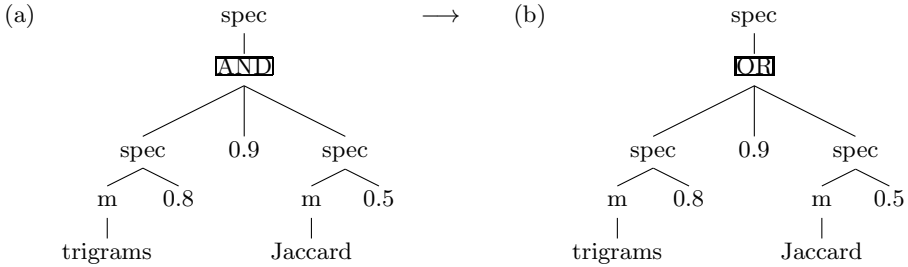


Fig. 6. Mutation example. Mutation changes boolean operator.

The individuals selected to build the population of g^{t+1} are the n fittest from the union of the set of newly created individuals and g^t . Note that we iteratively generate new populations of potential fitter individuals.

4.3 Fitness

The aim of the fitness function is to approximate how well a solution (i.e., a link specification) solves the problem at hand. In the supervised machine learning setting, this is equivalent to computing how well a link specification maps the training data at hand. To determine the fitness of an individual we first build the link specification that is described by the tree at hand. Given the set of available training data $\mathcal{O} = \{(x_i, y_i) \in S \times T\}$, we then run the specification by using the sets $S(\mathcal{O}) = \{s \in S : \exists t \in T : (s, t) \in \mathcal{O}\}$ and $T(\mathcal{O}) = \{t \in T : \exists s \in S : (s, t) \in \mathcal{O}\}$. The result of this process is a mapping \mathcal{M} that is then evaluated against \mathcal{O} by the means of the standard F-measure defined as

$$\frac{2PR}{P + R} \text{ where } P = \frac{|\mathcal{M} \cap \mathcal{O}|}{|\mathcal{M}|} \text{ and } R = \frac{|\mathcal{M} \cap \mathcal{O}|}{|\mathcal{O}|}. \tag{1}$$

Note that by running the linking on $S(\mathcal{O})$ and $T(\mathcal{O})$, we can significantly reduce EAGLE’s runtime.

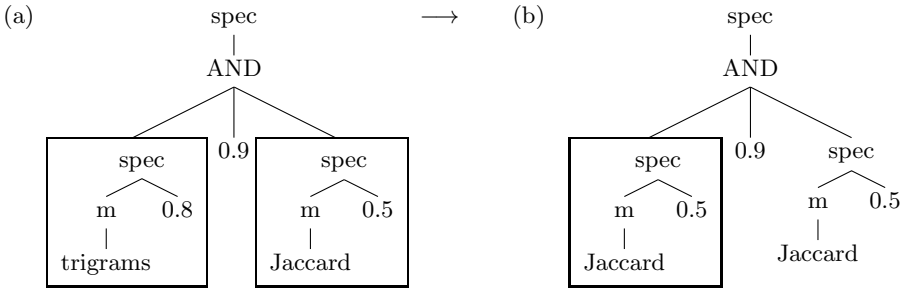


Fig. 7. Crossover example. Consider we have two individuals with a program tree like in (a). A crossover operation can replace subtrees to produce an offspring like (b).

4.4 Computation of Most Informative Link Candidates

The main idea behind the reduced of the amount of labeling effort required by active learning approaches is that they only required highly informative training data from the user. Finding these most informative pieces of information is usually carried out by measuring the amount of information that the labeling of a training data item would bear. Given the setting of EAGLE in which several possible solutions co-exist, we opted for applying the idea of active learning by committees as explicated in [19]. The idea here is to consistently entertain a finite and incomplete set of solutions to the problem at hand. The most informative link candidates are then considered to be the pairs $(s, t) \in S \times T$ upon which the different solutions disagree the most. In our case, these are the link candidates that maximize the disagreement function $\delta((s, t))$:

$$\delta((s, t)) = (n - |\{\mathcal{M}_i^t : (s, t) \in \mathcal{M}_i\}|)(n - |\{\mathcal{M}_i^t : (s, t) \notin \mathcal{M}_i\}|), \quad (2)$$

where \mathcal{M}_i are the mappings generated by the population g^t . The pairs (s, t) that lead to the highest disagreement score are presented to the user, who provides the system with the correct labels. This training set is finally updated and used to compute the next generations of solutions.

5 Evaluation

5.1 Experimental Setup

We evaluated our approach in three experiments. In our experiments, our main goal not only to show that we can discover link specifications of different complexity with high accuracy. In addition, we also aimed to study the effect of the population size and of active learning on the quality of link specifications. For this purpose, we devised three experiments whose characteristics are shown in Table 1.

The goal of the first experiment, called Drugs, was to measure how well we can detect a manually created LIMES specification. For this purpose, we generated owl:sameAs link candidates between Drugs in DailyMed and Drugbank by using their rdfs:label. The second experiment, Movies, was carried out by using the results of a LATC² link specification. Here, we fetched the links generated by a link specification that linked movies in DBpedia to movies in LinkedMDB [12], gathered the rdfs:label of the movies as well as the rdfs:label of their directors in the source and target knowledge bases and computed a specification that aimed to reproduce the set of links at hand as exactly as possible. Note that this specification is hard to reproduce as the experts who created this link specification applied several transformations to the property values before carrying out the similarity computation that led to the results at hand. Finally, in our third experiment (Publications), we used the ACM-DBLP dataset described in [17]. Our aim here was to compare our approach with other approaches with respect to both runtime and F-measure.

Table 1. Characteristics of the datasets used for the evaluation of EAGLE. *S* stands for source, *T* for target.

Label	<i>S</i>	<i>T</i>	$ S \times T $	Oracle size
Drugs	Dailymed	Drugbank	1.09×10^6	1046
Movies	DBpedia	LinkedMDB	1.12×10^6	1056
Publications	ACM	DBLP	6.01×10^6	2224

All experiments were carried out on one kernel of an AMD Opteron Quad-Core processor (2GHz) with the followings settings: the population size was set to 20 or 100. The maximal number of generations was set to 50. In all active learning experiments, we carried out 10 inquiries per iteration cycle. In addition, we had the population evolve for 10 generations between all inquiries. The mutation and crossover rates were set to 0.6. For the batch learners, we set the number of generations to the size of the training data. Note that this setup is of disadvantage for active learning as the batch learners then have more data and more iterations on the data to learn the best possible specification. We used this setting as complementary for the questions that can be asked by the active learning approach. During our experiments, the Java Virtual Machine was allocated 1GB RAM. All experiments were repeated 5 times.

5.2 Results

The results of our experiments are shown in the Figures below³. In all figures, Batch stands for the batch learners while AL stands for the active learners. The

² <http://lact-project.eu>

³ Extensive results are available at the LIMES project website at <http://limes.sf.net>

numbers in brackets are the sizes of the populations used. The results of the Drugs experiments clearly show that our approach can easily detect simple link specifications. In this experiment, 10 questions were sufficient for the batch and active learning versions of EAGLE to generate link specifications with an F-measure equivalent to the baseline of 99.9% F-measure. The standard deviation lied around 0.1% for all experiments with both batch and active learner.

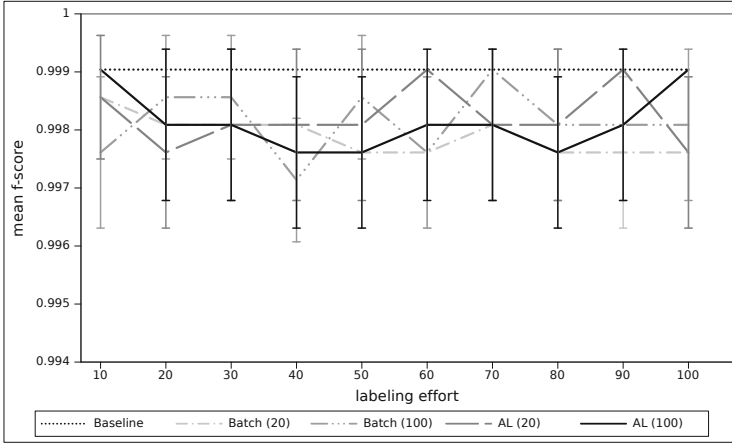


Fig. 8. Results of the Drugs experiment. Mean F-Measure of five runs of batch and active learner, both using population sizes of 20 and 100 individuals. The baseline is at 99.9% F-measure.

On the more complex Movies experiments, we required 50 inquiries to discover the best link specification that led to an F-measure of 94.1%. This experiment clearly shows the effect of active learning on genetic programming algorithms. While the batch learners fed with any data size tend to diverge significantly across the different experiments as shown by their standard deviation bars, the active learning approaches do not only perform better, they are also more stable as shown by the smaller standard deviation values. Similar results are shown on the most complex and largest data set at hand, ACM-DBLP.

In addition to being accurate, our approach is very time-efficient. For example, it only required approximately 250ms to run a specification on the first and second data sets when all the data was in memory. On average, it requires less than 700ms on the last data set. It is important to notice that the features of this dataset include real numbers, which considerably worsen the runtime of link specifications.

Our results suggest that the use of small populations affects the outcome of the learning process significantly, especially when the specification to be learned is complex. For example, on the Publications data set, the learners that rely on solely 20 individuals per generation are up to 9.8% worse than the learners which use populations of 100 individuals. Setting the population to 100 seems to

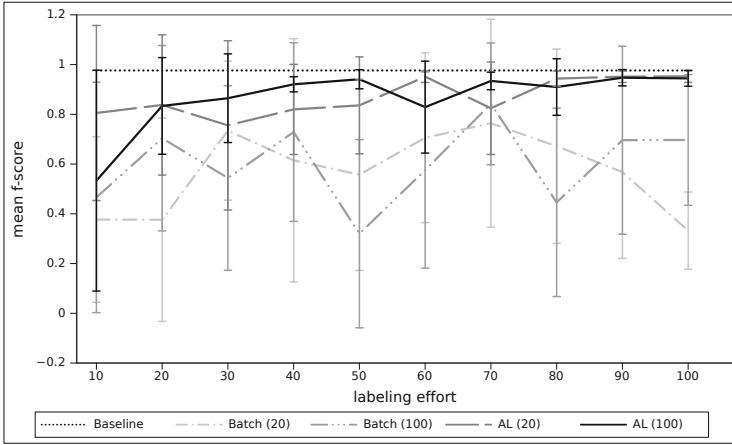


Fig. 9. Results of the Movies experiment. Mean F-measures of five runs of batch and active learner, both using population sizes of 20 and 100 individuals. The baseline is at 97.6% F-measure.

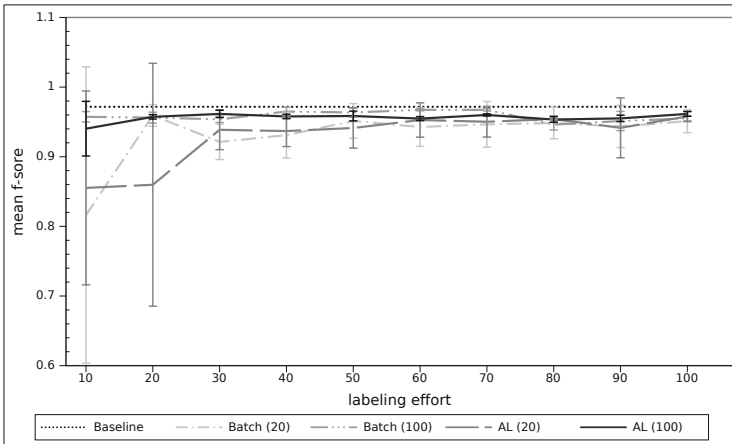


Fig. 10. Results of the Publications experiment. Mean F-measures of five runs of batch and active learner, both using population sizes of 20 and 100 individuals. The baseline is at 97.2% F-measure.

generate sufficiently good results without requiring a large amount of memory. Yet, when trying to link very complex datasets, an even larger setting would be advisable.

5.3 Comparison with other Approaches

As stated above, we chose the ACM-DBLP data set because it has been used in previous work to compare the accuracy and learning curve of different machine learning approaches for deduplication. As our results show (see Table 2), we reach an accuracy comparable to that of the other approaches. One of the main advantages of our approach is that it is considerably more time-efficient than all other approaches. Especially, while we are approximately 3 to 7 times faster than MARLIN, we are more than 14 times faster than FeBRL on this data set.

Table 2. Comparison of best performances of different machine learning approaches on ACM-DBLP

	EAGLE	FeBRL (SVM)	MARLIN (SVM)	MARLIN (AD-Tree)
F-Measure	97.2%	97.5%	97.6%	96.9%
Runtime	337s	4320s	2196s	1553s

So far, only a few other approaches have been developed for learning link specifications from data. RAVEN [22] is an active learning approach that views the learning of link specifications as a classification task. While it bears the advantage of being deterministic, it is limited to learning certain types of classifiers (boolean or linear). Thus, it is only able to learn a subset of the specifications that can be generated by EAGLE. Another genetic programming-based approach to link discovery is implemented in the SILK framework [15]. This approach is yet a batch learning approach and it consequently suffers of drawbacks of all batch learning approaches as it requires a very large number of human annotations to learn link specifications of a quality comparable to that of EAGLE.

6 Conclusion and Future Work

In this paper we presented EAGLE, an active learning approach for genetic programming that can learn highly accurate link specifications. We compared EAGLE with its batch learning counterpart. We showed that by using active learning, we can tackle complex datasets with more ease and generate solutions that are more stable (i.e., that display a smaller standard deviation over different runs). We also compared EAGLE with other approaches such as FeBRL and MARLIN on the ACM-DBLP dataset. We showed that for we achieve a similar F-measure while requiring a significantly smaller runtime. We also demonstrated that the runtime of our approach makes it suitable for interactive scenarios. In future work, we will study the effect of different parameterizations in more details. Especially, we will utilize different fitness functions and study the correlation of fitness functions with the overall F-score. Furthermore, we will aim at devising automatic configuration approaches for EAGLE.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. *SIGMOD Rec.* 22, 207–216 (1993)
2. Arasu, A., Götz, M., Kaushik, R.: On active learning of record matching packages. In: *SIGMOD Conference*, pp. 783–794 (2010)
3. Auer, S., Lehmann, J., Ngonga Ngomo, A.-C.: Introduction to Linked Data and Its Lifecycle on the Web. In: Polleres, A., d’Amato, C., Arenas, M., Handschuh, S., Kroner, P., Ossowski, S., Patel-Schneider, P. (eds.) *Reasoning Web 2011*. LNCS, vol. 6848, pp. 1–75. Springer, Heidelberg (2011)
4. Bilenko, M., Mooney, R.J.: Adaptive duplicate detection using learnable string similarity measures. In: *KDD*, pp. 39–48 (2003)
5. Bleiholder, J., Naumann, F.: Data fusion. *ACM Comput. Surv.* 41(1), 1–41 (2008)
6. Carvalho, M.G., Laender, A.H.F., Gonçalves, M.A., da Silva, A.S.: Replica identification using genetic programming. In: *Proceedings of the 2008 ACM Symposium on Applied Computing, SAC 2008*, pp. 1801–1806. ACM, New York (2008)
7. Christen, P.: Febrl -: an open source data cleaning, deduplication and record linkage system with a graphical user interface. In: *KDD 2008*, pp. 1065–1068 (2008)
8. Cristianini, N., Ricci, E.: Support vector machines. In: Kao, M.-Y. (ed.) *Encyclopedia of Algorithms*. Springer (2008)
9. Cudré-Mauroux, P., Haghani, P., Jost, M., Aberer, K., de Meer, H.: idmesh: graph-based disambiguation of linked data. In: *WWW*, pp. 591–600 (2009)
10. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering* 19, 1–16 (2007)
11. Glaser, H., Millard, I.C., Sung, W.-K., Lee, S., Kim, P., You, B.-J.: Research on linked data and co-reference resolution. Technical report, University of Southampton (2009)
12. Hassanzadeh, O., Consens, M.: Linked movie data base. In: Bizer, C., Heath, T., Berners-Lee, T., Idehen, K. (eds.) *Proceedings of the WWW 2009 Workshop on Linked Data on the Web, LDOW 2009* (2009)
13. Hogan, A., Polleres, A., Umbrich, J., Zimmermann, A.: Some entities are more equal than others: statistical methods to consolidate linked data. In: *Workshop on New Forms of Reasoning for the Semantic Web: Scalable & Dynamic (NeFoRS 2010)* (2010)
14. Isele, R., Jentzsch, A., Bizer, C.: Efficient Multidimensional Blocking for Link Discovery without losing Recall. In: *WebDB* (2011)
15. Isele, R., Bizer, C.: Learning Linkage Rules using Genetic Programming. In: *Sixth International Ontology Matching Workshop* (2011)
16. Sathya Keerthi, S., Lin, C.-J.: Asymptotic behaviors of support vector machines with gaussian kernel. *Neural Comput.* 15, 1667–1689 (2003)
17. Köpcke, H., Thor, A., Rahm, E.: Comparative evaluation of entity resolution approaches with fever. *Proc. VLDB Endow.* 2(2), 1574–1577 (2009)
18. Koza, J.R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)*. The MIT Press (1992)
19. Liere, R., Tadepalli, P.: Active learning with committees for text categorization. In: *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pp. 591–596 (1997)
20. Ngonga Ngomo, A.-C.: A Time-Efficient Hybrid Approach to Link Discovery. In: *Sixth International Ontology Matching Workshop* (2011)

21. Ngonga Ngomo, A.-C., Auer, S.: LIMES - A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data. In: Proceedings of IJCAI (2011)
22. Ngonga Ngomo, A.-C., Lehmann, J., Auer, S., Höffner, K.: RAVEN – Active Learning of Link Specifications. In: Proceedings of OM@ISWC (2011)
23. 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)
24. Papadakis, G., Ioannou, E., Niedere, C., Palpanasz, T., Nejd, W.: Eliminating the redundancy in blocking-based entity resolution methods. In: JCDL (2011)
25. Raimond, Y., Sutton, C., Sandler, M.: Automatic interlinking of music datasets on the semantic web. In: Proceedings of the 1st Workshop about Linked Data on the Web (2008)
26. Scharffe, F., Liu, Y., Zhou, C.: RDF-AI: an architecture for RDF datasets matching, fusion and interlink. In: Proc. IJCAI 2009 Workshop on Identity, Reference, and Knowledge Representation (IR-KR), Pasadena, CA, US (2009)
27. Settles, B.: Active learning literature survey. Technical Report 1648, University of Wisconsin-Madison (2009)
28. Sleeman, J., Finin, T.: Computing foaf co-reference relations with rules and machine learning. In: Proceedings of the Third International Workshop on Social Data on the Web (2010)
29. 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)
30. Winkler, W.: Overview of record linkage and current research directions. Technical report, Bureau of the Census - Research Report Series (2006)
31. Yuan, Y., Shaw, M.J.: Induction of fuzzy decision trees. *Fuzzy Sets Syst.* 69, 125–139 (1995)

Combining Information Extraction, Deductive Reasoning and Machine Learning for Relation Prediction

Xueyan Jiang², Yi Huang^{1,2}, Maximilian Nickel², and Volker Tresp^{1,2}

¹ Siemens AG, Corporate Technology, Munich, Germany

² Ludwig Maximilian University of Munich, Munich, Germany

Abstract. Three common approaches for deriving or predicting instantiated relations are information extraction, deductive reasoning and machine learning. Information extraction uses subsymbolic unstructured sensory information, e.g. in form of texts or images, and extracts statements using various methods ranging from simple classifiers to the most sophisticated NLP approaches. Deductive reasoning is based on a symbolic representation and derives new statements from logical axioms. Finally, machine learning can both support information extraction by deriving symbolic representations from sensory data, e.g., via classification, and can support deductive reasoning by exploiting regularities in structured data. In this paper we combine all three methods to exploit the available information in a modular way, by which we mean that each approach, i.e., information extraction, deductive reasoning, machine learning, can be optimized independently to be combined in an overall system. We validate our model using data from the YAGO2 ontology, and from Linked Life Data and Bio2RDF, all of which are part of the Linked Open Data (LOD) cloud.

1 Introduction

The prediction of the truth value of a (instantiated) relation or statement (i.e., a link in an RDF graph) is a common theme in such diverse areas as information extraction (IE), deductive reasoning and machine learning. In the course of this paper we consider statements in form of (s, p, o) RDF-triples where s and o are entities and where p is a predicate. In IE, one expects that the relation of interest can be derived from subsymbolic unstructured sensory data such as texts or images and the goal is to derive a mapping from sensory input to statements. In deductive reasoning, one typically has available a set of facts and axioms and deductive reasoning is used to derive additional true statements. Relational machine learning also uses a set of true statements but estimates the truth values of novel statements by exploiting regularities in the data. Powerful methods have been developed for all three approaches and all have their respective strengths and shortcomings. IE can only be employed if sensory information is available that is relevant to a relation, deductive reasoning can only derive a small subset of all statements that are true in a domain and relational machine learning is

only applicable if the data contains relevant statistical structure. The goal of this paper is to combine the strengths of all three approaches modularly, in the sense that each step can be optimized independently. In a first step, we extract triples using IE, where we assume that the extracted triples have associated certainty values. In this paper we will only consider IE from textual data. Second, we perform deductive reasoning to derive the set of provably true triples. Finally, in the third step, we employ machine learning to exploit the dependencies between statements. The predicted triples are then typically ranked for decision support. The complete system can be interpreted as a form of scalable hierarchical Bayesian modeling. We validate our model using data from the YAGO2 ontology, and from Linked Life Data and Bio2RDF, all of which are part of the Linked Open Data (LOD) cloud.

The paper is organized as follows. The next section discusses related work. Section 3 describes and combines IE and deductive reasoning. Section 4 describes the relational learning approach. Section 5 presents various extensions and in Section 6 we discuss scalability. Section 7 contains our experimental results and Section 8 presents our conclusions.

2 Related Work

Multivariate prediction generalizes supervised learning to predict several variables jointly, conditioned on some inputs. The improved predictive performance in multivariate prediction, if compared to simple supervised learning, has been attributed to the sharing of statistical strength between the multiple tasks, i.e., data is used more efficiently (see [32] and citations therein for a review). Due to the large degree of sparsity of the relationship data in typical semantic graph domains, we expect that multivariate prediction can aid the learning process in such domains.

Our approach is also related to conditional random fields [20]. The main differences are the modularity of our approach and that our data does not exhibit the linear structure in conditional random fields.

Recently, there has been quite some work on the relationship between kernels and graphs [7, 33, 11]. Kernels for semi-supervised learning have, for example, been derived from the spectrum of the Graph-Laplacian. Kernels for semantically rich domains have been developed by [8]. In [36, 35] approaches for Gaussian process based link prediction have been presented. Link prediction is covered and surveyed in [27, 13]. Inclusion of ontological prior knowledge to relational learning has been discussed in [28].

From early on there has been considerable work on supporting ontologies using machine learning [24, 9, 21], while data mining perspectives for the Semantic Web have been described by [1, 25]. A recent overview of the state of the art has been presented in [29]. The transformation of text into the RDF structure of the semantic web via IE is a highly active area of research [23, 30, 5, 6, 2, 4, 34, 3, 26, 14]. [22] describes a perspective of ILP for the Semantic Web. We consider machine learning approaches that have been applied to relation prediction in the context with the Semantic Web. In [19] the authors describe SPARQL-ML, a framework for adding data mining support to SPARQL. SPARQL-ML was

inspired by Microsoft’s Data Mining Extension (DMX). A particular ontology for specifying the machine learning experiment is developed. The approach uses Relational Bayes Classifiers (RBC) and Relational Probabilistic Trees (RPT).

3 Combining Sensory Information and Knowledge Base

3.1 Relation Prediction from Sensory Inputs

The derivation of relations from subsymbolic unstructured sensory information such as texts and images is a well-studied area in IE. Let X stand for a random variable that has state one if the (s, p, o) statement of interest is true and is zero otherwise. We assume that the IE component can estimate

$$P(X = 1|S)$$

which is the probability that the statement represented by X is true given the sensory information S . Otherwise no restrictions apply to the IE part in our approach, e.g., it could be based on rules or on statistical classifiers. Note that IE is limited to predict statements for which textual or other sensory information is available.

In the applications we have textual information text_s describing the subject and textual information text_o describing the object and we can write¹

$$P(X = 1|\text{text}_s, \text{text}_o). \quad (1)$$

In other applications we might also exploit text that describes the predicate text_p or text that describes the relationship $\text{text}_{s,p,o}$ (e.g, a document where a user (s) evaluates a movie (o) and the predicate is p ="likes") [16]. A recent overview on state of the art IE methods for textual data can be found in [29].

3.2 Relations from the Knowledge Base

In addition to sensory information, we assume that we have available a knowledge base in form of a triple store of known facts forming an RDF graph. Conceptually we add all triples that can be derived via deductive reasoning². State of the art scalable deductive reasoning algorithms have been developed, e.g., in [10]. Note that deductive reasoning typically can only derive a small number of nontrivial statements of all actually true statements in a domain.

We will also consider the possibility that the knowledge base contains some uncertainty, e.g., due to errors in the data base. So for triples derived from the knowledge base KB we specify

$$P(X = 1|KB)$$

to be a number close to one.

¹ For example, these texts might come from the corresponding Wikipedia pages.

² Here, those triples can either be inferred explicitly by calculating the deductive closure or on demand.

For all triples that cannot be proven to be true, we assume that $P(X = 1|KB)$ is a small nonnegative number. This number reflects our belief that triples not known to be true might still be true.

3.3 Combining Sensory Information and Knowledge Base

Now we combine sensor information and information from the knowledge base. Let $P(X = 1|S, KB)$ be the probability that the statement presented by X is true given the knowledge base and sensory information. The heuristic rule that we apply is very simple:

$$P(X = 1|S, KB) = P(X = 1|S) \quad \text{if } P(X = 1|S) > P(X = 1|KB)$$

$$P(X = 1|S, KB) = P(X = 1|KB) \quad \text{otherwise.}$$

Thus the probability of a statement derived from sensory information overwrites the default knowledge base values, if the former one is larger. Therefore, we rely on the knowledge base unless IE provides substantial evidence that a relation is likely.

4 Adding Relational Machine Learning

In many applications there is information available that is neither captured by sensory information nor by the knowledge base. A typical example is collaborative preference modeling which exploits correlations between preferences for items. Such probabilistic dependencies cannot easily be captured in logical expressions and typically are also not documented in textual or other sensory form. Relational machine learning attempts to capture exactly these statistical dependencies between statements and in the following we will present an approach that is suitable to also integrate sensory information and a knowledge base. Although there are probably a number of heuristic ways to combine sensory information and the knowledge base with machine learning, it is not straightforward to come up with consistent probabilistic models. Probabilistic generative models would require $P(S, KB|\{X\})$ where $\{X\}$ is the set of all random variables of all statements. Unfortunately, it is not clear how such a term could be derived. In the next subsections we develop an approach that works with the simpler $P(X|S, KB)$ and can be justified from a Bayesian modeling point of view.

4.1 Notation

Consider (s, p, o) triple statements where s and o are entities and p is a predicate. Note that a triple typically describes an attribute of a subject, e.g., (Jack, height, Tall), or a relationship (Jack, likes, Jane). Consider, that $\{e_i\}$ is the set of known entities in the domain. We assume that each entity is assigned to exactly one class $c(i)$. This assumption will be further discussed in Section 5. Let N_c be the number of entities in class c .

We also assume that the set of all triples in which an entity e_i can occur as a subject is known and is a finite, possibly large, ordered set (more details later) and contains $M_{c(i)}$ elements. For each potential triple (s, p, o) we introduce a random variable X which is in state one when the triple is true and is in state zero otherwise. More precisely, $X_{i,k} = 1$ if the k -th triple involving e_i as a subject is true and $X_{i,k} = 0$ otherwise. Thus, $\{X_{i,k}\}_{k=1}^{M_{c(i)}}$ is the set of all random variables assigned to the subject entity e_i .

We now assume that there are dependencies between all statements with the same subject entity.

4.2 A Generative Model

Following the independence assumptions we train a separate model for each class. So in this section we only consider the subset of statements which all have entities from the same entity class c .

The generative model is defined as follows. We assume that for each entity e_i which is a subject in class c there is a d -dimensional latent variable vector h_i which is generated as

$$h_i \sim N(0, I) \quad (2)$$

from a Gaussian distribution with independent components and unit-variance.

Then for each entity e_i a vector $\alpha_i = (\alpha_{i,1}, \dots, \alpha_{i,M_c})^T$ is generated, following

$$\alpha_i = Ah_i \quad (3)$$

where A is a $M_C \times d$ matrix with orthonormal columns.

From α_i we derive

$$P(X_{i,k} = 1 | S, KB) = \text{sig}(\alpha_{i,k}) \quad (4)$$

where $\text{sig}(in) = 1/(1 + \exp(-in))$ is the logistic function. In other words, $\alpha_{i,k}$ is the true but unknown activation that specifies the probability of observing $X_{i,k} = 1$. Note that $\alpha_{i,k}$ is continuous with $-\infty < \alpha_{i,k} < \infty$ such that a Gaussian distribution assumption is sensible, whereas discrete probabilities are bounded by zero and one.

We assume that $\alpha_{i,k}$ is not known directly, but that we have a noisy version available for each $\alpha_{i,k}$ in the form of

$$f_{i,k} = \alpha_{i,k} + \epsilon_{i,k} \quad (5)$$

where $\epsilon_{i,k}$ is independent Gaussian noise with variance σ^2 . $f_{i,k}$ is now calculated in the following way from sensory information and the knowledge base. We simply write

$$\hat{P}(X_{i,k} = 1 | S, KB) = \text{sig}(f_{i,k})$$

and sensory and the knowledge base is transferred into

$$f_{i,k} = \text{inv-sig}(\hat{P}(X_{i,k} = 1 | S, KB)) \quad (6)$$

where inv-sig is the inverse of the logistic function. Thus probabilities close to one are mapped to large positive f -values and probabilities close to zero are mapped to large negative f -values. The resulting F -matrix contains the observed data in the probabilistic model (see Figure 1).

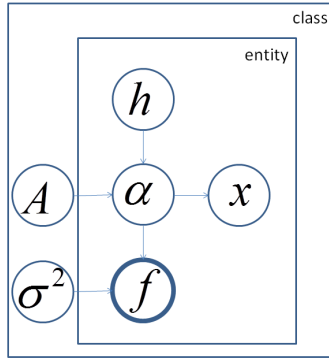


Fig. 1. Graphical plate model for the data generating process

4.3 Calculating the Solution

Note that our generative model corresponds to the probabilistic PCA (pPCA) described in [31] and thus we can use the learning equations from that paper.

Let F be the $N_c \times M_c$ matrix of f -values for class c and let

$$C = F^T F$$

be the empirical correlation matrix. The likelihood is maximized when

$$\hat{A} = U_d(\Lambda_d - \sigma^2 I)^{1/2} R \tag{7}$$

where the d column vectors in the $N_c \times d$ matrix U_d are the principal eigenvectors of C , with corresponding eigenvalues $\lambda_1, \dots, \lambda_d$ in the $d \times d$ diagonal matrix Λ_d and R is an arbitrary $d \times N_c$ orthogonal rotation matrix.³ We also get

$$\hat{\sigma}^2 = \frac{1}{M_c - d} \sum_{j=d+1}^{M_c} \lambda_j.$$

Finally, we obtain

$$\hat{\alpha}_i = \hat{A} M^{-1} \hat{A}^T f_i. \tag{8}$$

³ A practical choice is the identity matrix $R = I$. Also note that we assume that the mean is equal to zero, which can be justified in sparse domains.

Here, $f_i = (f_{i,1}, \dots, f_{i,M_c})^T$ is the vector of f -values assigned to e_i and $M = \hat{A}^T \hat{A} + \hat{\sigma}^2 I$. Note that M is diagonal such that the inverse is easily calculated as

$$\hat{\alpha}_i = U_d \operatorname{diag} \left(\frac{\lambda_j - \hat{\sigma}^2}{\lambda_j} \right) U_d^T f_i. \quad (9)$$

$\hat{\alpha}_i$ is now used in Equation 4 to determine the probability that $X_{i,k} = 1$, which is then, e.g., the basis for ranking. Also

$$\operatorname{diag} \left(\frac{\lambda_j - \hat{\sigma}^2}{\lambda_j} \right)$$

is a diagonal matrix where the j -th diagonal term is equal to $\frac{\lambda_j - \hat{\sigma}^2}{\lambda_j}$ 4

5 Comments and Extensions

5.1 A Joint Probabilistic Model

There are many ways of looking at this approach, maybe the most interesting one is a hierarchical Bayesian perspective. Consider each $\alpha_{i,k}$ to be predicted as a function of S and KB . In hierarchical Bayesian multitask learning one makes the assumption that, for a given entity e_i , the $\{\alpha_{i,k}\}_{k=1}^{M_{c(i)}}$ are not independent but are mutually coupled and share statistical strength [12]. This is achieved exactly by making the assumption that they are generated from a common multivariate Gaussian distribution. Thus our approach can be interpreted as hierarchical Bayesian multitask learning which can scale up to more than a million of tasks, i.e., potential statements per item.

Note that we suggest to train an independent model for each class and we obtain a joint probabilistic model over a complete domain with

$$P(\{X\}, \{h\} | \{F\}, \Theta) = \prod_c \prod_{i:c(i)=c} P(X_i | \alpha_i(h_i)) P(f_i | \alpha_i(h_i)) P(h_i).$$

$P(h_i)$ is given by Equation 2, where the dimension d might be dependent on the class $c(i)$ and $\alpha_i(h_i)$ is given by Equation 3. $P(X_i | \alpha_i(h_i))$ is given by Equation 4 (with $X_i = \{X_{i,k}\}_{k=1}^{M_{c(i)}}$) and $P(f_i | \alpha_i(h_i), \sigma_c^2)$ is given by Equation 5. Furthermore, $\{F\}$ is the set of F matrices for all classes and Θ is the set of all parameters, i.e., the A matrices and the σ^2 for all classes.

Note that each class is modeled separately, such that, if the number of entities per class and potential triples per entity are constant, machine learning scales linearly with the size of the knowledge base.

⁴ Note the great similarity of Equation 9 to the reduced rank penalized regression equation in the SUNS approach described in [15] which, in the notation of this paper, would assume the form $U_d \operatorname{diag}(\lambda_j / (\lambda_j + \gamma)) U_d^T f_i$ where $\gamma \geq 0$ is a regularization parameter. In some experiments we used this equation which exhibited greater numerical stability.

Finally we want to comment on how we define the set of all possible triples under consideration. In most applications there is prior knowledge available about what triples should be considered. Also, typed relations constrain the number of possible triples. In some applications it makes sense to restrict triples based on observed triples: We define the set of all possible statements in a class c to be all statements (s, p, o) where s is in class c and where the triple (s, p, o) has been observed in the data for at least one element of $s \in c$.

5.2 Generalization to New Entities

The most interesting case is when a new entity e_n that was not considered in training becomes known. If the class of the new entity is known, one can simply use Equation 8 to calculate a new α_n for a new f_n , which corresponds to the projection of a new data vector in pPCA. In case the class of the new entity is unknown, we can calculate α_n for the different classes under consideration and use Equation 5 to calculate the class specific probability.

5.3 Aggregation

After training, the learning model only considers dependencies between triples with the same subject entity. Here we discuss how additional information can be made useful for prediction.

Supplementing the Knowledge Base. The first approach is simply to add a logical construct into deductive reasoning that explicitly adds aggregated information. Let's assume that the triple $(?Person, livesIn, Germany)$ can be predicted with some certainty from $(?Person, bornIn, Germany)$. If the triple store does not contain the latter information explicitly but contains information about the birth city of a person, one can use a rule such as

$$(?Person, bornIn, Germany)$$

$$\leftarrow (?Person, bornIn, ?City) \wedge (?City, locatedIn, Germany)$$

and the derived information can be used in machine learning to predict the triple $(?Person, livesIn, Germany)$.

Enhancing IE. Some aggregation happens at the IE level. As an example, consider a text that describes a person (subject) and reveals that this person is a male teenager and consider another text that reveals that a movie (object) is an action movie. Then an IE system can learn that $(Person, likes, Movie)$ is more likely when the keywords “male”, “young” are present in the text describing the person and the keyword “action” is present in the text describing the movie.

We can also enhance the textual description using information from the knowledge base. If the knowledge base contains the statement $(Person, gender, Male)$ and $(Person, age, Young)$, we add the terms “male” and “young” to the keywords

describing the person. Similarly, if the knowledge base contains the statement (Movie, isGenre, Action), we add the term “action” to the keywords describing the movie.

5.4 Multiple Class Memberships

So far we have assumed that each entity can uniquely be assigned to a class. In many ontologies, an entity is assigned to more than one class. The most straightforward approach is to define for each entity a most prominent class. For example we might decide that from the class assignments (Jack, rdf:type, Student), (Jack, rdf:type, Person), (Jack, rdf:type, LivingBeing) that the second one is the prominent class which is used in the probabilistic model. The other two class assignments (i.e., type-of relations) are simply interpreted as additional statements (Jack, rdf:type, Student), (Jack, rdf:type, LivingBeing) assigned to the entity. As part of future work we will develop mixture approaches for dealing with multiple class assignments, but this is beyond the scope of this paper.

6 Scalability

We consider the scalability of the three steps: deductive reasoning, IE, and machine learning. Deductive reasoning with less expressive ontologies scales up to billions of statements [10]. Additional scalability can be achieved by giving up completeness. As already mentioned, each class is modeled separately, such that, if the number of entities per class and potential triples per entity are constant, machine learning scales linearly with the size of the knowledge base. The expensive part of the machine learning part is the eigen decomposition required in Equation 7. By employing sparse matrix algebra, this computation scales linearly with the number of nonzero elements in F . To obtain a sparse F , we exploit the sensory information only for the test entities and train the machine learning component only on the knowledge base information, i.e., replace $\hat{P}(X_{i,k} = 1|S, KB)$ with $\hat{P}(X_{i,k} = 1|KB)$ in Equation 6. Then we assume that $P(X = 1|KB) = \epsilon$ is a small positive constant ϵ for all triples that are not and cannot be proven true. We then subtract $\text{inv-sig}(\epsilon)$ from F prior to the decomposition and add $\text{inv-sig}(\epsilon)$ to all α . The sparse setting can handle settings with millions of entities in each class and millions of potential triples for each entity.

7 Experiments

7.1 Associating Diseases with Genes

As the costs for gene sequencing are dropping, it is expected to become part of clinical practice. Unfortunately, for many years to come the relationships between genes and diseases will remain only partially known. The task here is to predict diseases that are likely associated with a gene based on knowledge about gene and disease attributes and about known gene-disease patterns.

Disease genes are those genes involved in the causation of, or associated with a particular disease. At this stage, more than 2500 disease genes have been discovered. Unfortunately, the relationship between genes and diseases is far from simple since most diseases are polygenic and exhibit different clinical phenotypes. High-throughput genome-wide studies like linkage analysis and gene expression profiling typically result in hundreds of potential candidate genes and it is still a challenge to identify the disease genes among them. One reason is that genes can often perform several functions and a mutational analysis of a particular gene reveal dozens of mutation cites that lead to different phenotype associations to diseases like cancer [18]. An analysis is further complicated since environmental and physiological factors come into play as well as exogenous agents like viruses and bacteria.

Despite this complexity, it is quite important to be able to rank genes in terms of their predicted relevance for a given disease as a valuable tool for researchers and with applications in medical diagnosis, prognosis, and a personalized treatment of diseases.

In our experiments we extracted information on known relationships between genes and diseases from the LOD cloud, in particular from Linked Life Data and Bio2RDF, forming the triples (Gene, related_to, Disease). In total, we considered 2462 genes and 331 diseases. We retrieved textual information describing genes and diseases from corresponding text fields in Linked Life Data and Bio2RDF. For IE, we constructed one global classifier that predicts the likelihood of a gene-disease relationship based on the textual information describing the gene and the disease. The system also considered relevant interaction terms between keywords and between keywords and identifiers and we selected in total the 500 most relevant keywords and interaction terms. We did the following experiments

- ML: We trained a model using only the gene disease relationship, essentially a collaborative filtering system. Technically, Equation 6 uses $\hat{P}(X_{i,k} = 1|KB)$, i.e., no sensory information.
- IE: This is the predictive performance based only on IE, using Equation 11.
- ML + IE: Here we combine ML with IE, as discussed in the paper. We combine the knowledge base with IE as described in Section 3.3 and then apply Equation 6 and Equation 8.

Figure 2 shows the results. As can be seen, the performance of the IE part is rather weak and ML gives much better performance. It can nicely be seen that the combination of ML and IE is effective and provides the best results.

7.2 Predicting Writer’s Nationality in YAGO2

The second set of experiments was done on the YAGO2 semantic knowledge base. YAGO2 is derived from Wikipedia and also incorporates WordNet and GeoNames. There are two available versions of YAGO2: core and full. We used the first one, which currently contains 2.6 million entities, and describes 33 million facts about these entities. Our experiment was designed to predict the

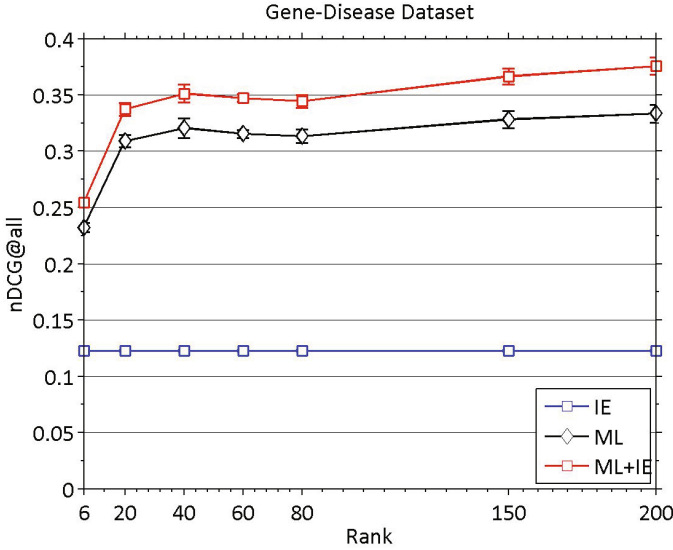


Fig. 2. Results on the Gene-Disease Data as a function of the rank d of the approximation. For each gene in the data set, we randomly selected one related_to statement to be treated as unknown (test statement). In the test phase we then predicted all unknown related_to entries, including the entry for the test statement. The test statement should obtain a high likelihood value, if compared to the other unknown related_to entries. The normalized discounted cumulative gain (nDCG@all) [17] is a measure to evaluate a predicted ranking.

nationalities of writers. We choose four different types of writers: American, French, German and Japanese. E.g., the triples for American writers are obtained with the SPARQL query:

```
SELECT ?writer ?birthPlace ?location WHERE {
  ?writer rdf:type ?nationality .
  ?writer yago:wasBornIn ?birthPlace .
  ?birthPlace yago:isLocatedIn ?location .
  FILTER regex(str( ?nationality ), "American_writers", "i")
}
```

We obtained 440 entities representing the selected writers. We selected 354 entities with valid yago:hasWikipediaUrl statements. We built the following five models:

- ML: Here we considered the variables describing the writers' nationality (in total 4) and added information on the city where a writer was born. In total, we obtained 233 variables. Technically, Equation 6 uses $\hat{P}(X_{i,k} = 1|KB)$, i.e., no sensory information.

- IE: As textual source, we used the Wikipage of the writers. We removed the terms 'German, French, American, Japanese' and ended up with 36943 keywords.
- ML+IE: We combined the knowledge base with IE as described in Section 3.3 and then applied Equation 6 and Equation 8.
- ML+AGG: We performed geo-reasoning to derive the country where a writer is born from the city that a writer was born. This aggregate information was added as a statement to the writer. Naturally, we expect a high correlation between country of birth and the writer's nationality (but there is no 100% agreement!).
- ML+AGG+IE: As ML+AGG but we added IE information using Equation 11.

We performed 10-fold cross validation for each model, and evaluated them with the area under precision and recall curve. Figure 3 shows the results. We see that the ML contribution was weak but could be improved significantly by adding information on the country of birth (ML+AGG). The IE component gives excellent performance but ML improves the results by approximately 3 percentage points. Finally, by including geo-reasoning, the performance can be improved by another percentage point. This is a good example where all three components, geo-reasoning, IE and machine learning fruitfully work together.

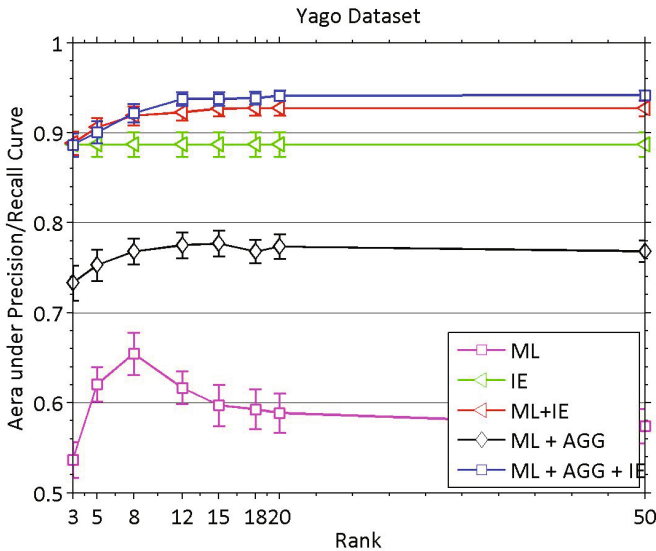


Fig. 3. The area under curve for the YAGO2 Core experiment as a function of the rank d of the approximation

8 Conclusions

In this paper we have combined information extraction, deductive reasoning and relational machine learning to integrate all sources of available information in a modular way. IE supplies evidence for the statements under consideration and machine learning models the dependencies between statements. Thus even if it is not evident that a patient has diabetes just from IE from text, our approach has the ability to provide additional evidence by exploiting correlations with other statements, such as the patient's weight, age, regular exercise and insulin intake. We discussed the case that an entity belongs to more than one ontological class and addressed aggregation. The approach was validated using data from the YAGO2 ontology, and the Linked Life Data ontology and Bio2RDF. In the experiments associating diseases with genes we could show that our approach to combine IE with machine learning is effective in applications where a large number of relationships need to be predicted. In the experiments on predicting writer's nationality we could show that IE could be combined with machine learning and geo-reasoning for the overall best predictions. In general, the approach is most effective when the information supplied via IE is complementary to the information supplied by statistical patterns in the structured data and if reasoning can add relevant covariate information.

References

1. Berendt, B., Hotho, A., Stumme, G.: Towards Semantic Web Mining. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 264–278. Springer, Heidelberg (2002)
2. Biemann, C.: Ontology learning from text: A survey of methods. LDV Forum 20(2) (2005)
3. Buitelaar, P., Cimiano, P.: Ontology Learning and Population: Bridging the Gap between Text and Knowledge. IOS Press (2008)
4. Cimiano, P.: Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. Springer (2006)
5. Cimiano, P., Hotho, A., Staab, S.: Comparing conceptual, divide and agglomerative clustering for learning taxonomies from text. In: Proceedings of the 16th European Conference on Artificial Intelligence, ECAI 2004 (2004)
6. Cimiano, P., Staab, S.: Learning concept hierarchies from text with a guided agglomerative clustering algorithm. In: Proceedings of the ICML 2005 Workshop on Learning and Extending Lexical Ontologies with Machine Learning Methods (2005)
7. Cumby, C.M., Roth, D.: On kernel methods for relational learning. In: ICML (2003)
8. D'Amato, C., Fanizzi, N., Esposito, F.: Non-parametric statistical learning methods for inductive classifiers in semantic knowledge bases. In: IEEE International Conference on Semantic Computing - ICSC 2008 (2008)
9. Fanizzi, N., d'Amato, C., Esposito, F.: DL-FOIL Concept Learning in Description Logics. In: Železný, F., Lavrač, N. (eds.) ILP 2008. LNCS (LNAI), vol. 5194, pp. 107–121. Springer, Heidelberg (2008)
10. Fensel, D., van Harmelen, F., Andersson, B., Brennan, P., Cunningham, H., Della Valle, E., Fischer, F., Huang, Z., Kiryakov, A., Lee, T.K.-L., Schooler, L., Tresp, V., Wesner, S., Witbrock, M., Zhong, N.: Towards larkc: A platform for web-scale reasoning. In: ICSC, pp. 524–529 (2008)

11. Gärtner, T., Lloyd, J.W., Flach, P.A.: Kernels and distances for structured data. *Machine Learning* 57(3) (2004)
12. Gelman, A., Carlin, J.B., Stern, H.S., Rubin, D.B.: *Bayesian Data Analysis*, 2nd edn. Chapman and Hall/CRC Texts in Statistical Science (2003)
13. Getoor, L., Diehl, C.P.: Link mining: a survey. *SIGKDD Explorations* (2005)
14. Grobelnik, M., Mladenic, D.: Knowledge discovery for ontology construction. In: Davies, J., Studer, R., Warren, P. (eds.) *Semantic Web Technologies*. Wiley (2006)
15. Huang, Y., Tresp, V., Bundschuh, M., Rettinger, A., Kriegel, H.-P.: Multivariate Prediction for Learning on the Semantic Web. In: Frasconi, P., Lisi, F.A. (eds.) *ILP 2010. LNCS*, vol. 6489, pp. 92–104. Springer, Heidelberg (2011)
16. Jakob, N., Müller, M.-C., Weber, S.H., Gurevych, I.: Beyond the stars: Exploiting free-text user reviews for improving the accuracy of movie recommendations. In: *TSA 2009 - 1st International CIKM Workshop on Topic-Sentiment Analysis for Mass Opinion Measurement* (2009)
17. Jarvelin, K., Kekalainen, J.: IR evaluation methods for retrieving highly relevant documents. In: *SIGIR 2000* (2000)
18. Kann, M.G.: Advances in translational bioinformatics: computational approaches for the hunting of disease genes. *Briefing in Bioinformatics* 11 (2010)
19. Kiefer, C., Bernstein, A., Locher, A.: Adding Data Mining Support to SPARQL via Statistical Relational Learning Methods. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) *ESWC 2008. LNCS*, vol. 5021, pp. 478–492. Springer, Heidelberg (2008)
20. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: *ICML* (2001)
21. Lehmann, J.: DL-learner: Learning concepts in description logics. *JMLR* (2009)
22. Lisi, F.A., Esposito, F.: An ilp perspective on the semantic web. In: *Semantic Web Applications and perspectives* (2005)
23. Maedche, A., Staab, S.: Semi-automatic engineering of ontologies from text. In: *Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering* (2000)
24. Maedche, A., Staab, S.: *Ontology Learning*. In: *Handbook on Ontologies 2004*. Springer (2004)
25. Mika, P.: *Social Networks and the Semantic Web*. Springer (2007)
26. Paaß, G., Kindermann, J., Leopold, E.: Learning prototype ontologies by hierarchical latent semantic analysis. In: *Knowledge Discovery and Ontologies* (2004)
27. Popescul, A., Ungar, L.H.: Statistical relational learning for link prediction. In: *Workshop on Learning Statistical Models from Relational Data* (2003)
28. Rettinger, A., Nickles, M., Tresp, V.: Statistical Relational Learning with Formal Ontologies. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) *ECML PKDD 2009. LNCS*, vol. 5782, pp. 286–301. Springer, Heidelberg (2009)
29. Sarawagi, S.: Information extraction. *Foundations and Trends in Databases* 1(3), 261–377 (2008)
30. Sowa, J.F.: Ontology, metadata, and semiotics. In: *International Conference on Computational Science* (2000)
31. Tipping, M.E., Bishop, C.M.: Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B* 61, 611–622 (1999)
32. Tresp, V., Yu, K.: Learning with dependencies between several response variables. In: *Tutorial at ICML 2009* (2009)
33. Vishwanathan, S.V.N., Schraudolph, N., Kondor, R.I., Borgwardt, K.: Graph kernels. *Journal of Machine Learning Research - JMLR* (2008)

34. Völker, J., Haase, P., Hitzler, P.: Learning expressive ontologies. In: Buitelaar, P., Cimiano, P. (eds.) *Ontology Learning and Population: Bridging the Gap between Text and Knowledge*. IOS Press (2008)
35. Xu, Z., Kersting, K., Tresp, V.: Multi-relational learning with gaussian processes. In: *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI 2009* (2009)
36. Yu, K., Chu, W., Yu, S., Tresp, V., Xu, Z.: Stochastic relational models for discriminative link prediction. In: *Advances in Neural Information Processing Systems, NIPS 2006* (2006)

Automatic Configuration Selection Using Ontology Matching Task Profiling*

Isabel F. Cruz¹, Alessio Fabiani¹, Federico Caimi¹, Cosmin Stroe¹,
and Matteo Palmonari²

¹ ADVIS Lab, Department of Computer Science, University of Illinois at Chicago, USA
{ifc, fabiani, fcaimi, cstroe1}@cs.uic.edu
² DISCO, University of Milan-Bicocca, Italy
palmonari@disco.unimib.it

Abstract. An ontology matching system can usually be run with different configurations that optimize the system’s effectiveness, namely precision, recall, or F-measure, depending on the specific ontologies to be aligned. Changing the configuration has potentially high impact on the obtained results. We apply matching task profiling metrics to automatically optimize the system’s configuration depending on the characteristics of the ontologies to be matched. Using machine learning techniques, we can automatically determine the optimal configuration in most cases. Even using a small training set, our system determines the best configuration in 94% of the cases. Our approach is evaluated using the AgreementMaker ontology matching system, which is extensible and configurable.

1 Introduction

Ontology matching is becoming increasingly important as more semantic data, i.e., data represented with Semantic Web languages such as RDF and OWL, are published and consumed over the Web especially in the *Linked Open Data (LOD)* cloud [14]. Automatic ontology matching techniques [10] are increasingly supported by more complex systems, which use a strategy of combining several *matching algorithms* or *matchers*, each taking into account one or more ontology features. Methods that combine a set of matchers range from linear combination functions [17] to matcher cascades [7, 10, 25], and to arbitrary combination strategies modeled as processes where specific matchers play the role of combiners [4, 16]. The choice of parameters for each matcher and of the ways in which the matchers can be combined may yield a large set of possible configurations. Given an *ontology matching task*, that is, a set of ontologies to be matched, an ontology engineer will painstakingly create and test many of those configurations manually to find the most effective one as measured in terms of precision, recall, or

* Research partially supported by NSF Awards IIS-0812258 and IIS-1143926 and by the Intelligence Advanced Research Projects Activity (IARPA) via Air Force Research Laboratory (AFRL) contract number FA8650-10-C-7061. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, AFRL, or the U.S. Government.

F-measure. Therefore, when given a new matching task, ontology engineers would like to start from their own set of available configurations and automatically determine the configuration that provides the best results without starting from scratch.

However, an ontology matching system that has been configured to match specific ontologies may not produce good results when matching other types of ontologies. For example, LOD ontologies vary widely; they can be very small (at the schema level), shallow, and poorly axiomatized such as GeoNames¹ large with medium depth and medium axiomatization such as in DBpedia² or large, deep, and richly axiomatized such as Yago³

In this paper we propose a machine learning method that takes as input an ontology matching task (consisting of two ontologies) and a set of configurations and uses matching task profiling to automatically select the configuration that optimizes matching effectiveness. Our approach is implemented in the AgreementMaker⁴ ontology matching system and evaluated against the datasets provided by the 2011 Ontology Alignment Evaluation Initiative (OAEI)⁴. We show that the automatically configured system outperforms the manually configured system. Furthermore, since the OAEI datasets are designed to test systems on a variety of ontologies characterized by heterogeneous features, they provide a rich and varied testbed. This testbed demonstrates the capability of our method to match many real-world ontologies and to achieve good effectiveness on new matching tasks without requiring manual tuning.

Although there are several machine learning approaches for ontology and schema matching, our approach differs from them in key aspects. Some approaches learn to classify mappings as correct or incorrect by combining several matchers [8] or similarity functions [13]. Others exploit user feedback to learn optimal settings for a number of parameters such as the similarity threshold used for mapping selection [25], or the weights of a linear combination of several matchers and the rate of iterative propagation of similarity [7]. Our approach learns how to select the optimal configuration for a matching task without imposing any restriction on the complexity of the adopted configurations, thus allowing for the reuse of fine-tuned configurations tested in previous ontology matching projects. We also achieve very good results with a limited training set by focusing on the feature selection problem, identifying fine-grained ontology profiling metrics, and combining these metrics to profile the ontology matching task.

The rest of this paper is organized as follows: Section 2 defines the problem of configuration selection and describes the configurations of the AgreementMaker system used in our experiments. Section 3 describes the proposed learning method to automatically select the best configuration and discusses the matching task profiling techniques adopted. Section 4 describes the experiments carried out to evaluate the approach. Section 5 discusses related work. Finally, Section 6 presents our conclusions.

2 Preliminaries

In this section we will explain some preliminary concepts, define the problem, and give a brief description of our AgreementMaker system and its configurations.

¹ <http://www.geonames.org>

² <http://www.dbpedia.org>

³ <http://www.mpi-inf.mpg.de/yago-naga/yago/>

⁴ <http://oaei.ontologymatching.org/2011/>

2.1 Problem Definition

Ontology Matching (or *Ontology Alignment*) is defined as the process of finding correspondences between semantically related concepts in different ontologies [10]. Typically two ontologies are considered at a time—the *source* and the *target*—but the overall problem can involve several ontologies to be matched. The resulting correspondences are called *mappings*, a set of mappings is called an *alignment* and the correct alignment, as determined by domain experts, is called the *reference alignment* or *gold standard*. The algorithms used to discover the alignments are called *matchers*. A *basic matcher* takes into account a single aspect of the concepts to be matched, while a *complex matcher* is the result of combining and tuning basic matchers. The aggregation of matching results is called *combination*. The possible ways of combining results are mainly two: *series* and *parallel*. The former means that a matcher starts its computation based on the output result of another matcher, while the latter consists of taking the results from several matchers as input and providing a single output [4]. The resulting combination is called a *matcher stack*.

The word *configuration* refers to an already tuned matcher stack, usually organized in combination layers, whose number depends on the various series and parallel combinations that take place. Developing a configuration is a very complicated and time-consuming process that requires many experiments performed by a domain expert to define the matcher stack and the parameters for each matcher. That is, many choices must be made: which basic matchers to use, how to combine them, how many layers, how to set the thresholds and other parameters for each matcher. Therefore, it is clear that the exploration of the whole space of possible configurations is not a practical option.

It is also highly unlikely for the same configuration to perform well on totally different domains. We will approach this challenge by considering a small number of different configurations that have been manually developed for a variety of ontology matching tasks. This means that we do not address the problem of developing configurations, as we use the most successful ones that we have manually implemented with the AgreementMaker ontology matching system; instead, we focus on how to choose the best configuration for each task. That is, given two ontologies to be matched—an *ontology matching task*—and a set of configurations of a given ontology matching system, the problem consists of determining automatically the configuration that maximizes the quality of the output alignment, measured in terms of precision, recall, or F-measure (the weighted harmonic mean of precision and recall).

2.2 Matcher Configurations

The AgreementMaker system [2, 3, 4, 5] is an extensible ontology matching framework that has been expanded to include many types of matching algorithms so as to handle different matching scenarios. At the heart of the system is its ability to efficiently combine the results from several matching algorithms into one single and better alignment [4]. This capability allows us to focus on developing new matching algorithms, which can be combined with previously developed algorithms, with the objective of improving our results.

The configurations used in our system, which are shown as block diagrams in Figure 1, are derived from our experience of matching ontologies over the years. Our

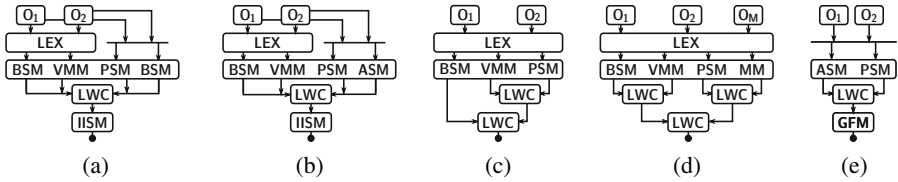


Fig. 1. The five configurations used in our system

work began with a general purpose configuration that consisted of syntactic matchers—the Base Similarity Matcher (BSM), the Parametric String Matcher (PSM), and the Vector-based Multi-word Matcher (VMM)—running in parallel and combined into a final alignment by the Linear Weighted Combination (LWC) Matcher [4]. We later extended our syntactic algorithms with lexicon lookup capabilities (LEX) and added a refining structural matcher—the Iterative Instance and Structural Matcher (IISM)—leading to the configuration shown in Figure 1(a). The configuration consists of running syntactic and lexical matching algorithms, combining their results, and then refining the combined alignment using a structural algorithm; this is a recurring pattern in all of our configurations.

Some ontologies may contain labels and local names that require more advanced string similarity techniques that are needed to perform syntactic matching. For this reason, our second configuration, which is shown in Figure 1(b), features the Advanced Similarity Matcher (ASM) [5]. This matcher extends BSM to find the similarity between syntactically complex concept labels.

As our work extended into matching biomedical ontologies [6] we found that the lexicon lookup capability of our algorithms became very important in those tasks. Such capability was especially useful because the biomedical ontologies include synonym and definition annotations for some of the concepts. For these types of ontologies, we use the configuration shown in Figure 1(c), which aggregates the synonyms and definitions into the lexicon data structure. There is more than one combination step so as to group similar matching algorithms before producing a final alignment.

An extension of the previous lexical-based configuration is shown in Figure 1(d). This configuration adds the Mediating Matcher (MM) to aggregate the synonyms and definitions of a third ontology, called the *mediating ontology*, into the lexicon data structure to improve recall. Finally, when matching several ontologies at a time, overall precision and runtime is more important. For this purpose we use a configuration, shown in Figure 1(e), which features the combination of two syntactic matchers and is refined by a structural matching algorithm that ensures precision and runtime efficiency.

3 Learning to Automatically Select the Best Configuration

Our approach can be sketched as follows: the matching task is profiled using several metrics, then the values for those metrics are used as input features to a machine learning algorithm that correlates the profiles to the performance of different configurations of the system. We are using a supervised learning approach and training the classifier on a subset of our dataset. Our goal is to use as small a subset as possible for training, as this reflects a real-world use of matching systems where users are able to provide only a very small subset of reference alignments.

Our proposed matching process follows the steps represented in Figure 2. First the pair of ontologies to be matched is evaluated by the matching task profiling algorithm. Based on the detected profile, a configuration is selected and the matcher stack is instantiated. Finally, the ontology matching step is performed and an output alignment is obtained.

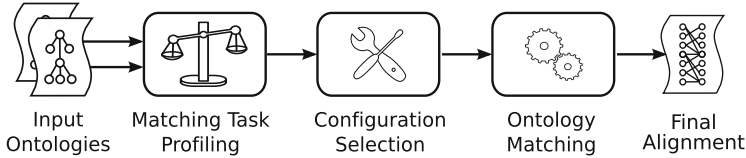


Fig. 2. Automatic configuration selection process

3.1 Ontology Profiling

Ontology profiling is the process of describing ontologies using a set of metrics. In particular, we include metrics that have been introduced in the context of ontology evaluation [26], where the goal is to evaluate the quality of an ontology. To the best of our knowledge, our approach is the first to apply these metrics in the context of ontology matching.

Since the aim of our work is to determine the most suitable system configuration for a given matching task, the first step is to characterize the task in terms of the two input ontologies. In particular, we use information about their classes and properties and combine the obtained profiles. This is a preprocessing phase with the goal of gathering information that is later used to discriminate between possible configurations. The metrics we use for ontology profiling can be evaluated by their expressiveness, clarity, and simplicity. A good metric should lend itself to be efficiently computed and to be correlated with some of the characteristics of the matchers that make up a configuration. The machine learning algorithm then exploits these correlations when building a classifier.

Our approach is unique because it is totally automatic, given a small training set. It makes no difference which matchers are part of a configuration, nor how they are combined. The machine learning algorithm will take care of summarizing the impact that some of the values of the metrics have on the performance of the matchers, without the need for explicit descriptions.

The metrics we used to profile the ontologies are shown in Table 1. The table also shows which ontology characteristics are taken into account by each metric (syntactic, lexical, structural, or instance). Some of these metrics are well known in the literature [8, 26], while others are being introduced in this paper. In what follows, C represents the set of classes in the ontology and T the set of terms, that is, the set of classes and properties. Therefore $|C|$ is the number of classes and $|T|$ is the number of terms.

Relationship Richness. The Relationship Richness (RR) of a schema is defined as the percentage of relations (object properties) at the schema level that are different from *subclassOf* relations. In the equation, P is the set of such relations and SC is the set of *subclassOf* relations [26].

Table 1. Ontology profiling metrics used by our system

Metric	Equation	Syntactic	Lexical	Structural	Instance
Relationship Richness	$RR = \frac{ P }{ SC + P }$			✓	
Attribute Richness	$AR = \frac{ att }{ C }$	✓		✓	
Inheritance Richness	$IR = \frac{\sum_{C_i \in C} H^C(C_i, C_i) }{ C }$			✓	
Class Richness	$CR = \frac{ C_i }{ C }$				✓
Label Uniqueness	$LU = \frac{ diff }{ T }$	✓			
Average Population	$P = \frac{ I }{ C }$				✓
Average Depth	$D = \frac{\sum_{C_i \in C} D(C_i)}{ C }$			✓	
WordNet Coverage	$WC_{i \in \{label, id\}} = \frac{ covered_i }{ C }$		✓		
Null Label and Comment	$N_{i \in \{label, comment\}} = \frac{ NC_i }{ T }$	✓			

Attribute Richness. Attribute Richness (AR) is defined as the average number of attributes (datatype properties) per class and is computed as the number of attributes for all classes (att) divided by the number of classes [26].

Inheritance Richness. Inheritance Richness (IR) describes the structure of an ontology. It is defined as the average number of subclasses per class [26]. The number of subclasses for a class C_i is defined as $|H^C(C_i, C_i)|$, where C_l is a subclass of C_i . An ontology with low IR has few children per class but many inheritance levels, while an ontology with high IR has few inheritance levels but many children per class.

Class Richness. Class Richness (CR) is defined as the ratio of the number of classes for which instances exist ($|C_i|$) divided by the total number of classes defined in the ontology [26].

Label Uniqueness. This metric captures the number of terms whose local name and label differ so as to determine whether we can find additional information in the term labels. We define Label Uniqueness (LU) as the percentage of terms that have a label that differs from their local name ($diff$).

Average Population. This metric provides an indication of the average distribution of instances across all the classes. Average Population (P) is defined as the number of instances $|I|$ divided by the number of classes [26].

Average Depth. This metric describes the average depth (D) of the classes in an ontology defined as the mean of the depth over all the classes C_i ($D(C_i)$).

WordNet Coverage. WordNet is a lexical database for the English language [20]. It groups English words into sets of synonyms called synsets, provides short, general definitions, and records the various semantic relations between these synsets. WordNet is generally used in the ontology matching process to get information about the words

contained in the attributes of a term. WordNet Coverage (WC) has been introduced as a feature that evaluates for each couple of terms whether none, one, or both of them can be found in WordNet [8]. Differently from previous approaches, in our system we compute WC as the percentage of terms with label or local name (id) present in WordNet ($covered$). We compute two WordNet Coverage metrics, one for local names and one for labels.

Null Label and Comment. Two other very important features of the ontology that we profile in our approach are the percentage of terms with no comment or no label, named respectively $N_{comment}$ or N_{label} . They are defined as the number of terms that have no comment ($|NC_{comment}|$) or no label ($|NC_{label}|$) divided by the number of terms.

3.2 Matching Task Profiling

Once all the metrics described in the previous section are computed for the two ontologies in the matching task, there are several methods we can use to compute the final matching task profile. We show that process in Figure 3. The first method consists of considering each metric value as a separate feature in the profile, which will be used as the feature vector for the classifier. In this case, the profile will contain just the values of the metrics for the source and target ontologies. We refer to this Source-Target profile configuration as ST . This approach, though, describes the ontologies singularly and does not give a compact summary of the pair. The second approach, instead, consists of combining the value pairs using a mathematical function. We experimented with two different combination functions: *Average* (A) and a newly defined function called *Feature Similarity* (FS). The former averages the values for a particular metric for both ontologies. It can be used to detect how much a particular characteristic is present in the matching task. For example, the average of the WordNet Coverage (WC) will be an indicator of how suitable a configuration relying on lexical matchers is for a particular matching task. When the average of WC is low, the model may learn to select a configuration without lexical matcher. Instead, the FS function evaluates how similar a characteristic is in the two ontologies. For example, when the average WC is high, FS explains whether WC is similar in the two ontologies or high in one and low in the other. Thus, the model may learn to select a configuration without lexical matcher even when WC is quite high on average, but very low in one ontology.

Given an ontology profiling metric m with two values, one for each ontology in the matching task, FS is defined as the ratio between the lower (m_L) and the higher (m_H)

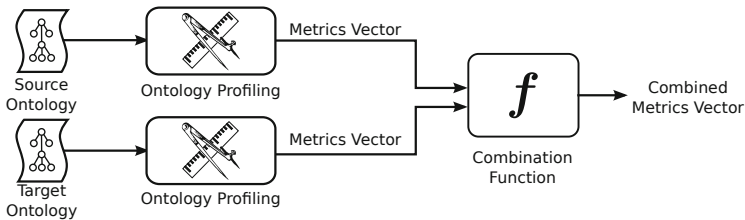


Fig. 3. Matching task profiling

of the two metric values, divided by the logarithm of their difference. When the two values are equal, the value of FS is one, while the more they differ, the closer to zero that value will be.

$$FS_m = \frac{m_L}{m_H [\log(m_H - m_L + 1) + 1]} \quad (1)$$

The concatenation of A and FS into a function $FS-A$ provides a more expressive measure. It considers how much a particular aspect is present in each of the two ontologies as well as how much they share that aspect. Experiments described in Section 4 confirm this intuition showing that $FS-A$ outperforms all the other functions.

3.3 Automatic Configuration Selection

For each input ontology pair (matching task) we compute the metrics previously described and use them at runtime to predict the best configuration. The space of all the possible ontology pairs to be matched can be divided into a number of subspaces. We define those subspaces as the sets of ontology pairs sharing the same configuration as the best configuration for those matching tasks. From a machine learning point of view, each subspace corresponds to a class to be predicted.

Differently from rule-based approaches, our approach allows us to find correlations between metrics and the matching configurations without explicitly define them. No matter how complex a matcher is, its performance on the training set will allow the learning algorithm to determine its suitability to new matching tasks.

Supervised learning consists of techniques which, based on a set of manually labeled training examples, create a function modeling the data [19]. Each training example is composed of a feature vector and a desired output value. When the range of the output value is a continuous interval, it is called a regression problem. In the discrete case, it is called classification. The function created by the algorithm should be able to predict the output value for a new input feature vector. In our problem, the input feature vector is the result of matching task profiling while the output class is one of the possible configurations.

Building a robust classifier is not trivial: the main problem is how to generate a good training set. It should be a highly representative subset of the problem's data. In addition, the larger it is, the higher the quality of the classification will be. An important aspect is the distribution of the classes in the training set. It should reflect the nature of the data to be classified and contain an adequate number of examples for every class.

In order to generate our training set, the system goes through the steps shown in Figure 4. For each ontology pair in a matching task, we compute the metrics introduced in Section 3.1 and stored as data points in the training set. The system is then run with all the given configurations, each generating an alignment for every matching task. We then evaluate the precision, recall, and F-measure of each generated alignment and store the results in an *evaluation matrix*. For each matching task, the configuration that optimizes the precision, recall, or F-measure (depending on the users' needs) is chosen and stored as the correct class for the corresponding data points in the training set.

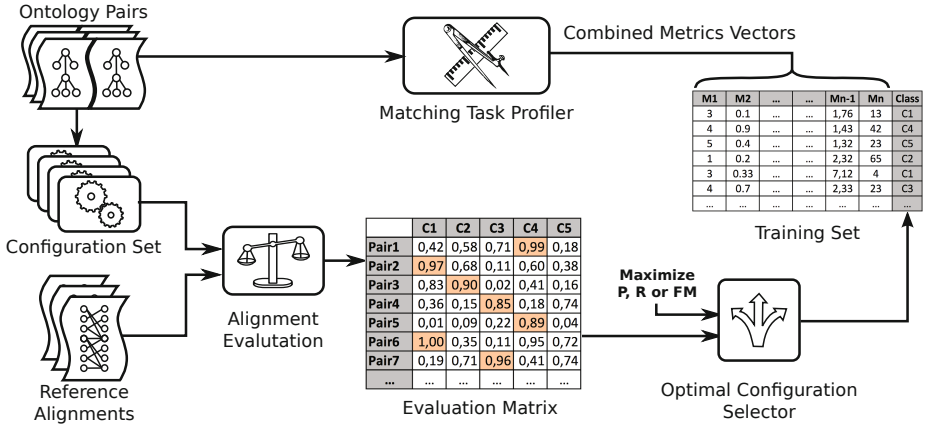


Fig. 4. Generation of the training set

We have tested this approach using many standard classifiers and including different feature vectors, which are compared in Section 4. The trained model is used by AgreementMaker to predict the correct configuration before starting a matching process. The system takes also into account the confidence value returned by the classification algorithm, which represents the certainty of the classifier’s prediction. If this value is under a certain threshold the system ignores the classification and chooses the default configuration (Figure 1(a)).

4 Evaluation and Results

In this section we describe the experimental results obtained using the approach introduced in Section 3. Our approach has been evaluated using the AgreementMaker system with the configurations explained in Section 2.2. Our experiments have been run using 233 matching tasks with reference alignments provided in OAEI 2011.

4.1 Learning Evaluation

For our evaluation, we use several classifiers in order to gauge their impact on correctly classifying the matching tasks. The classifiers we use are well-known, each belonging to a different category: k -NN (Instance-based) [11], Naive Bayes (Probability-based) [12], Multilayer Perceptron (Neural Network) [11], and C4.5 (Decision Tree) [24].

For our first experiment, which is shown in Table 2, we perform a k -fold cross-validation with $k = \{2, 10\}$ using each classifier and comparing the obtained accuracy, defined as the percentage of correctly classified instances. 10-fold cross-validation is considered the standard for evaluating a learning algorithm while we also tested with 2-fold cross-validation to gauge the robustness of our approach, since we want as small a training set as possible. We experimented with all of our combination functions to generate the matching task profile. As can be seen from the table, a k -NN classifier (with $k = 3$) exhibits the best accuracy in all the tests. Furthermore, we can see that

Table 2. Cross-validation accuracy of different classifiers and combination functions

Combination Function	Cross-validation	k-NN	Naive Bayes	Multilayer	C4.5
<i>ST</i>	10-fold	88.1%	55.0%	84.1%	85.9%
	2-fold	86.0%	57.0%	82.9%	83.6%
<i>A</i>	10-fold	85.7%	55.3%	84.5%	86.1%
	2-fold	82.9%	56.2%	82.6%	84.9%
<i>FS</i>	10-fold	87.6%	54.9%	84.9%	88.0%
	2-fold	85.8%	55.3%	82.6%	83.7%
<i>FS-A</i>	10-fold	89.9%	55.7%	88.1%	88.1%
	2-fold	89.4%	57.4%	83.8%	83.8%

the *FS-A* combination function has a significant impact on the overall results and in particular on making the approach more robust.

The graph of Figure 5(a) shows the accuracy that was obtained by varying the training set size and the classifier used. Each point is computed by averaging 100 runs. In each run, a random subset of matching tasks is selected for the training set and the model is evaluated against the remaining part of the dataset. Due to the obtained results, in this test we use *FS-A* as the combination function.

Naive Bayes is the worst performing classifier because the conditional independence assumption between the selected metrics does not hold. In other words, some of our metrics are interdependent. Since the dataset used is not big, instance-based classifiers perform slightly better than others. This is because they are able to learn quickly even from very small datasets. Other methods require a significant number of examples for each class to be capable of training robust models, while an instance-based classifier can make useful predictions using few examples per class. Therefore, this kind of classifier works well when limited data is available, as in our case. The other two classifiers,

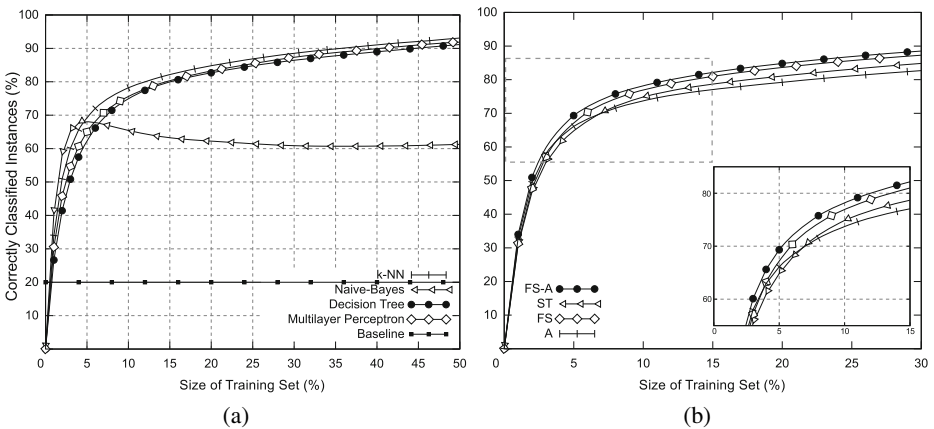


Fig. 5. Comparison of (a) classifiers and of (b) combination functions

Multilayer Perceptron and C4.5, show similar results, each being slightly worse than k -NN. The more the training set size increases, the more they close the gap with k -NN, confirming this characteristic of instance-based classifiers.

The results are also compared with a baseline. It is represented by the case in which the system randomly chooses between configurations. Since the number of configurations is five, the probability of selecting the right one is 20%, independently of the size of the training set. We believe these results are really valuable and encouraging, because even with a small training set our selection is able to outperform random selection substantially. As an example, when only 5% of the dataset is used for training, our method is able to choose the best configuration in seven cases out of ten. Moreover, in the other three cases, the results are only slightly worse than the best configuration. The two experiments above lead us to choose the k -NN classifier.

In Figure 5(b) we compare the combination functions, which we described in Section 3.2, as a function of the size of the training set. From the graphs in the figure it is clear that *FS-A* outperforms the other combination methods. Therefore we use *FS-A* in our following experiments.

4.2 OAEI Results

In the previous section, we have shown that our learning system is robust. In this section we use three tracks of the OAEI 2011 competition, namely Benchmark, Benchmark2, and Conference, to study the impact of our approach in terms of F-measure.

In order to guarantee an equally distributed and representative training set, we introduce the constraint that it should contain at least some representative instances for each of the five classes. Using this approach, our experiments were run using only 20% of the whole dataset as the training set, obtaining 94% accuracy. The remaining 6% of misclassified examples are further divided: 3.83% for which the algorithm chooses the second best configuration, and 2.17% for which it selects the third best one. The worst performing configurations (fourth and fifth) are never chosen. It is worth noting that even in the case where the algorithm fails to choose the best configuration, the chosen configuration is acceptable and in some cases is better than the one chosen manually.

Overall, our results were improved by using the automatic selection instead of the manual selection previously used in our system. This is because we are now able to choose the best configuration on a task-by-task basis, rather than on a track-by-track basis. Different configurations are used to match different tasks of the same domain, while selection performed by an expert is usually domain-specific.

In Table 3 we show the percentage of tasks in which our automatic approach outperforms the manual selection, for each of the considered tracks (Benchmark, Benchmark2, and Conference). We also show the average increase in F-measure for the tasks that were improved (the remaining ones were already optimal). Our new automatic selection leads to a significant improvement in all of the tracks, especially in the Conference track where more than half of the matching tasks showed improvement.

While in Table 3 we compare the results obtained over the whole testset, in Table 4 we show our improvements in terms of F-measure on some particularly interesting sub-tracks. We present an automatic versus manual selection comparison as before, but also versus an ideal classifier (i.e., a classifier for which the best configuration is always selected).

Table 3. Improvement of automatic vs. manual selection

	Benchmark	Benchmark2	Conference
Improved Tasks	8.0%	17.0%	57.0%
Avg. Gain/Task	3.4%	3.0%	12.7%

Table 4. Comparison between manual, automatic, and ideal selections

	Benchmark (301-304)	Benchmark2 (221-249)	Conference
Manual (M)	83.7%	82.4%	56.5%
Automatic (A)	86.7%	83.3%	61.0%
Ideal (I)	87.0%	83.6%	63.8%
$\Delta(A - M)$	3.0%	0.9%	4.5%
$\Delta(I - A)$	0.3%	0.3%	2.8%

We selected for this comparison a subset of the previously mentioned tracks, choosing the ones that we consider the most relevant, because they represent real-world examples. The sub-tracks are: Benchmark (301-304), Benchmark2 (221-249), and the Conference track as a whole. The Anatomy track is not included because it is composed of a single sub-track, which is correctly classified by both the automatic and manual selections. The table shows the average F-measures obtained by the selection approaches in these sub-tracks. We also show the difference between the automatic and manual selections ($\Delta(A - M)$) and between the ideal and automatic selections ($\Delta(I - A)$).

In Figure 6 we show the performance obtained by the manual, automatic, and ideal selections in the specified tasks of the Benchmark and Conference tracks. In most of these test cases the automatic selection chooses the correct configuration, that is, the automatically chosen configuration is also the ideal configuration. In some of these cases the manual selection chooses the correct configuration as well. An interesting case is provided by *confOf-ekaw*, where the three different modalities choose three different configurations. However, even in this case, the automatic selection chooses a better configuration than the one chosen manually.

5 Related Work

Several approaches have been proposed, whose objective is to improve the performance of ontology schema matching by automatically setting configuration parameters. An early approach considers a decision making technique that supports the detection of suitable matchings based on a questionnaire that is filled out by domain and matching experts [22]. In the continuation of this work, a rule-based system is used to rank a set of matchers by their expected performance on a particular matching task [21].

The RiMOM system profiles the ontology matching task by evaluating an overall lexical and structural similarity degree between the two ontologies [17]. These similarity measures are used to change the linear combination weights associated with a lexical and a structural matchers. These weights are adaptively set using matching task

profiling, however only two metrics are used, a much smaller number than the number considered in this paper. Furthermore, the configuration cannot be changed.

The learning-based approaches for system configuration fall under two types: (1) learning to classify correspondences as correct/incorrect [8, 15, 23]; (2) learning optimal parameter values for a system [13, 16, 18]. As compared to these approaches, our contribution is that we learn to select the best configuration among a set of available configurations for each matching task. This introduces a new challenge: we must consider features describing the ontologies and their mutual compatibility as a whole and define ontology metrics for this purpose, both being novel contributions.

We now describe in more detail three of the machine-learning approaches [8, 13, 16]. The first approach learns the combination of several matchers, considered as black boxes [8]. The system uses a *mapping classifier* to label mappings as correct or incorrect based on a mapping profile, which encompasses features of the matchers, lexical and structural features of the concepts to be matched, and very simple ontology features. Our configuration selection is based exclusively on ontology features and embeds several more expressive features. It also has the following two advantages. First, we avoid executing every configuration of the system because, given a matching task, only the optimal configuration is selected, leading to significant savings of time and computational resources. Second, our classifier is trained on a small subset of the overall dataset. In particular, where in our experiments the overall dataset is split in a 1:4 ratio between training and evaluation data, the results for [8] are obtained with a ratio of 3:1 in the Benchmark and 4:1 in the Conference datasets. Our datasets are approximately the same size, but we use a much smaller training set, requiring less user effort. To the best of our knowledge, the minimization of the training set size has not been investigated by others. The second approach features a technique to automatically configure a schema matching system, which has been implemented in the eTuner application [16]. This approach is interesting because it is based on the use of synthetic data obtained by perturbing some features of the schemas to be aligned. However, it

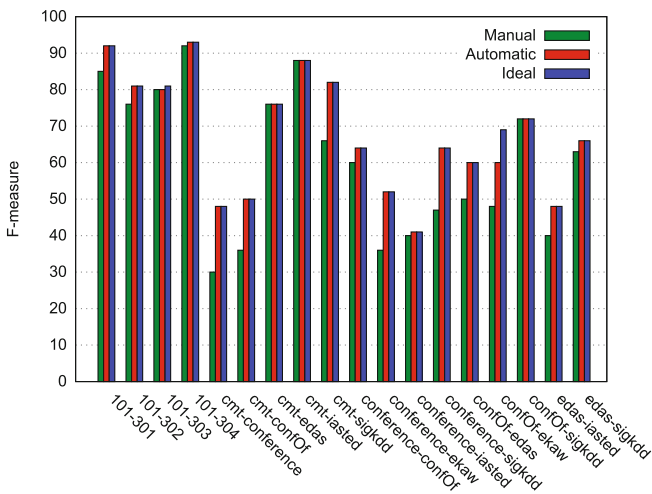


Fig. 6. Comparison between manual, automatic, and ideal selections for some of the track tasks

applies to matching relational schemas not to ontologies. The third approach combines several similarity metrics by learning an aggregated similarity function [13]. However, the degree to which the matching system is configured in our approach is significantly higher.

Finally, we mention some approaches that adopt active learning techniques in order to automatically tune the system on the basis of user feedback [7, 9, 25]. However, all these approaches learn to set system-specific parameters, such as the threshold [25], the optimal weights in a linear combination function, or the number of iterations of a similarity propagation matcher [7], while our system allows for configurations of arbitrary complexity.

6 Conclusions and Future Work

In this paper we made several contributions to the process of matching ontologies automatically. We proposed a learning approach to automatically select the best configuration from a predefined set. We used a set of metrics expressly chosen to describe ontologies and introduced combination functions to define a matching task profile. While other approaches can be more adaptive by not assuming an available configuration set and the effectiveness of our approach depends on the availability of a good configuration for that matching task, we show that using a relatively small number of heterogeneous configurations it is possible to achieve very good results. Furthermore, many ontology matching systems provide default configurations, which our approach can exploit.

Although we used the AgreementMaker ontology matching system, the strength of our approach is that it can produce excellent results for any system. The only requirements are: a set of matching tasks with reference alignment (for training), and a set of heterogeneous configurations covering diverse domains. Our work can be also extended by creating and evaluating new metrics and matching configurations. Another interesting research topic consists of adding a feedback mechanism to our approach [9].

References

- [1] Aha, D.W., Kibler, D., Albert, M.K.: Instance-based Learning Algorithms. *Machine Learning* 6(1), 37–66 (1991)
- [2] Cruz, I.F., Palandri Antonelli, F., Stroe, C.: Agreementmaker: Efficient matching for large real-world schemas and ontologies. *PVLDB* 2(2), 1586–1589 (2009)
- [3] Cruz, I.F., Palandri Antonelli, F., Stroe, C.: Integrated Ontology Matching and Evaluation. In: *International Semantic Web Conference, Posters & Demos* (2009)
- [4] Cruz, I.F., Palandri Antonelli, F., Stroe, C.: Efficient Selection of Mappings and Automatic Quality-driven Combination of Matching Methods. In: *ISWC International Workshop on Ontology Matching (OM) CEUR Workshop Proceedings*, vol. 551, pp. 49–60 (2009)
- [5] Cruz, I.F., Stroe, C., Caci, M., Caimi, F., Palmonari, M., Antonelli, F.P., Keles, U.C.: Using AgreementMaker to Align Ontologies for OAEI 2010. In: *ISWC International Workshop on Ontology Matching (OM). CEUR Workshop Proceedings*, vol. 689, pp. 118–125 (2010)
- [6] Cruz, I.F., Stroe, C., Pesquita, C., Couto, F., Cross, V.: Biomedical Ontology Matching Using the AgreementMaker System (Software Demonstration). In: *International Conference on Biomedical Ontology (ICBO). CEUR Workshop Proceedings*, vol. 833, pp. 290–291 (2011)

- [7] Duan, S., Fokoue, A., Srinivas, K.: One Size Does Not Fit All: Customizing Ontology Alignment Using User Feedback. 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. 177–192. Springer, Heidelberg (2010)
- [8] Eckert, K., Meilicke, C., Stuckenschmidt, H.: Improving Ontology Matching Using Meta-level Learning. 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. 158–172. Springer, Heidelberg (2009)
- [9] Ehrig, M., Staab, S., Sure, Y.: Bootstrapping Ontology Alignment Methods with APFEL. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 186–200. Springer, Heidelberg (2005)
- [10] Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2007)
- [11] Gardner, M.W., Dorling, S.R.: Artificial Neural Networks (the Multilayer Perceptron)—a Review of Applications in the Atmospheric Sciences. *Atmospheric Environment* 32(14-15), 2627–2636 (1998)
- [12] John, G.H., Langley, P.: Estimating Continuous Distributions in Bayesian Classifiers. In: Conference on Uncertainty in Artificial Intelligence (UAI), pp. 338–345 (1995)
- [13] Hariri, B.B., Sayyadi, H., Abolhassani, H., Esmaili, K.S.: Combining Ontology Alignment Metrics Using the Data Mining Techniques. In: ECAI International Workshop on Context and Ontologies, pp. 65–67 (2006)
- [14] Jain, P., Hitzler, P., Sheth, A.P., Verma, K., Yeh, P.Z.: Ontology Alignment for Linked Open 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. 402–417. Springer, Heidelberg (2010)
- [15] Köpcke, H., Rahm, E.: Training Selection for Tuning Entity Matching. In: International Workshop on Quality in Databases and Management of Uncertain Data (QDB/MUD), pp. 3–12 (2008)
- [16] Lee, Y., Sayyadian, M., Doan, A., Rosenthal, A.: eTuner: Tuning Schema Matching Software Using Synthetic Scenarios. *VLDB Journal* 16(1), 97–122 (2007)
- [17] Li, J., Tang, J., Li, Y., Luo, Q.: RiMOM: A Dynamic Multistrategy Ontology Alignment Framework. *IEEE Transactions on Data and Knowledge Engineering* 21(8), 1218–1232 (2009)
- [18] Marie, A., Gal, A.: Boosting Schema Matchers. In: Meersman, R., Tari, Z. (eds.) OTM 2008. LNCS, vol. 5331, pp. 283–300. Springer, Heidelberg (2008)
- [19] Marsland, S.: Machine Learning: an Algorithmic Perspective. Chapman & Hall/CRC (2009)
- [20] Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.: Introduction to WordNet: an On-line Lexical Database. *International Journal of Lexicography* 3(4), 235–244 (1990)
- [21] Mochol, M., Jentzsch, A.: Towards a Rule-Based Matcher Selection. In: Gangemi, A., Euzenat, J. (eds.) EKAW 2008. LNCS (LNAI), vol. 5268, pp. 109–119. Springer, Heidelberg (2008)
- [22] Mochol, M., Jentzsch, A., Euzenat, J.: Applying an Analytic Method for Matching Approach Selection. In: ISWC International Workshop on Ontology Matching (OM). CEUR Workshop Proceedings, vol. 225 (2006)
- [23] Ngo, D., Bellahsene, Z., Coletta, R.: YAM++ Results for OAEI 2011. In: ISWC International Workshop on Ontology Matching (OM). CEUR Workshop Proceedings, vol. 814, pp. 228–235 (2011)
- [24] Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo (1993)

- [25] Shi, F., Li, J., Tang, J., Xie, G., Li, H.: Actively Learning Ontology Matching via User Interaction. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 585–600. Springer, Heidelberg (2009)
- [26] Tartir, S., Budak Arpinar, I., Moore, M., Sheth, A.P., Aleman-Meza, B.: OntoQA: Metric-Based Ontology Quality Analysis. In: IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources, vol. 9, pp. 45–53 (2005)

TELIX: An RDF-Based Model for Linguistic Annotation

Emilio Rubiera¹, Luis Polo¹, Diego Berrueta¹, and Adil El Ghali²

¹ Fundación CTIC

`name.surname@fundacionctic.org`

² IBM CAS France

`adil.elghali@fr.ibm.com`

Abstract. This paper proposes the application of the RDF framework to the representation of linguistic annotations. We argue that RDF is a suitable data model to capture multiple annotations on the same text segment, and to integrate multiple layers of annotations. As well as using RDF for this purpose, the main contribution of the paper is an OWL ontology, called TELIX (Text Encoding and Linguistic Information eXchange), which models annotation content. This ontology builds on the SKOS XL vocabulary, a W3C standard for representation of lexical entities as RDF graphs. We extend SKOS XL in order to capture lexical relations between words (e.g., synonymy), as well as to support word sense disambiguation, morphological features and syntactic analysis, among others. In addition, a formal mapping of feature structures to RDF graphs is defined, enabling complex composition of linguistic entities. Finally, the paper also suggests the use of RDFa as a convenient syntax that combines source texts and linguistic annotations in the same file.

1 Introduction

A linguistic annotation is a descriptive or analytic mark dealing with raw language data extracted from texts or any other kind of recording. A large and heterogeneous number of linguistic features can be involved. Typically linguistic annotations include part-of-speech tagging, syntactic segmentation, morphological analysis, co-references marks, phonetic segmentation, prosodic phrasing and discourse structures, among others.

There is an increasing need for vendors to interchange linguistic information and annotations, as well as the source documents they refer to, among different software tools. Text analysis and information acquisition often require incremental steps with associated intermediate results. Moreover, tools and organizations make use of shared resources such as thesauri or annotated corpus. Clearly, appropriate standards that support this open information interchange are necessary. These standards must provide the means to model and serialize the information as files.

In [16], the following requirements for a linguistic annotation framework are identified: expressive adequacy, media independence, semantic adequacy, uniformity, openness, extensibility, human readability, processability and consistency.

We postulate that the RDF framework features these properties, and therefore constitutes a solid foundation. RDF graphs make use of custom vocabularies defined by ontologies. Therefore, we introduce TELIX, a lightweight ontology that provides comprehensive coverage of linguistic annotations and builds on previous resources, such as feature structures and SKOS concept schemes. TELIX takes advantage of the RDF/OWL expressive power and is compatible with legacy materials. Moreover, translations from traditional linguistic annotation formats to RDF is possible as shown in [6].

This paper is organized as follows. The next section revises the RDF framework and discusses the advantages of representing linguistic annotations as RDF graphs. The main contribution of this paper is TELIX, an OWL ontology which is described in Section 3. Details on how to embed linguistic annotations using the RDFa syntax are given in Section 4. Finally, Section 5 examines previous initiatives, and conclusions and connections to ongoing, similar proposals are presented in Section 6.

2 Linguistic Annotations as RDF Graphs

Effective and open communication between independent parties requires an agreement on shared standards. This paper proposes the application of the RDF framework to linguistic annotations. We sustain that RDF, in combination with the TELIX ontology which is discussed in the next section, facilitate the exchange of expressive linguistic annotations among software tools.

RDF is the W3C recommended framework for making resource descriptions available on the web. RDF graphs can be authored, stored, and web-published. They can also be queried by means of the SPARQL query language, which also defines a web service protocol (SPARQL endpoints) to enable queries on remote graphs. In fact, RDF graphs capturing the linguistic annotations of a given text can be located anywhere on the web, and retrieved as needed, as long as a simple set of principles known as “linked data” are adopted [14]. RDF graphs can be split into many interlinked files, or combined into a single one. Using the RDFa syntax, it is even possible to combine the RDF graph and the source text document into a single file, as it will be discussed in Section 4. This flexibility satisfies several exchange scenario requirements. For instance, a reduced number of files eases management, minimizes the risk of inconsistencies and simplifies internal references. On the other hand, some scenarios demand fine-grained separation of aspects and annotation layers into multiple files.

One of the advantages of RDF is its ability integrate multiple annotation layers in the same framework. The heterogeneity of linguistic analysis is reconciled thanks to the versatility of the RDF graph-based data model. For instance, a single RDF graph may include both syntactic and discourse structures without conflict or interference. Moreover, uniform identifiers (URIs) make it possible to link linguistic resources across multiple RDF graphs, even if they belong to alternative annotation layers. This gluing power is a notable advantage over other annotation formats, such as the XML-based alternatives.

The introduction of URIs as a means to identify and make reference to structures, and particularly text fragments, is a notable departure from more traditional techniques based on positions and offsets, i.e., counting the number of preceding structures or characters. These location-based references are sometimes multi-dimensional (e.g., “the token starting at character 23 of sentence 8”), and are dependent on assumptions about text segmentation. For instance, it is implicit that the reference producer and consumer have previously agreed on the sentence segmentation algorithm. Sometimes, these references are unreliable due to text encoding divergences, e.g., the number of bytes used to represent non-ASCII characters and line breaks are an historical source of interoperability issues. Resolving location-based references is computationally expensive because it requires repeating the segmentation of the text. Moreover, location-based references are extremely sensitive to changes in the source document: even slight modifications of the text render the references invalid, forcing a recalculation. URIs do not present any of these issues, and can be used to make unambiguous, maintainable and easy to resolve references to structures.

3 TELIX, an Ontology of Linguistic Information

This section introduces TELIX (Text Encoding and Linguistic Information eXchange), an OWL vocabulary designed to permit the representation linguistic information as RDF graphs. It extends SKOS XL, which allows capturing lexical entities as RDF resources, and it overcomes SKOS limitations of expressiveness. TELIX introduces a number of classes and properties to provide natural language acquisition and extraction tools with interchangeable, multilingual lexical resources such as dictionaries or thesauri, as well as representing the outcomes of text analyses, i.e., annotations content. The reader is invited to read the TELIX specification [11] where complete details about the ontology and modeling decisions are provided.

The TELIX namespace is <http://purl.org/telix/ns#>, although for the sake of brevity, in this paper it is assumed to be the default namespace. The namespace URL uses HTTP content negotiation to redirect to the OWL specification or the HTML documentation. The OWL file can be downloaded from <http://purl.org/telix/telix.owl>

3.1 Text Segmentation

TELIX provides machinery to describe a given piece of text. More precisely, TELIX types the document and the corpus with Dublin Core concepts, namely `dctype:Text` and `dctype:Collection`. A set of textual units to grasp the text structure are also defined: `Section`, `Paragraph`, `Sentence` and `Token`. In addition, inspired by LAF annotations [16], TELIX introduces the superconcept `Segment` to capture any fragment of choice that does not match any of the former. These entities can be combined by annotation tools to segment the primary data.

Multiple segmentations of the same textual fragment are possible. For example, tokens are assumed to be auxiliary entities, defined as contiguous string of alphabetic or numerical characters and separated by separation characters such as whitespaces. Note that punctuation is included in the resulting list of tokens in some parsers and discourse analysis tools. Tokens enable tools to provide divergent lexical understandings of a given piece of text. The same bunch of words can be interpreted differently depending on the focus of the analysis. Consider, for instance, the string “maraging steel”, composed of two tokens (t_1 , t_2). It can be seen either as a composition of two single words “[maraging $_{t_1}$] [steel $_{t_2}$]” or as the collocation “[maraging $_{t_1}$ steel $_{t_2}$]” making the whole string a single lexical unit. These lexical issues are critical when dealing with technical terminology, where term boundaries are fuzzy and disputed. TELIX does not enforce a concrete analysis regarding the lexical disambiguation of texts. The concept `Token` provides a free-focus word segmentation of the text, over which upper segmentation layers (such as term identification) can be built.

More refined textual units, such as title, chapter, itemized list, etc., are not part of TELIX. However, as TELIX is an OWL ontology, it can be extended or combined with other ontologies to fit the specific requirements of a particular application.

3.2 Words and Senses in RDF

The W3C SKOS vocabulary [3] is a lightweight OWL ontology created to facilitate web-oriented taxonomies and thesauri. SKOS supports multilingual information by means of three labeling properties: `skos:prefLabel`, `skos:altLabel` and `skos:hiddenLabel`. However, SKOS lacks the expressiveness to fully describe the labels of concepts as it treats them as RDF literals, which cannot play the role of subject in a triple. Labels cannot then be further detailed nor linked to other labels. To overcome this limitation, W3C recommends SKOS eXtension for Labels (SKOS XL) [20], which is an extension to the vanilla SKOS vocabulary that provides mechanisms for identifying and describing lexical entities. Fundamentally, a new class `skosxl:Label` is introduced to deal with lexical entities as RDF resources. A label in SKOS-XL can be either a single word or a multiword expression [22], such as a collocation.

As natural languages are inherently complex, in linguistics, three complementary entities, with different natures and properties, are distinguished: concepts, lexemes and words (occurrences). For the sake of precision, TELIX refines SKOS XL to seamlessly meet the requirements of linguistic text analysis as explained above. Table 1 sums up the TELIX proposal. Note that the entities are described according to their bound to languages and texts.

Concepts, abstract ideas formed in mind, can be extrapolated, to a greater or lesser extent, from language to language. Concepts are also called *meanings*. In order to be treated as single resources, concepts are represented in TELIX both as instances of `skos:Concept` or elements of a domain ontology.

A language captures these concepts by means of words or sets of words. However, a distinction must be made between the physical realization of the words

Table 1. TELIX proposal to represent concepts, lexemes and words (occurrences)

Linguistic Entity	OWL class	Language-dependence	Text-dependence
Concept	<code>skos:Concept</code>	-	-
Lexeme	<code>skosxl:Label</code>	+	-
Word (occurrence)	<code>LabelOccurrence</code>	+	+

(in a speech or written down in a document) and their abstract interpretation. The latter is often called a **lexeme**, i.e., a meaningful linguistic unit belonging to the vocabulary of a language. A lexeme is merely a theoretical notion not traceable in actual textual samples. TELIX put lexemes at the same level as `skosxl:Label`, thus restricting the interpretation of lexical entities under the specification. Canonical forms of lexemes (*lemmas*) are the values of the property `skosxl:literalForm` in a SKOS XL terminology. TELIX also refines the generic property `skosxl:labelRelation` in order to capture lexical relationships between lexemes, for instance synonymy (for *synsets*), homonymy and hyponymy. The connections between lexemes and concepts are borrowed from the relationship between SKOS and SKOS XL.

Finally, **words** occur in natural language materialization, typically being part of a communicative act. For instance, a text piece such as “Bronze, an alloy of copper and tin, was one of the first alloys discovered” contains 14 words including the words “alloy” and “alloys” (i.e., two occurrences of the lexeme “alloy”). This interpretation of a word is not a theoretical entity, but real, concrete realization of the natural language. Therefore, TELIX introduces the class `LabelOccurrence` to capture physical realizations¹. For each term identified in a given sentence, a new RDF resource (typed as a `LabelOccurrence`) is created and linked to its corresponding `skosxl:Label` by means of the property `realizes`. Morpho-syntactic information of word forms is captured by RDF-based feature structures, as described below and illustrated in the example of Figure 2.

The word sense annotation of the term occurrence may involve connecting the lexical entities (`skosxl:Label` instances) to domain ontologies and SKOS thesauri concepts. Note that, in this situation, links between the occurrence and its word senses are carried out by the indirect mediation of `skosxl:Label` entities. Complementarily, TELIX also provides the property `sense` to directly relate the label occurrence and its meaning. Thus, it is not necessary to go from word occurrences to word senses through word definitions. Figure 1 illustrates both mediated and direct links. Note that the resource used to disambiguate the lexical entities is drawn from DBpedia [2].

3.3 Linguistic Feature Structures

Lexical items descriptions can be enriched by means of feature structures that capture their grammatical properties. Feature structures are a recursive

¹ At the time of writing, TELIX only covers written realizations in texts. In the future, it is planned to extend the ontology to capture other forms of word materializations, such as sounds/phonemes.

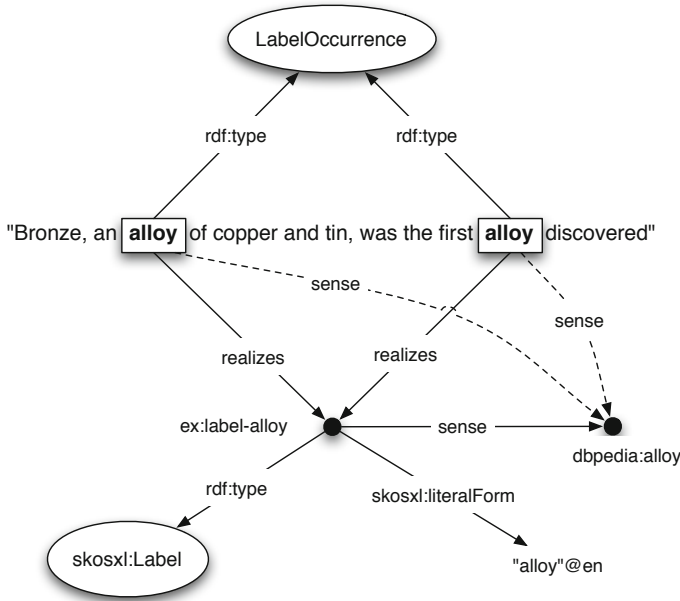


Fig. 1. This figure shows the links between label occurrences in a text and lexical entities in the lexicon. In addition, the property *sense* attaches meanings to both words and their occurrences.

representation of linguistic information. They are sets of feature-value pairs, where features stand for atomic grammatical properties, and values are either atomic symbols or other feature structures.

A common mathematical representation of feature structures is a direct acyclic graph [21][18]. Let us assume there are two disjoint, finite sets \mathcal{F} of *feature names* and \mathcal{S} of *species names*. In linguistics, \mathcal{S} interprets the sorts (types) of entities according to a grammatical theory (for instance, cases, verbs, types of phrases, etc.), and \mathcal{F} interprets the grammatical features, such as agreement or tenses of verbs. A *feature graph* is an ordered triple $\mathcal{G} = \langle \mathcal{V}, \phi, \psi \rangle$, where: \mathcal{V} is the set of vertices of \mathcal{G} ; ϕ is a function which maps each feature name $f \in \mathcal{F}$ to a partial function $\phi(f)$ from $\mathcal{V} \times \mathcal{V}$; and ψ is a function which maps each species name $s \in \mathcal{S}$ a subset of \mathcal{V} .

We define directed graph as an ordered pair $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, where \mathcal{V} is again the set of vertices (nodes) of the graph and \mathcal{E} a subset of $\mathcal{V} \times \mathcal{V}$, called the edges of \mathcal{G} . The node n_j is *accessible* from n_i , where $n_i, n_j \in \mathcal{V}$, if there is a path (finite sequence of edges) from n_i to n_j . A feature graph is then reinterpreted as a directed graph. Given the feature graph $\mathcal{G} = \langle \mathcal{V}, \phi, \psi \rangle$, the relation $\mathcal{E}_\phi \subseteq \mathcal{V} \times \mathcal{V}$ is defined as the union of the interpretation of the feature names: $\bigcup \{ \phi(f) : f \in \mathcal{F} \}$. Therefore $\langle \mathcal{V}, \mathcal{E}_\phi \rangle$ is a directed graph.

Finally, we define a *feature structure* as a quadruple $\langle n_0, \mathcal{V}, \phi, \psi \rangle$, where: $\langle \mathcal{V}, \phi, \psi \rangle$ is a feature graph; and $\langle n_0, \mathcal{V}, \mathcal{E}_\phi \rangle$ is a directed graph where n_0 is the root of the structure, as every node $n_k \in \mathcal{V}$ is accessible from it.

3.4 Feature Structures as RDF Graphs

Given the definitions above, the translation of feature structures to RDF is straightforward because both data structures are directed graphs. The RDF language distinguishes three sets of disjoint syntactic entities. Let U denote the set of *URI references* and Bl the set of *blank nodes*, i.e., variables. Let L be the set of *literals*, i.e., data values such as floats or strings. An RDF graph G is a set of *triples*, where the tuple $\langle s p o \rangle \in (U \cup Bl) \times U \times (U \cup Bl \cup L)$ is called an RDF triple. In the tuple, s is the subject, p the predicate and o the object.

The vocabulary of an RDF graph G , denoted by $V(G)$, is the set of names that occur as subject, predicate or object of a triple in G [13]. A *simple interpretation* I of a vocabulary V is a 6-tuple $I = (R_I, P_I, E_I, S_I, L_I, LV_I)$, where R_I is a non-empty set, called the set of resources or the universe, P_I is the set of properties, LV_I is the set of literal values, which is a subset of R_I that contains at least all the plain literals in V , and where E_I , S_I and L_I are functions:

- $E_I: P_I \rightarrow \wp(R_I \times R_I)$, where $\wp(X)$ denotes the power set of the set X . The function E_I defines the extension of a property as a set of pairs of resources.
- $S_I: (V \cap U) \rightarrow (R_I \cup P_I)$ defines the interpretation of URI references.
- $L_I: (V \cap L) \rightarrow (L \cup R_I)$ defines the interpretation of literals.

If $t = \langle s p o \rangle$ is an RDF triple, then a simple interpretation I of a vocabulary V is said to satisfy t if $s, p, o \in V$, $I(p) \in P_I$, and $(I(s), I(o)) \in E_I(I(p))$. We extend this interpretation with *Class* and *type* from the RDFS vocabulary, where $I(\text{type}) \in P_i$, and the set C_I of classes of I is defined as: $C_I = \{c \in R_I : (c, I(\text{Class})) \in E_I(I(\text{type}))\}$. Thus, given a feature structure $\mathcal{G} = \langle n_0, \mathcal{V}, \phi, \psi \rangle$ and an augmented RDF vocabulary $V \cup \{\text{type}, \text{Class}\}$, we define a *mapping* $\pi = \langle \pi_1, \pi_2, \pi_3, \pi_4, \pi_5 \rangle$ to transform \mathcal{G} to an RDF graph G given an interpretation I as follows:

- $\forall n \in \mathcal{V} : \pi_1(n) = \eta$, where $\eta \in U$ and $I(\eta) \in R_I$. This mapping function introduces a new node η in G . In other words, π_1 is a labeling function, providing URIs for each node of the feature structure. Note that the root node n_0 is also included in the transformation.
- $\forall f \in \mathcal{F} : \pi_2(f) = p$, where $p \in U$, and $I(p) \in P_I$. Observe that p is a property defined in the TELIX vocabulary.
- $\forall s \in \mathcal{S} : \pi_3(s) = c$, where $c \in U$, and $I(c) \in (R_I \cap C_I)$. Note that c is a class defined in the TELIX vocabulary.
- $\pi_4(\phi) = E_I$, where for each pair $\langle n_i, n_j \rangle \in \phi(f)$ in \mathcal{G} , the application of π_4 returns $(I(\pi_1(n_i)), I(\pi_1(n_j))) \in E_I(I(\pi_2(f))) = (I(\eta_i), I(\eta_j)) \in E_I(I(p))$ in G . This mapping retains the feature names interpretation of \mathcal{G} , $\phi : \mathcal{F} \mapsto \mathcal{V} \times \mathcal{V}$, in G . In other words, this mapping is an isomorphism between the

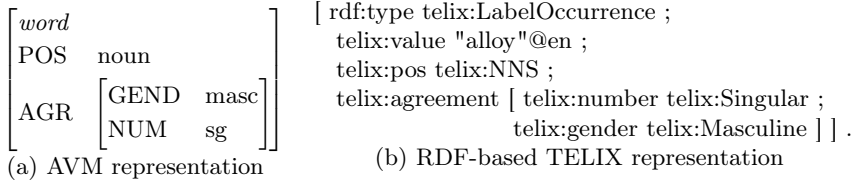


Fig. 2. Feature structure of the word “alloy”

original feature structure and the resultant RDF graph, where each pair of nodes of \mathcal{G} , connected by a grammatical feature, is translated to an RDF triple.

- $\pi_5(\psi) = CE_I$, where CE_I is the class *extension function* of I , defined by: $CE_I(c) = \{a \in R_I : (a, c) \in E_I(I(\textit{type}))\}$ in \mathcal{G} . Therefore, for each $n_i \in \psi(s)$, the application of π_5 returns $(I(\pi_1(n_i)), I(\pi_3(s)) \in E_I(I(\textit{type})) = (I(\eta_i), I(c)) \in E_I(I(\textit{type}))$ in G . This mapping retains the species names interpretation of \mathcal{G} (types of nodes) in the RDF graph: $\psi : \mathcal{S} \mapsto \mathcal{V}$.

TELIX introduces a core set of concepts and properties that represent \mathcal{S} and \mathcal{F} respectively. This vocabulary permits the application of mappings π_1 , π_2 and π_3 to a given linguistic theory in order to express feature structures using the RDF data model. Part of these grammatical features refer to morpho-syntactic information, such as number (**telix:number**), person (**telix:person**), gender (**telix:gender**) and tense (**telix:tense**) for agreement, or part-of-speech information (**telix:pos**). Furthermore, TELIX provides collections of values over which these properties range. Some of these collections are based on existing linguistic classifications. This is the case of part-of-speech tags, adapted from the list used in the Penn Treebank Project (which can be extended to deal with other languages). The purpose is to facilitate the exchange and integration of linguistic information by reusing resources widely-adopted by the community.

As an example, Figure 2 illustrates the outcome produced by the application of π mappings to a feature structure. The left-side of the Figure is the feature structure, represented here with the graphical Attribute-Value Matrix notation, which captures the grammatical analysis of the word “alloy”. The right-side shows the resulting RDF graph (in N3 syntax).

TELIX also covers other aspects of text analysis, such as syntactic and discourse structures. RDF translations are provided for both constituent parse trees (as partially illustrated in Figure 4) and dependency graphs. Furthermore, discursive entities are defined. With regards to referring expressions, TELIX introduces properties: **correfers**, **antecedent** and **anaphora**, to express different coreference nuances. Rhetorical relations are also supplied to represent the underlying structure at the discourse level of a given text.

It is worth mentioning that although TELIX provides machinery to represent feature structures as RDF graphs, it does not cover complex constraints or feature structures operations (such as unification). In other words, TELIX permits

rich, interchangeable descriptions of linguistic entities, but does not represent grammars such as HPSG, LFG or other linguistic theory. To this end, more expressive, rule-based formalisms on top of TELIX are necessary.

3.5 Annotation Support

TELIX also introduces the concept **Annotation** to capture linguistic annotations as entities within the model. This makes it possible to describe the annotation itself, for instance by means of the Dublin Core vocabulary to express authorship (`dc:creator`), date (`dc:date`) and the source of the annotation (`dc:source`).

Another strong point of RDF is that it natively supports both the combination of complementary linguistic analysis and multi-authored annotations over the same text fragment, as segments are univocally identified by URIs. An annotation needs only to link a given fragment in order to describe it.

Firstly, RDF facilitates the amalgam of multiple annotation layers. For instance, Figure 4 contains an example of an annotation which merges the parse tree of the nominal phrase “the first alloy” enriched with morpho-syntactic and lexical information of the terminal node “alloy” (Figure 2). Both analyses, even coming from different NLP analyzers and platforms, are easily integrated in the same graph. In the case of the syntactic analysis, the parsing of constituents was performed by the Stanford parser (Figure 3) and subsequently translated to an RDF graph using the TELIX vocabulary. Although for the sake of readability, relations that capture the order of the terminal nodes of the parse tree in Figure 4 have been omitted, TELIX introduces the property `telix:precedes` with this purpose. Moreover, the new property navigation feature defined in SPARQL 1.1 [24] is particularly useful for querying this kind of tree-shaped graph structure. For instance, the property path `telix:childNodes*` traverses the edges of the derived parse tree.

Secondly, TELIX takes advantage of *named graphs* [24,5] to handle multi-authored annotations over the same piece of text. If analyses performed by different NLP tools at a given linguistic layer (such as word sense disambiguation) are mixed into a single annotation or graph, tracking them is practically impossible. Therefore, a set \mathcal{A} of annotations is defined as an RDF dataset, where: $\mathcal{A} = \{G_0, \langle u_1, G_1 \rangle, \dots, \langle u_n, G_n \rangle\}$. G_0 and each G_i are graphs, and u_1, \dots, u_n are distinct IRIs. The pairs $\langle u_i, G_i \rangle$ are named graphs, where G_i an RDF graph

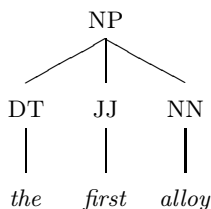


Fig. 3. Constituents parsing performed by the Stanford parser

```
[ rdf:type telix:NounPhrase ;
  telix:childNode [ rdf:type telix:DT ;
    telix:childNode [ rdf:type telix:LabelOccurrence ;
      telix:value "the"@en ] ] ;
  telix:childNode [ rdf:type telix:JJ ;
    telix:childNode [ rdf:type telix:LabelOccurrence ;
      telix:value "first"@en ] ] ;
  telix:childNode [ rdf:type telix:NN ;
    telix:childNode [ rdf:type telix:LabelOccurrence ;
      telix:value "alloy"@en ;
      telix:realizes ex:label-alloy ;
      telix:sense dbpedia:alloy ;
      telix:agreement [ telix:number telix:Singular ;
        telix:gender telix:Masculine ] ] ] ] .
```

Fig. 4. RDF graph combining multiple annotation layers

containing the annotation data (potentially featuring multiple layers) and u_i the name of the annotation. G_0 is the default graph of the RDF dataset and it provides the definition attached to each annotation u_i , where $u_i \in \mathbf{Annotation}$.

4 Adding Annotations to Structured Text Documents Using RDFa

Among the RDF syntaxes mentioned in Section 2, RDFa [4] seems particularly well suited for adding linguistic annotations to source documents. RDFa is a W3C standard that can weave RDF graphs within a host mark-up language, typically (X)HTML. One of the advantages of RDFa is that it is possible to combine the original content (the source text) and the resulting segmented corpus, in a single file. As the number of files to be exchanged decreases, benefits appear in the form of easier management, simplified change tracking and consistency maintenance. Moreover, RDFa does not alter the appearance of a document, which preserves its original styling information.

The essence of RDFa is a set of coordinated attributes that are attached to elements of the XML infoset [25]. In terms that are more familiar to HTML developers, this means that attributes are appended to opening *tags* such as `<p>`. It is not possible to annotate units of text that are not delimited by tags, but this is not an issue because ad hoc mark-up can be added to the document, usually by means of invisible `<div>` and `` tags (for units larger and smaller than paragraphs, respectively).

4.1 Document Preparation

Prior to annotation, some preparation is often required. Firstly, the original document must be converted into XHTML. Plain text documents can be trivially upgraded to XHTML documents by simply enclosing the text in an XHTML

```

<body about="#document1" typeof="dctype:Text" datatype=""
  property="telix:value" rel="dct:hasPart" id="document1">
<p about="#para1" typeof="telix:Paragraph" datatype=""
  property="telix:value" rel="dct:hasPart" id="para1">
  <span about="#sent11" typeof="telix:Sentence" datatype=""
    property="telix:value" id="sent11">Steel is an alloy that consists
    mostly of iron and has a carbon content between 0.2% and 2.1% by weight,
    depending on the grade.</span>
  <span about="#sent12" typeof="telix:Sentence" datatype=""
    property="telix:value" id="sent12">Carbon is the most common alloying
    material for iron, but various other alloying elements are used, such as
    manganese, chromium, vanadium, and tungsten.</span>
</p>
</body>

```

Fig. 5. Document body annotated with RDFa attributes. Namespaces are omitted

template. Then, additional `<p>`, `<div>` and `` tags must be introduced as required until the document mark-up structure matches the text segmentation. At this point, the document looks like the example in Figure 5. Note that sentences are delimited by `` tags nested inside the `<p>` tags. Tag nesting captures multiple levels of structure (sections, paragraphs, sentences, parts of sentences, words. . .). Due to the tree-based model of XML documents, it is not possible to build structures that overlap without one being contained within the other. However, the RDFa document can be combined with other RDF documents sharing the same URIs

Note that tags make sentence segmentation explicit in the document. Therefore, it is no longer necessary that the producer and the consumer of the document implement the same segmentation algorithm in order to unequivocally agree on the scope of each sentence. As the boundaries of each sentence are explicitly marked in the document, and the number of sentences is unambiguous, location-based references have a solid ground to build on. TELIX supports location-based references as a fallback option to be backward compatible with legacy tools.

4.2 In-place Document Annotation

The simplest RDFa annotation involves attributes `about` and `typeof`, which introduce identifiers (URIs) for structures of the document and specify their type (as explained in Section 3.1).

Even if the relationship between the text structure and the text content is implicit due to the mark-up nesting, RDFa parsers do not automatically convert it into RDF triples. To this effect, the pair of attributes `property="telix:value"` and `datatype=""` must be added, as will be shown in the final example.

The hierarchy of structures implicit by the mark-up nesting (e.g., the sentences are contained in the paragraphs) must be explicitly named in order to

be captured in the RDF graph. A pair of inverse properties (`dct:hasPart` and `dct:isPartOf`) can be used for this purpose, in combination with the `rel` and `rev` attributes of RDFa. In fact, just one of them is enough to express the hierarchy, and the choice depends only on syntactic convenience. Figure 5 contains the final result of annotating the document body.

4.3 Separate Annotations

Although it is a convenient choice in many scenarios, there are a number of reasons that may render RDFa unsuitable to embed complex RDF graphs into source documents. These reasons include RDFa limitations regarding annotation of non-contiguous text fragments, its inability to capture coexistent but divergent text segmentations, and its verbosity (when compared to other RDF syntaxes). Moreover, some scenarios simply require separating life-cycles for the source document and its annotations.

For these scenarios, we suggested decoupling the linguistic annotations and the source document. A basic, uncontroversial and shared segmentation of the source text may be added using RDFa, while the other annotation layers are kept in separate files (possibly using other RDF syntaxes). Even in this case, annotations can still univocally point to the corresponding text fragments by means of their URIs identifiers.

5 Previous Work

TELIX builds on the experience of a chain of proposed languages, ontologies and frameworks that have previously addressed the effective exchange of textual resources in order to facilitate automated processing.

Most notably, TEI (Text Encoding Initiative) [1] is an XML-based encoding scheme that is specifically designed for facilitating the interchange of data among research groups using different programs or application software. TEI provides an exhaustive analysis about how to encode the structure of textual sources, feature structures or graphics and tables in XML format. Although TEI defines a very detailed specification of linguistic annotations, its XML syntax does not facilitate the integration of heterogeneous layers of annotations. Since most of the linguistic workflows (UIMA, Gate, etc.) rely on multiple modules covering different layers of annotations, an RDF-based format to represent the annotations, such as TELIX, is more suitable to be used by these systems. More concretely, TELIX offers some advantages over TEI, derived from the more flexible nature of RDF graphs with respect to XML trees, permitting the description of several layers of annotations linked to the source document.

GrAF [17] is another graph-based format for linguistic annotation encoding, although it does not rely on RDF but on an ad-hoc XML syntax. As it is based on RDF, our proposal elegantly solves the graph merging problem. Moreover, GrAF annotations can be translated into RDF [6], thus existing GrAF annotations can easily be translated into TELIX. Furthermore, another advantage of using the RDF framework is the availability of a standard query language,

namely SPARQL. Both GrAF and TELIX are motivated by LAF (Linguistic Annotation Framework [16]), which identifies the requirements and defines the main decision principles for a common annotation format. TELIX supports integrated multilayered annotations and enables multiple annotations to the same text fragment. However, although TELIX includes support for stand-off annotations (based on offsets), it discourages them. Instead TELIX proposes a combination of URI identifiers and RDFa annotations in mark-up documents.

LMF (Lexical Markup Framework, ISO 24613:2008) is a model of lexical resources. It is suitable for the levels of annotations that are attached to a lexical entry, but not for syntactic annotations in the case of non-lexicalized grammars. Being an XML format, it lacks the advantages of RDF discussed in this paper.

Other ontologies have been proposed to represent linguistic information. The most noteworthy one is GOLD [12], which is specified in OWL and provides a vocabulary to represent natural languages. GOLD is designed as a refined extension of the SUMO ontology. TELIX and GOLD have some resemblances, although they diverge in their goals: TELIX is more annotation-oriented, while GOLD aims to provide the means to describe natural languages formally.

The OLiA ontologies [7] provide OWL vocabularies to describe diverse linguistic phenomena, from terminology and grammatical information to discourse structures. TELIX and OLiA take different approaches to similar goals, in particular regarding constituent-based syntactic trees. TELIX also contributes a formal foundation to translate feature structures in RDF.

The Lemon model [19] proposes its own vocabulary to model words and senses in RDF. However, TELIX prefers to take advantage of (and extend) the SKOS framework for modeling lexical entities, as discussed in Section 3.2. Regarding WordNet [23], there are some overlaps with TELIX regarding lexical entities treatment. Nevertheless, they are potentially complementary, e.g., WordNets synsets can be used as values of TELIX's `sense` property.

6 Conclusions

This paper proposes the use of the RDF framework in combination with an ontology (TELIX) for linguistic annotation. Despite the considerable body of previous and current proposals with similar goals, the authors believe that TELIX sits in a previously unoccupied space because of its comprehensiveness and its orientation to the information exchange on the web of data. A comprehensive evaluation of TELIX with respect to the related works is planned for the coming months. Among the works that are concurrently being developed and that are closely tied to TELIX, POWLA [8] is a recent proposal of an OWL/DL formalization to represent linguistic corpora based on the abstract model PAULA [10], a complete and complex XML model of linguistic annotations. Another ongoing initiative is NIF [13], also based on OLiA. NIF and TELIX coincide in the use of URIs to univocally identify text fragments, enabling the handling of multiply-anchored annotations over them. The main difference between NIF and TELIX is that the latter offers a corpus level that is not provided by the former.

TELIX is driven by the goal to provide usable, expressive linguistic annotations. Being able to query and to reason on these annotations is a key requirement for a widely-adopted annotation format. Moreover, the success of SKOS as a web-oriented standard for concept schemes inspires confidence in the possibility of a web-oriented standard for linguistic annotations.

For TELIX to be successful, it must be embraced by the community and implemented by NLP tools. The authors are working in both fronts. Firstly, plans are in place to submit the specification of TELIX to W3C, either as a Member Submission or as a contribution to the Ontology-Lexica Community Group. Secondly, prototype implementations of TELIX in some NLP tools, such as an UIMA workflow for rules extraction [9], are being produced in the context of the ONTORULE project [11].

Acknowledgment. The work described in this paper has been partially supported by the European Commission under ONTORULE Project (FP7-ICT-2008-3, project reference 231875).

References

1. TEI P5: Guidelines for Electronic Text Encoding and Interchange. Technical report, TEI Consortium (2012), <http://www.tei-c.org/Guidelines/P5/>
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: 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.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
3. Bechhofer, S., Miles, A.: SKOS Simple Knowledge Organization System Reference. W3C recommendation, W3C (August 2009), <http://www.w3.org/TR/2009/REC-skos-reference-20090818/>
4. Birbeck, M., Adida, B.: RDFa primer. W3C note, W3C (October 2008), <http://www.w3.org/TR/2008/NOTE-xhtml-rdfa-primer-20081014/>
5. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, Provenance and Trust. In: WWW 2005: Proceedings of the 14th International Conference on World Wide Web, pp. 613–622. ACM, New York (2005)
6. Cassidy, S.: An RDF realisation of LAF in the DADA annotation server. In: Proceedings of ISA-5, Hong Kong (2010)
7. Chiarcos, C.: An Ontology of Linguistic Annotations. LDV Forum 23(1), 1–16 (2008)
8. Chiarcos, C.: POWLA: Modeling Linguistic Corpora in OWL/DL. In: Simperl, E., et al. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 225–239. Springer, Heidelberg (2012)
9. Derdek, S., El Ghali, A.: Une chaîne UIMA pour l'analyse de documents de réglementation. In: Proceeding of SOS 2011, Brest, France (2011)
10. Dipper, S.: XML-based stand-off representation and exploitation of multi-level linguistic annotation. In: Proceedings of Berliner XML Tage 2005 (BXML 2005), pp. 39–50 (2005)
11. Lévy, F. (ed.): D1.4 Interactive ontology and policy acquisition tools. Technical report, Ontorule project (2011), <http://ontorule-project.eu/>

12. Farrar, S., Langendoen, T.: A Linguistic Ontology for the Semantic Web. *GLOT International* 7, 95–100 (2003)
13. Hayes, P.: RDF semantics. W3C recommendation. W3C (February 2004), <http://www.w3.org/TR/2004/REC-rdf-mt-20040210/>
14. Heath, T., Bizer, C.: *Linked Data: Evolving the Web into a Global Data Space*, 1st edn. Morgan & Claypool (2011)
15. Hellmann, S.: NLP Interchange Format (NIF) 1.0 specification, <http://nlp2rdf.org/nif-1-0>
16. Ide, N., Romary, L.: International Standard for a Linguistic Annotation Framework. *Journal of Natural Language Engineering* 10 (2004)
17. Ide, N., Suderman, K.: GrAF: a graph-based format for linguistic annotations. In: *Proceedings of the Linguistic Annotation Workshop, LAW 2007*, Stroudsburg, PA, USA, pp. 1–8. Association for Computational Linguistics (2007)
18. King, P.J.: *An Expanded Logical Formalism for Head-Driven Phrase Structure Grammar*. Arbeitspapiere des SFB 340 (1994)
19. McCrae, J., Spohr, D., Cimiano, P.: Linking Lexical Resources and Ontologies on the Semantic Web with Lemon. 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. 245–259. Springer, Heidelberg (2011)
20. Miles, A., Bechhofer, S.: SKOS Simple Knowledge Organization System eXtension for Labels (SKOS-XL). W3C recommendation, W3C (August 2009), <http://www.w3.org/TR/2009/REC-skos-reference-20090818/skos-xl.html>
21. Pollard, C.: *Lectures on the Foundations of HPSG*. Technical report, Unpublished manuscript: Ohio State University (1997), <http://www-csli.stanford.edu/~sag/L221a/cp-1ec-notes.pdf>
22. Sag, I.A., Baldwin, T., Bond, F., Copestake, A., Flickinger, D.: Multiword Expressions: A Pain in the Neck for NLP. In: Gelbukh, A. (ed.) *CICLing 2002*. LNCS, vol. 2276, pp. 1–15. Springer, Heidelberg (2002)
23. Schreiber, G., van Assem, M., Gangemi, A.: RDF/OWL representation of WordNet. W3C working draft, W3C (June 2006), <http://www.w3.org/TR/2006/WD-wordnet-rdf-20060619/>
24. Seaborne, A., Harris, S.: SPARQL 1.1 query. W3C working draft, W3C (October 2009), <http://www.w3.org/TR/2009/WD-sparql11-query-20091022/>
25. Tobin, R., Cowan, J.: XML information set, W3C recommendation, W3C, 2nd edn. (February 2004), <http://www.w3.org/TR/2004/REC-xml-infoset-20040204>

LODifier: Generating Linked Data from Unstructured Text

Isabelle Augenstein, Sebastian Padó, and Sebastian Rudolph

Department of Computational Linguistics, Universität Heidelberg, DE
Institute AIFB, Karlsruhe Institute of Technology, DE
{augenste,pado}@cl.uni-heidelberg.de, rudolph@kit.edu

Abstract. The automated extraction of information from text and its transformation into a formal description is an important goal in both Semantic Web research and computational linguistics. The extracted information can be used for a variety of tasks such as ontology generation, question answering and information retrieval. LODifier is an approach that combines deep semantic analysis with named entity recognition, word sense disambiguation and controlled Semantic Web vocabularies in order to extract named entities and relations between them from text and to convert them into an RDF representation which is linked to DBpedia and WordNet. We present the architecture of our tool and discuss design decisions made. An evaluation of the tool on a story link detection task gives clear evidence of its practical potential.

1 Introduction

The term Linked Data (*LD*) stands for a new paradigm of representing information on the Web in a way that enables the global integration of data and information in order to achieve unprecedented search and querying capabilities. This represents an important step towards the realization of the Semantic Web vision. At the core of the LD methodology is a set of principles and best practices describing how to publish structured information on the Web. In recent years these recommendations have been adopted by an increasing number of data providers ranging from public institutions to commercial entities, thereby creating a distributed yet interlinked global information repository.

The formalism underlying this “Web of Linked Data” is the Resource Description Framework (*RDF*) which encodes structured information as a directed labelled graph. Hence, in order to publish information as Linked Data, an appropriate graph-based representation of it has to be defined and created. While this task is of minor difficulty and can be easily automatized if the original information is already structured (as, e.g., in databases), the creation of an adequate RDF representation for unstructured sources, particularly textual input, constitutes a challenging task and has not yet been solved to a satisfactory degree.

Most current approaches [7,19,16,6] that deal with the creation of RDF from plain text fall into the categories of relation extraction or ontology learning. Typically, these approaches process textual input very selectively, that is, they scan the text for linguistic

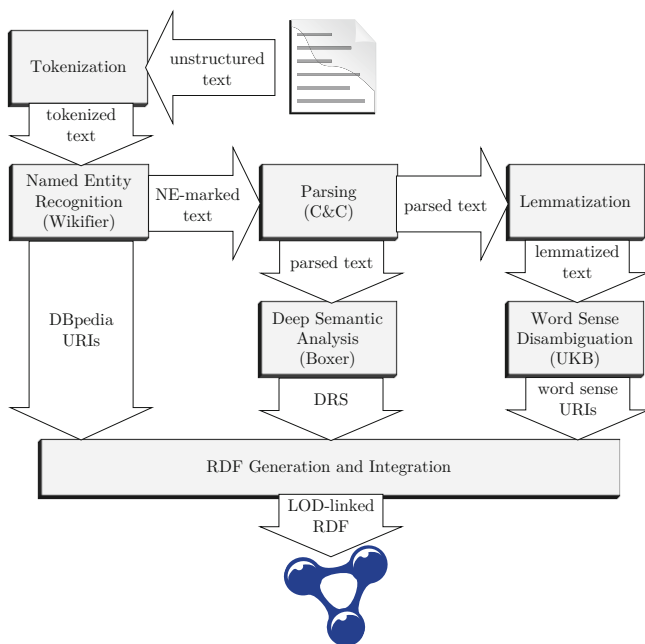


Fig. 1. The architecture of LODifier

patterns that realize a small number of pre-specified types of information (e.g., *is-CEO-of* relations). This strategy is oriented toward a high precision of the extracted structured information and certainly adequate if the result of the extraction process is meant to be used for accumulation of large sets of factual knowledge of a predefined form.

In contrast to these approaches, we propose a strategy which aims at translating the textual input *in its entirety* into a structural RDF representation. We aim at open-domain scenarios in which no a-priori schema for the information to be extracted is available. Applying our method to a document similarity task we demonstrate that it is indeed both practical and beneficial to retain the richness of the full text as long as possible.

Our system, *LODifier*, employs robust techniques from natural language processing (NLP) including named entity recognition (*NER*), word sense disambiguation (*WSD*) and deep semantic analysis. The RDF output is embedded in the Linked Open Data (*LOD*) cloud by using vocabulary from DBpedia and WordNet 3.0.

Plan of the Paper. Section 2 begins by sketching the architecture of the system. Section 3 presents an evaluation of *LODifier* on a document similarity task. After discussing related work in Section 4, we conclude in Section 5.

2 The System

This section describes the resources and algorithms used to build *LODifier*. Figure 1 shows the architecture of the system. After tokenization, mentions of entities in the

input text are recognized using the NER system *Wikifier* [18] and mapped onto DBpedia URIs. Relations between these entities are detected using the statistical parser *C&C* and the semantics construction toolkit *Boxer* [8], which generates discourse representation structures (*DRSs*) [14]. Thereafter, the text is lemmatized and words are disambiguated with the WSD tool *UKB* [1] to get WordNet mappings. The RDF graph is then created by further processing the Boxer DRS output, transforming it into triples. Finally, it is enriched with the DBpedia URIs (to link its entities to the LOD cloud) and the WordNet sense URIs (to do the same for the relations). The following subsections provide details on the individual processing steps.

2.1 Recognizing Named Entities

The first step is to identify mentioned individuals. They are recognized using the NER system *Wikifier* [18] that enriches English plain text with Wikipedia links. If *Wikifier* finds a named entity, it is substituted by the name of the corresponding English Wikipedia page. Applied to the test sentence

The New York Times reported that John McCarthy died. He invented the programming language LISP.

Wikifier recognizes the named entities and generates the output

```
[[The New York Times]] reported that [[John McCarthy (computer scientist)|John McCarthy]] died. He invented the [[Programming language|programming language]] [[Lisp (programming language)|Lisp]].
```

To disambiguate the Wikipedia links, *Wikifier* employs a machine learning approach that uses the links between Wikipedia articles as training data. Since the links between Wikipedia articles are manually created by Wikipedia editors, the training data consists of highly reliable disambiguation choices.

Note that the *Boxer* system itself also performs a NER. We employ *Wikifier* to increase NER coverage and, most notably, to obtain links to the LOD cloud via DBpedia URIs.

2.2 Linking DBpedia URIs to Recognized Named Entities

The next step is to generate DBpedia URIs out of the *Wikifier* output and link those DBpedia URIs to previously introduced *Boxer* classes.

DBpedia [4] is a large, freely available domain-independent multilingual ontology extracted from Wikipedia, comprising Wikipedia page names, infobox templates, categorization information, images, geo-coordinates and links to external webpages. *DBpedia* contains links to various data sets including *FOAF*, *Geonames* and *WordNet*.

We exploit the fact that every Wikipedia page has a corresponding DBpedia page, which allows for a straightforward conversion of Wikipedia URLs to DBpedia URIs.

2.3 Recognizing Relations

Next, relations between the entities are determined. This is done by the parser C&C and the Boxer system developed by Curran, Clark and Bos [8].

The C&C parser first tags input words with parts of speech from the Penn Treebank tagset. It then constructs parse trees in the combinatorial categorial grammar (CCG) paradigm. In addition, C&C contains a named entity recognizer that distinguishes between ten different named entity types: organization (*org*), person (*per*), title (*ttl*), quotation (*quo*), location (*loc*), first name (*fst*), surname (*sur*), URL (*url*), e-mail (*ema*) and unknown name (*nam*). The parser is rather robust for a “deep” natural language processing tool, with precision and recall scores well above 80%. The C&C output for our example is displayed in Fig. 2. It forms a derivation tree in which each non-terminal is labelled with the CCG rule that was used to construct it (e.g., *fa* for ‘forward application’) as well as its CCG category. The terminals, labelled *t*, provide information about the words’ CCG categories, forms and lemmas and parts of speech (in this order). The two last elements of each terminal specify information on shallow chunks and Named Entities, using IOB (inside-outside-begin) notation, a common format for representing non-hierarchical chunks. An IOB label starting with I-, like I-NP, indicates that a given word is *inside* an NP chunk. The label 0 means that the word is not part of any chunk. For example, the only Named Entity recognized in the first sentence in *John_McCarthy* (a PER) [1].

Boxer builds on the output of the statistical parser C&C and produces *discourse representation structures* (DRSs, cf. [14]). DRSs model the meaning of texts in terms of the relevant entities (*discourse referents*) and the relations between them (*conditions*). Figure 3 shows the DRSs for our example. Discourse referents are shown above the dotted lines and conditions below.

Discourse referents are introduced by new noun phrases or events and are, from a logical standpoint, essentially variables. For previously introduced discourse referents, Boxer attempts to resolve anaphora by either binding them to previously introduced discourse referents or accommodating them. A condition, which is described by a unary or binary predicate, is created for every relation found between discourse referents. Unary relations (also referred to as *classes*) are introduced by nouns, verbs, adverbs and adjectives, these are e.g., *person*, *event* or *topic*. Binary relations are introduced by prepositions and verb roles, e.g., *agent*, *patient* or *theme*. As the example shows, conditions can embed DRSs themselves. Such conditions are called complex conditions and are used to specify logical dependencies between partial propositions: *disjunction*, *implication*, *negation*, *necessity*, *possibility*.

Note that the DRS conditions only use unary and binary relations. Therefore, DRSs are structurally very similar to RDF, and can hence serve as a convenient intermediate data structure for converting text into RDF.

2.4 Assigning RDF WordNet URIs to Boxer Relations

Our first candidate for a target vocabulary for linking Boxer relations onto Linked Open Data entities was DBpedia. DBpedia contains about 44.000 different property types

¹ IOB supports labels of type B- to mark the first word in a chunk, but this is not used by Boxer.


```

ccg(1,
  rp(s:dcl,
    ba(s:dcl,
      lx(np, n,
        t(n, 'The_New_York_Times', 'The_New_York_Times', 'NNS', 'I-NP', 'O')),
        fa(s:dcl\np,
          t((s:dcl\np)/s:em, 'reported', 'report', 'VBD', 'I-VP', 'O'),
          fa(s:em,
            t(s:em/s:dcl, 'that', 'that', 'IN', 'I-SBAR', 'O'),
            ba(s:dcl,
              lx(np, n,
                t(n, 'John_McCarthy', 'John_McCarthy', 'NNP', 'I-NP', 'I-PER')),
                t(s:dcl\np, 'died', 'die', 'VBD', 'I-VP', 'O'))))),
          t(period, '.', '.', '.', 'O', 'O')))).

ccg(2,
  rp(s:dcl,
    ba(s:dcl,
      t(np, 'He', 'he', 'PRP', 'I-NP', 'O'),
      fa(s:dcl\np,
        t((s:dcl\np)/np, 'invented', 'invent', 'VBD', 'I-VP', 'O'),
        fa(np:nb,
          t(np:nb/n, 'the', 'the', 'DT', 'I-NP', 'O'),
          fa(n,
            t(n/n, 'programming_language', 'programming_language', 'NN', 'I-NP', 'O'),
            t(n, 'LISP', 'LISP', 'NNP', 'I-NP', 'O'))))),
      t(period, '.', '.', '.', 'O', 'O')))).

```

Fig. 2. C&C output for the example sentences

created by extracting properties from infoboxes and templates within Wikipedia articles. The *Raw Infobox Property Definition Set* consists of a URI definition for each property as well as a label. However, this property set turned out to be much too restricted to cover all the relations identified by Boxer.

Therefore, we decided to map Boxer relations onto RDF WordNet class types instead.

WordNet [10] is a large-scale lexical database for English. Its current version contains more than 155.000 words (nouns, verbs, adjectives and adverbs), grouped into sets of synonyms, which are called *synsets*. Ambiguous words belong to several synsets (one per *word sense*). The synsets are linked to other synsets by *conceptual relations*. Synsets contain *glosses* (short definitions) and short example sentences. *RDF WordNet* [3] is a Linked Data version of WordNet. For each word it provides one URI for each word sense. To map instances of words onto URIs the words have to be disambiguated.

For word sense disambiguation (WSD), we apply *UKB* [1], an unsupervised graph-based WSD tool, to all our input words, but focus on the results for words which have given rise to relations in the Boxer output. We use the disambiguated RDF WordNet URIs as the Linked Data hooks for these relations.

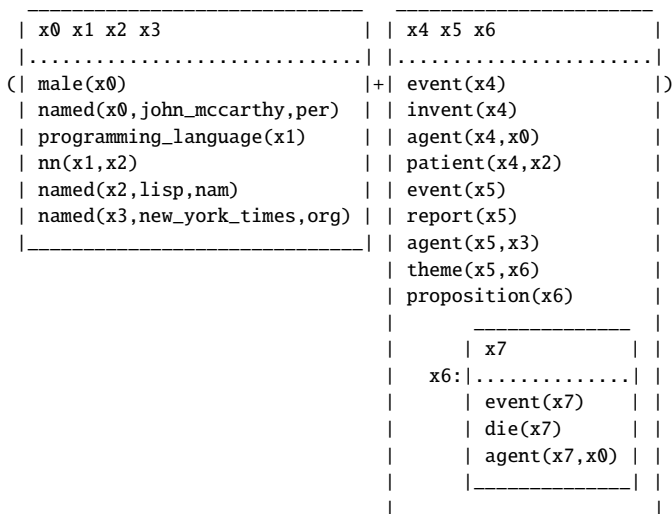


Fig. 3. Discourse representation structure generated for the example sentences

2.5 Generating an RDF Graph

Finally, we construct an RDF graph. Our first step is to define URIs for the predicate and relation types provided by Boxer. In this manner, we distinguish between predicate and relation types which come from a *closed* class (*event*, *agent*, etc.), and the *open* classes of predicate and relation types that represent words (*programming_language*, *die*, etc.). The second step is a translation of discourse referents and DRS conditions (unary and binary relations) into RDF triples according to the following strategy:

- For each discourse referent, a blank node (bnode) is introduced. If it has been recognized as a NE by Boxer, we assign an URI from the `ne:` namespace to it via the property `drsc:named`. If an according DBpedia URI could be identified via Wikifier, we link the blank node to the according DBpedia URI via `owl:sameAs`.
- The assignment of a Boxer class (that is, a unary predicate) to a discourse referent is expressed by an RDF typing statement which associates a class URI to the discourse referent’s bnode. For closed-class relations (like *event*), the class URI comes from the predefined vocabulary (using the namespace `drsc:`), for relations from the open class we use the appropriate word sense URI extracted from WordNet via UKB (in the namespace `wn30:`) or create a URL (in the namespace `class:`).
- A closed-class binary relation between two discourse referents (e.g., *agent*) is expressed by an “ordinary” RDF triple with the referents’ bnodes as subject and object, and using the corresponding URI from the closed-class Boxer vocabulary namespace `drsrel:`. For open-class relations, the namespace `rel:` is used instead.
- Finally, we may encounter embedded DRSs, possibly related by complex conditions expressing logical (*disjunction*, *implication*, *negation*) or modal (*necessity*,

```

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix reify: <http://www.aifb.kit.edu/web/LODifier/reify#> .
@prefix ne: <http://www.aifb.kit.edu/web/LODifier/ne#> .
@prefix drsclass: <http://www.aifb.kit.edu/web/LODifier/drsclass#> .
@prefix class: <http://www.aifb.kit.edu/web/LODifier/class#> .
@prefix drsrel: <http://www.aifb.kit.edu/web/LODifier/drsrel#> .
@prefix dbpedia: <http://dbpedia.org/resource/> .
@prefix wn30: <http://purl.org/vocabularies/princeton/wn30/> .
_:var0x0 drsclass:named ne:john_mccarthy ;
  rdf:type drsclass:male , foaf:Person ;
  owl:sameAs dbpedia:John_McCarthy_(computer_scientist) .
_:var0x1 rdf:type class:programming_language ;
  owl:sameAs dbpedia:Programming_language .
_:var0x2 drsrel:nn _:var0x1 .
_:var0x2 drsclass:named ne:lisp ;
  owl:sameAs dbpedia:Lisp_(programming_language) .
_:var0x3 drsclass:named ne:the_new_york_times ;
  owl:sameAs dbpedia:The_New_York_Times .
_:var0x4 rdf:type drsclass:event , wn30:wordsense-invent-verb-2 .
  drsrel:agent _:var0x0 ; drsrel:patient _:var0x2 .
_:var0x5 rdf:type drsclass:event , wn30:wordsense-report-verb-3 ;
  drsrel:agent _:var0x3 ; drsrel:theme _:var0x6 .
_:var0x6 rdf:type drsclass:proposition , reify:proposition , reify:conjunction ;
  reify:conjunct [ rdf:subject _:var0x7 ;
    rdf:predicate rdf:type ;
    rdf:object drsclass:event . ]
  reify:conjunct [ rdf:subject _:var0x7 ;
    rdf:predicate rdf:type ;
    rdf:object wn30:wordsense-die-verb-1 . ]
  reify:conjunct [ rdf:subject _:var0x7 ;
    rdf:predicate drsrel:agent ;
    rdf:object _:var0x0 . ]

```

Fig. 4. LODifier output for the test sentences

possibility) operators. We recursively reify the RDF representations of these subordinate DRSs by using the predefined RDF reification vocabulary (consisting of the property URIs `rdf:subject`, `rdf:predicate`, and `rdf:object`, for an introduction to reification in RDF see e.g. [13], Section 2.5.2) [4]. The logical or modal dependencies between these sub-DRSs are then expressed by means of additional fixed vocabulary in the namespace `reify:`. This results in flat RDF output for nested DRSs.

The result of applying this strategy to our example text is shown in Figure 4.

² To obtain output that adheres to the current W3C RDF specification and is entirely supported by standard-compliant tools, we refrain from using named graphs or quads to encode nested DRS.

3 Automatic Evaluation: Story Link Detection

3.1 Task and Setup

In our evaluation we use LODifier to assess document similarity. More specifically, we consider a *story link detection* task, part of the topic detection and tracking (TDT) family of tasks. It is defined as “[...] the problem of deciding whether two randomly selected stories discuss the same news topic” [2].

We use a subset of the TDT-2 benchmark dataset. TDT-2 consists of a total of almost 84.000 documents from the year 1998, drawn from newspapers, radio news, and television news in English, Arabic and Mandarin. Each document is manually assigned to one of a set of 100 topics. Each topic also comes with one “seed” story that is presumed to be representative for the topic.

We follow the general lead of the original TDT-2 benchmark evaluation schema. We focus on English as a language and newspapers as the source since LODifier can currently only deal with English text and presumably degrades on potentially noisy automatic radio and TV transcripts. We therefore restrict our attention to the 19 topics whose seed story was an English newspaper article. For each topic, we pair the seed story with all other articles of this topic that met our constraints. However, since the distribution of topics over documents is very skewed and we want to avoid undue influence of very large topics, we restrict the number of document pairs for each topic to 50. This results in a total of 183 *positive* document pairs, an average of 11.2 document pairs per topic. We then sample the same number of *negative* document pairs from the dataset by pairing each document with the seed document from a different topic. The total number of document pairs is 366 with an equal positive/negative distribution.

We approach the task by defining various document similarity measures *sim*. We assume that this similarity is a direct indicator of relatedness, which leads to a very simple classification procedure for document pairs *dp*, given a threshold θ :

$$\text{class}(dp, \theta) = \begin{cases} \text{positive} & \text{if } \text{sim}(dp) \geq \theta \\ \text{negative} & \text{if } \text{sim}(dp) < \theta \end{cases}$$

Thus, document pairs are predicted to describe the same topic exactly if they have a similarity of θ or more.

The parameter θ is usually determined with supervised learning. We randomly split our dataset k times (we use $k=1000$) into equally-sized training and testing sets. For each split, we compute an optimal decision boundary $\hat{\theta}$ as the threshold which predicts the training set as well as possible. More precisely, we choose $\hat{\theta}$ so that its distance to wrongly classified training document pairs is minimized. Formally, $\hat{\theta}$ is defined as follows: Let pos_{train} and neg_{train} be the positive and negative partitions of the training set respectively. Then:

$$\hat{\theta} = \arg \min_{\theta} \left[\sum_{dp \in pos_{train}} \min(0, \text{sim}(dp) - \theta)^2 + \sum_{dp \in neg_{train}} \min(0, \theta - \text{sim}(dp))^2 \right]$$

We can then compute the accuracy of $\hat{\theta}$ on the current split’s test set, consisting of the positive and negative partitions pos_{test} and neg_{test} , as follows:

$$acc_{\hat{\theta}} = \frac{|\{dp \in pos_{test} \mid sim(dp) \geq \hat{\theta}\}| + |\{dp \in neg_{test} \mid sim(dp) < \hat{\theta}\}|}{|pos_{test}| + |neg_{test}|}$$

After repeating this procedure for all k splits, the final accuracy is computed by averaging the k accuracies.

3.2 Similarity Computation without Structure

As baselines, we consider a number of measures that do not take structural information into account. The first one is a *random baseline*, which performs at 50% in our setup, since the two classes (positive and negative) are balanced. Second, we consider a simple *bag-of-words baseline* which measures the word overlap of two documents in a document pair without any preprocessing. Third, we experiment with a family of *bag-of-URI baselines* that measure the URI overlap of two RDF documents. We experiment with three variants which correspond to different assumptions about the relative importance of various URI classes:

Variante 1 considers all NEs identified by Wikifier and all words successfully disambiguated by UKB (namespaces `dbpedia:` or `wn30:`).

Variante 2 adds all NEs recognized by Boxer (namespace `ne:`).

Variante 3 further adds the URIs all words that were not recognized by either Wikifier, UKB or Boxer (namespace `class:`).

Extended setting. For each of the variants we also construct an extended setting, where the generated RDF graph was enriched by information from DBpedia and WordNet, namely DBpedia categories, WordNet synsets and WordNet senses related to the respective URIs in the generated graph. This setting aims at drawing more information from Linked Open Data into the similarity computation.

3.3 Structurally Informed Similarity Computation

Recall our motivation for using LODifier, namely the intuition that structural RDF information can provide an improvement over the simple comparison of words or URIs. This requires the formulation of structure-aware similarity measures between documents (i.e., RDF graphs). First attempts showed that full-fledged graph similarity measures based on homomorphic or isomorphic subgraphs of arbitrary size or common cliques [21], which are generally NP-complete, are infeasible due to the size of the RDF graphs we consider (up to 19.000 nodes).

We decided to perform a more relaxed structural comparison based on the shortest paths between relevant nodes. This mirrors our intuition that a short path in a RDF graph between two URIs denotes a salient semantic relation between those entities. For such URI pairs, we would expect that they are related by a short path in other documents (graphs) on the same topic as well.

Formally, let G_1 and G_2 be two RDF graphs. We write $\ell(a, b)$ to denote the length of the shortest path between two nodes a and b in a graph G , and let $C_k(G)$ denote the set

of all paths of length $\leq k$ in G , that is, the set of salient relations in G .³ Furthermore, we write $Rel(G)$ for the set of relevant nodes in an RDF graph G . As motivated in Section 3.2, not all URIs are equally relevant and we experiment with different choices. We can now define a family of similarity measures called *path relevance overlap similarity* (proSim):

$$\text{proSim}_{k,Rel,f}(G_1, G_2) = \frac{\sum_{\substack{a,b \in Rel(G_1) \\ (a,b) \in C_k(G_1) \cap C_k(G_2)}} f(\ell(a, b))}{\sum_{\substack{a,b \in Rel(G_1) \\ (a,b) \in C_k(G_1)}} f(\ell(a, b))}$$

In words, the denominator of proSim determines the set of relevant semantic relations Rel in G_1 – modelled as the set of pairs of relevant URIs that are linked by a path of length $\leq k$ – and quantifies them as the sum over a function applied to their path lengths. The numerator does the same for the intersection of the relevant nodes from G_1 and G_2 .

We experiment with three instantiations for the function f . The first one, $\text{proSim}_{\text{cnt}}$, uses $f(\ell) = 1$, that is, just counts the number of paths irrespective of their length. The second one, $\text{proSim}_{\text{len}}$, uses $f(\ell) = 1/\ell$, giving less weight to longer paths. The third one, $\text{proSim}_{\text{sqlen}}$, uses $f(\ell) = 1/\sqrt{\ell}$, discounting long paths less aggressively than $\text{proSim}_{\text{len}}$.

All measures of the proSim family have the range $[0;1]$, where 0 indicates no overlap and 1 perfect overlap. It is deliberately asymmetric: the overlap is determined relative to the paths of G_1 . This reflects our intuitions about the task at hand. For a document to be similar to a seed story, it needs to subsume the seed story but can provide additional, new information on the topic. Thus, the similarity should be maximal whenever $G_1 \subseteq G_2$, which holds for proSim.

3.4 Results

As described in Section 3.2, we experiment with several variants for defining the set of relevant URIs (Variants 1 to 3, both normal and extended). These conditions apply to all bag-of-URI and proSim models.

The results of our evaluation are shown in Table 1. The upper half shows results for similarity measures without structural knowledge. At 63%, the bag-of-words baseline clearly outperforms the random baseline (50%). It is in turn outperformed by almost all bag-of-URI baselines, which yield accuracies of up to 76.4%. Regarding parameter choice, we see the best results for Variant 3, the most inclusive definition of relevant URIs (cf. Section 3.2). The URI baseline also gains substantially from the extended setting, which takes further Linked Open Data relations into account.

Moving over to the structural measures, proSim, we see that all parametrizations of proSim perform consistently above the baselines. Regarding parameters, we again see a consistent improvement for Variant 3 over Variants 1 and 2. In contrast, the performance is relatively robust with respect to the path length cutoff k or the inclusion of further Linked Open Data (extended setting).

³ Shortest paths can be computed efficiently using Dijkstra’s algorithm [9]. We exclude paths across “typing” relations such as `event` which would establish short paths between every pair of event nodes (cf. Figure 3) and drown out meaningful paths.

Table 1. Accuracy on Story Link Detection Task

Model	normal	extended
Similarity measures without structural knowledge		
Random Baseline	50.0	–
Bag of Words	63.0	–
Bag of URIs (Variant 1)	61.6	75.1
Bag of URIs (Variant 2)	70.6	76.0
Bag of URIs (Variant 3)	73.4	76.4
Similarity measures with structural knowledge		
proSim _{cnt} (k=6, Variant 1)	79.0	78.9
proSim _{cnt} (k=6, Variant 2)	80.3	80.3
proSim _{cnt} (k=6, Variant 3)	81.6	81.6
proSim _{cnt} (k=8, Variant 1)	77.7	77.6
proSim _{cnt} (k=8, Variant 2)	79.2	79.0
proSim _{cnt} (k=8, Variant 3)	82.1	81.9
proSim _{len} (k=6, Variant 3)	81.5	81.4
proSim _{len} (k=8, Variant 3)	80.3	80.1
proSim _{len} (k=10, Variant 3)	80.0	79.8
proSim _{slen} (k=6, Variant 3)	80.4	80.4
proSim _{slen} (k=8, Variant 3)	81.1	80.9
proSim _{slen} (k=10, Variant 3)	80.5	80.4

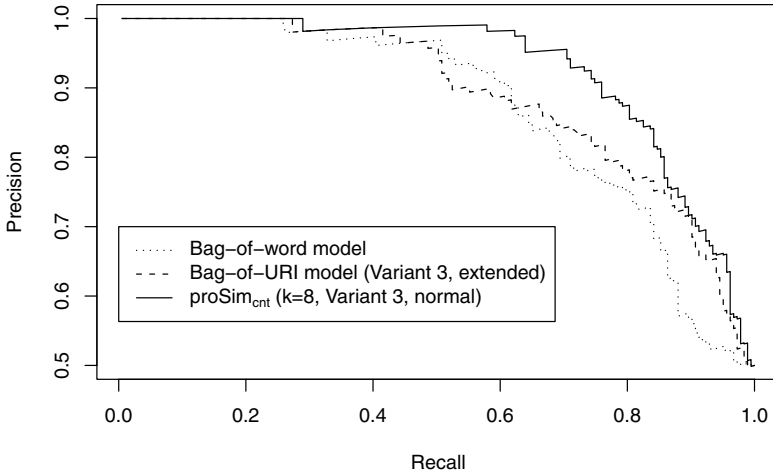


Fig. 5. Precision-Recall-plot for best Story Link Detection models

Moving from $\text{proSim}_{\text{cnt}}$ to more involved similarity functions, we decided to concentrate on Variant 3 which showed the best results. However, neither $\text{proSim}_{\text{len}}$ nor $\text{proSim}_{\text{sqlen}}$ yielded considerably different results: On this dataset, performance plateaued between 80% and 82% accuracy. The numerically best result was 82.1%, obtained for $\text{proSim}_{\text{cnt}}$, Variant 3, with $k=8$ in the normal setting. The difference to the best bag-of-URI model is more than 5% (absolute) accuracy. We tested the difference for statistical significance with bootstrap resampling [15] and obtained a negative result. We believe, however, that this outcome is mainly due to the small size of our current test set.

To illustrate the behavior of proSim family in more detail, Figure 5 shows a precision-recall evaluation for the most promising models for each class. This mode of evaluation still assumes the same decision rule (cf. Section 3.1) but does not optimize the threshold on training data: Rather, it varies the threshold between 0 and 1 and computes at each point the precision and recall for the positive (“same topic”) class. At $\theta=0$, the recall is 1, but the precision only 0.5, due to the design of the dataset. At $\theta=1$, the precision is (close to) 1, but the recall (close to) 0. The closer the curve to the top right corner (high precision and recall) the better. This evaluation provides more detail on the performance of the similarity measures across the range of document similarities.

Figure 5 demonstrates that the precision-recall evaluation corresponds well to the accuracy-based results reported above. The bag-of-word model is outperformed by the bag-of-URI model for almost all values of θ , which is in turn outperformed by $\text{proSim}_{\text{cnt}}$. The plot shows that the particular benefit provided by the structural models is the ability to retain a high precision for much higher recall rates compared to the other models. For example, they show an almost perfect precision for a recall of up to 0.6, where the shallower models have dropped to a precision below 0.90; for recall values between 0.6 and 0.8, the difference in precision remains at about 10% (absolute) before all curves converge for very high recall values.

In sum, we found a consistent numerical improvement for the structural measures compared to the URI measures. More generally, our results indicate that structurally informed measures of graph similarity can deliver practical benefits for applications, even for document-level tasks. Currently, however, our proSim measures do not profit either from more involved weighting schemes or from the inclusion of further Linked Open Data. Progress in this direction will require more research on possible weighting schemes and strategies to select informative features from the range of information present in LOD.

The raw data of all evaluations performed here including the generated RDF graphs is available via <http://www.aifb.kit.edu/web/LODifier>

4 Related Work

There are various approaches to extracting relationships from text. These approaches usually include the annotation of text with named entities and relations and the extraction of those relations. Two approaches that are very similar to LODifier are of [5] and [19]. They both use NER, POS-tagging and parsing to discover named entities and relations between them. The resulting relations are converted to RDF. The disadvantage of these methods is however that they use labelled data as a base for extracting relations, which is not flexible, as labelled data requires manual annotation.

In terms of the pursued goal, that is, processing natural language, converting the result into RDF and possibly exhibiting it as linked (open) data, LODifier shares the underlying motivation with the NLP2RDF framework⁴. The latter provides a generic and flexible framework of how to represent any kind of NLP processing result in RDF. While the current version of LODifier is a stand-alone tool not resting on this framework, a future integration might improve its interoperability and reusability further.

The Pythia system [20] which is targeted at natural language question answering in information systems also employs deep semantic analysis on posed questions in order to come up with a translation into SPARQL queries which are then posed against RDF stores. Pythia does, however, presume the existence of a lexicon specifying how lexical expressions are to be mapped to RDF entities of the queried data source. Thereby, the approach is inherently domain-specific, whereas we aim at an open domain setting where no a-priori lexical mappings or specific schematic information is available.

The *AsKNet* system [12] is aimed at automatically creating a semantic network. Thereby, the processing strategy is similar to ours: the system uses C&C and Boxer to extract semantic relations. To decide which nodes refer to the same entities, similarity scores are computed based on spreading activation and nodes are then mapped together. An approach building on *AsKNet* comes from [22]. They use *AsKNet* to build a semantic network based on relations between concepts instead of relations between named entities as already present in *AsKNet*. The resulting graph is then converted to RDF. *AsKNet* and LODifier differ in the way they disambiguate named entities. LODifier uses NER and WSD methods before generating RDF triples and describes the entities and relations using DBpedia and WordNet URIs whereas *AsKNet* first generates semantic structures from text and then tries to map nodes and edges together based on similarity. Moreover, the graph output of the latter is not interlinked with other data sources. This is one of the key features of LODifier, and we feel that we have only scratched the surface regarding the benefit of interlinking.

5 Conclusion and Outlook

Much current work in text processing makes exclusive use of shallow methods, either statistical or pattern-based, and makes up for their limitations by the redundancy in large data collections. Their main criticism towards deeper processing are the lack of robustness and efficiency.

In this paper, we have argued that (lack of) robustness is not a knock-out argument. We have presented LODifier, a proof-of-concept implementation which converts open-domain unstructured natural language into Linked Data. The system incorporates several well-established NLP methods: named entities are recognized using NER; normalized relations are extracted by parsing the text and performing deep semantic analysis; WSD helps identifying the proper senses for event types. The meaning of a document is finally consolidated into an RDF graph whose nodes are connected to the broad-coverage Linked Data vocabularies DBpedia and WordNet.

The central benefits that LODifier provides are (a) abstracting away from linguistic surface variation such as lexical or syntactic choice; (b) making explicit structural

⁴ <http://nlp2rdf.org/about>

information as a semantic graph; and (c) linking up the concepts and relations in the input to the LOD cloud. These benefits provide types of additional information for subsequent processing steps, which are generally not provided by “shallow” approaches. Concretely, we have demonstrated that the LODifier representations improve topic detection over competitive shallow models by using a document similarity measure that takes semantic structure into account. More generally, we believe that methods like ours are suitable whenever there is little data, for example, in domain-specific settings.

A few of the design decisions made for the RDF output may not be adequate for all conceivable applications of LODifier. The use of blank nodes is known to bring about computational complications, and in certain cases it is beneficial to Skolemize them by URIs e.g. using MD5 hashes. Employing `owl:sameAs` to link discourse referents to their DBpedia counterparts might lead to unwanted logical ramifications in case the RDF output is flawed. Hence, we will provide a way to configure LODifier to produce RDF in an encoding that meets the specific requirements of the use case.

A current shortcoming of LODifier is its pipeline architecture which treats the modules as independent so that errors cannot be recovered. We will consider joint inference methods to find a globally most coherent analysis [11]. Regarding structural similarity measures, we have only scratched the surface of the possibilities. More involved graph matching procedures remain a challenge due to efficiency reasons; however, this is an area of active research [17].

Possible applications of LODifier are manifold. It could be used to extract DBpedia relation instances from textual resources, provided a mapping from WordNet entities to DBpedia relations is given. Moreover, our system could also be applied for hybrid search, that is, integrated search over structured and non-structured sources. In such a setting, the role of LODifier would be to “pre-process” unstructured information source (off-line) into a representation that matches the structured information sources. This would reduce on-line search to the application of structured query processing techniques to a unified dataset. Presuming that a good accuracy at the semantic micro-level can be achieved, our method could also prove valuable in the domain of question answering. In that case, LODifier could be used to transform structured resources into RDF against which structured (SPARQL) queries generated by question answering systems such as Pythia [20] could be posed.

Acknowledgements. We thank Pia Wojtinnik, Johan Bos and Stephen Clark for stimulating discussions and technical advice about the Boxer system as well as Christina Unger and Philipp Cimiano for pointers to related work. We acknowledge the funding of the European Union (project XLike in FP7).

References

1. Agirre, E., de Lacalle, O.L., Soroa, A.: Knowledge-based WSD and specific domains: performing over supervised WSD. In: Proceedings of the International Joint Conferences on Artificial Intelligence, Pasadena, CA (2009)
2. Allan, J.: Introduction to topic detection and tracking, pp. 1–16. Kluwer Academic Publishers, Norwell (2002)

3. van Assem, M., van Ossenbruggen, J.: WordNet 3.0 in RDF (2011), <http://semanticweb.cs.vu.nl/Lod/wn30/> (Online; accessed July 12, 2011)
4. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. *Web Semant.* 7, 154–165 (2009)
5. Byrne, K., Klein, E.: Automatic extraction of archaeological events from text. In: *Proceedings of Computer Applications and Quantitative Methods in Archaeology*, Williamsburg, VA (2010)
6. Cafarella, M.J., Ré, C., Suciú, D., Etzioni, O., Banko, M.: Structured querying of web text: A technical challenge. In: *Proceedings of the Conference on Innovative Data Systems Research*, Asilomar, CA (2007)
7. Cimiano, P., Völker, J.: Text2Onto. In: Montoyo, A., Muñoz, R., Métais, E. (eds.) *NLDB 2005*. LNCS, vol. 3513, pp. 227–238. Springer, Heidelberg (2005)
8. Curran, J.R., Clark, S., Bos, J.: Linguistically Motivated Large-Scale NLP with C&C and Boxer. In: *Proceedings of the ACL 2007 Demo Session*, pp. 33–36 (2007)
9. Dijkstra, E.W.: A note on two problems in connexion with graphs. *Numerische Mathematik* 1, 269–271 (1959)
10. Fellbaum, C. (ed.): *WordNet: An Electronic Lexical Database*. MIT Press (1998)
11. Finkel, J.R., Manning, C.D.: Hierarchical Joint Learning: Improving Joint Parsing and Named Entity Recognition with Non-Jointly Labeled Data. In: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 720–728 (2010)
12. Harrington, B., Clark, S.: Asknet: Automated semantic knowledge network. In: *Proceedings of the American Association for Artificial Intelligence*, Vancouver, BC, pp. 889–894 (2007)
13. Hitzler, P., Krötzsch, M., Rudolph, S.: *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC (2009)
14. Kamp, H., Reyle, U.: *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. *Studies in Linguistics and Philosophy*, vol. 42. Kluwer, Dordrecht (1993)
15. Koehn, P.: Statistical Significance Tests for Machine Translation Evaluation. In: *Proceedings of Empirical Methods in Natural Language Processing*, Barcelona, Spain, pp. 388–395 (2004)
16. Li, Y., Chu, V., Blohm, S., Zhu, H., Ho, H.: Facilitating pattern discovery for relation extraction with semantic-signature-based clustering. In: *Proceedings of the ACM Conference on Information and Knowledge Management*, pp. 1415–1424 (2011)
17. Lösch, U., Bloehdorn, S., Rettinger, A.: Graph Kernels for RDF Data. In: Simperl, E., et al. (eds.) *ESWC 2012*. LNCS, pp. 134–148. Springer, Heidelberg (2012)
18. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: *Proceedings of the ACM Conference on Information and Knowledge Management* (2008)
19. Ramakrishnan, C., Kochut, K.J., Sheth, A.P.: A Framework for Schema-Driven Relationship Discovery from Unstructured Text. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) *ISWC 2006*. LNCS, vol. 4273, pp. 583–596. Springer, Heidelberg (2006)
20. Unger, C., Cimiano, P.: Pythia: Compositional Meaning Construction for Ontology-Based Question Answering on the Semantic Web. In: Muñoz, R., Montoyo, A., Métais, E. (eds.) *NLDB 2011*. LNCS, vol. 6716, pp. 153–160. Springer, Heidelberg (2011)
21. Valiente, G.: *Algorithms on Trees and Graphs*. Springer, Berlin (2002)
22. Wojtinnik, P.-R., Harrington, B., Rudolph, S., Pulman, S.: Conceptual Knowledge Acquisition Using Automatically Generated Large-Scale Semantic Networks. In: Croitoru, M., Ferré, S., Lukose, D. (eds.) *ICCS 2010*. LNCS, vol. 6208, pp. 203–206. Springer, Heidelberg (2010)

POWLA: Modeling Linguistic Corpora in OWL/DL

Christian Chiarcos

Information Sciences Institute, University of Southern California, 4676 Admiralty
Way # 1001, Marina del Rey, CA 90292
chiarcos@daad-alumni.de

Abstract. This paper describes POWLA, a generic formalism to represent linguistic annotations in an interoperable way by means of OWL/DL. Unlike other approaches in this direction, POWLA is not tied to a specific selection of annotation layers, but it is designed to support any kind of text-oriented annotation.

1 Background

Within the last 30 years, the maturation of language technology and the increasing importance of corpora in linguistic research produced a growing number of linguistic corpora with increasingly diverse annotations. While the earliest annotations focused on part-of-speech and syntax annotation, later NLP research included also on semantic, anaphoric and discourse annotations, and with the rise of statistic MT, a large number of parallel corpora became available. In parallel, specialized technologies were developed to represent these annotations, to perform the annotation task, to query and to visualize them. Yet, the tools and representation formalisms applied were often specific to a particular type of annotation, and they offered limited possibilities to combine information from different annotation layers applied to the same piece of text. Such multi-layer corpora became increasingly popular¹ and, more importantly, they represent a valuable source to study interdependencies between different types of annotation. For example, the development of a semantic parser usually takes a syntactic analysis as its input, and higher levels of linguistic analysis, e.g., coreference resolution or discourse structure, may take both types of information into consideration. Such studies, however, require that all types of annotation applied to a particular document are integrated into a common representation that provides lossless and comfortable access to the linguistic information conveyed in the annotation without requiring too laborious conversion steps in advance.

At the moment, state-of-the-art approaches on corpus interoperability build on standoff-XML [5,26] and relational data bases [12,17]. The underlying data models are, however, graph-based, and this paper pursues the idea that RDF and

¹ For example, parts of the Penn Treebank [29], originally annotated for parts-of-speech and syntax, were later annotated with nominal semantics, semantic roles, time and event semantics, discourse structure and anaphoric coreference [30].

RDF data bases can be applied for the task to represent all possible annotations of a corpus in an interoperable way, to integrate their information without any restrictions (as imposed, for example, by conflicting hierarchies or overlapping segments in an XML-based format), and to provide means to store and to query this information regardless of the annotation layer from which it originates. Using OWL/DL defined data types as the basis of this RDF representation allows to specify and to verify formal constraints on the correct representation of linguistic corpora in RDF. POWLA, the approach described here, formalizes data models for generic linguistic data structures for linguistic corpora as OWL/DL concepts and definitions (POWLA TBox) and represents the data as OWL/DL individuals in RDF (POWLA ABox).

POWLA takes its conceptual point of departure from the assumption that any linguistic annotation can be represented by means of directed graphs [3,26]: Aside from the primary data (text), linguistic annotations consist of three principal components, i.e., segments (spans of text, e.g., a phrase), relations between segments (e.g., dominance relation between two phrases) and annotations that describe different types of segments or relations.

In graph-theoretical terms, segments can be formalized as **nodes**, relations as **directed edges** and annotations as **labels** attached to nodes and/or edges. These structures can then be connected to the primary data by means of pointers. A number of generic formats were proposed on the basis of such a mapping from annotations to graphs, including ATLAS [3] and GrAF [26]. Below, this is illustrated for the PAULA data model, that is underlying the POWLA format. Traditionally, PAULA is serialized as an XML standoff format, it is specifically designed to support multi-layer corpora [12], and it has been successfully applied to develop an NLP pipeline architecture for Text Summarization [35], and for the development of the corpus query engine ANNIS [38]. See Fig. 1 for an example for the mapping of syntax annotations to the PAULA data model.

RDF also formalizes directed (multi-)graphs, so, an RDF linearization of the PAULA data model yields a generic RDF representation of text-based linguistic annotations and corpora in general. The idea underlying POWLA is to represent linguistic annotations by means of RDF, and to employ OWL/DL to define PAULA data types and consistency constraints for these RDF data.

2 POWLA

This section first summarizes the data types in PAULA, then their formalization in POWLA, and then the formalization of linguistic corpora with OWL/DL.

2.1 PAULA Data Types

The data model underlying PAULA is derived from labeled directed acyclic (multi)graphs (DAGs). Its most important data types are thus different types of nodes, edges and labels [14]:

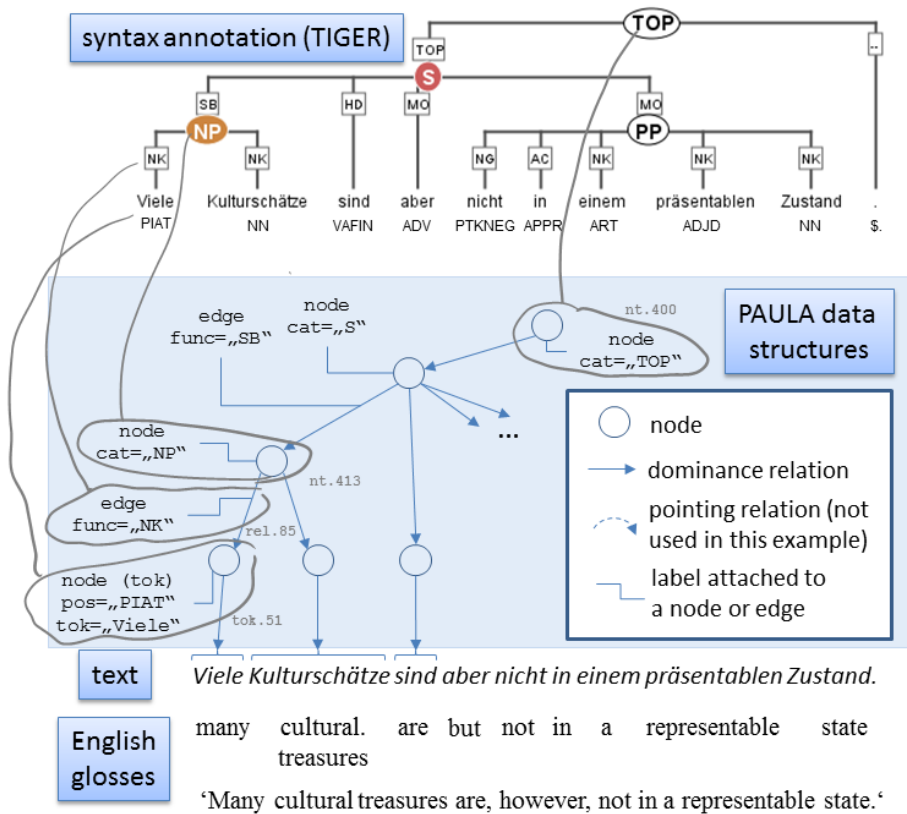


Fig. 1. Using PAULA data structures for constituent syntax (German example sentence taken from the Potsdam Commentary Corpus, [34])

node	(structural units of annotation)
terminal	character spans in the primary data
markable	span of terminals (data structure of flat, layer-based annotations defined e.g., by their position)
struct	hierarchical structures (forming trees or DAGs), establishes parent-child relations between a (parent) struct and child nodes (of any type)
edge	(relational unit of annotation, connecting nodes)
dominance relation	directed edge between a struct and its children, coverage inheritance (see below)
pointing relation	general directed edge, no coverage inheritance
label	(attached to nodes or edges)
feature	linguistic annotation

A unique feature of PAULA as compared to other generic formats is that it introduces a clear distinction between two types of edges that differ with respect to their relationship to the primary data. For hierarchical structures, e.g., phrase structure trees, a notion of **coverage inheritance** is necessary, i.e., the text covered by a child node is always covered by the parent node, as well. In PAULA, such edges are referred to as **dominance relations**. For other kinds of relational annotation, no constraints on the coverage of the elements connected need to be postulated (e.g., anaphoric relations, alignment in parallel corpora, dependency analyses), and source and target of a relation may or may not overlap at all. Edges without coverage inheritance are referred to in PAULA as **pointing relations**. This distinction does not constrain the generic character of PAULA (a general directed graph would just use pointing relations), but it captures a fundamental distinction of linguistic data types. As such, it was essential for the development of convenient means of visualization and querying of PAULA data: For example, the appropriate visualization (hierarchical or relational) within a corpus management system can be chosen on the basis of the data structures alone, and it does not require any external specifications.

Additionally, PAULA includes specifications for the organization of annotations, i.e.

- layer** (grouping together nodes and relations that represent a single annotation layer, in PAULA represented by a namespace prefixed to a label, e.g., `tiger:...` for original TIGER XML)
- document** (or ‘annoset’, grouping together all annotations of one single resource of textual data)
- collection** (an annoset that comprises not only annotations, but also other annosets, e.g., constituting a subcorpus)
- corpus** (a collection not being part of another collection)

Also, layers and documents can be assigned labels. These correspond to metadata, rather than annotations, e.g., date of creation or name of the annotator.

2.2 POWLA TBox: The POWLA Ontology

The POWLA ontology represents a straight-forward implementation of the PAULA data types in OWL/DL. `Node`, `Relation`, `Layer` and `Document` correspond to PAULA node, edge, layer and document, respectively, and they are defined as subclasses of `POWLAElement`.

A `POWLAElement` is anything that can carry a label (property `hasLabel`). For `Document` and `Layer`, these annotations contain metadata (subproperty `hasMetadata`), for `Node` and `Relation`, they contain string values of the linguistic annotation (subproperty `hasAnnotation`). The properties `hasAnnotation` and `hasMetadata` are, however, not to be used directly, but rather, subproperties are to be created for every annotation phenomenon, e.g., `hasPos` for part-of-speech annotation, or `hasCreationDate` for the date of creation.

A **Node** is a **POWLAElement** that covers a (possibly empty) stretch of primary data. It can carry **hasChild** properties (and the inverse **hasParent**) that express coverage inheritance. A **Relation** is another **POWLAElement** that is used for every edge that carries an annotation. The properties **hasSource** and **hasTarget** (resp. the inverse **isSourceOf** and **isTargetOf**) assign a **Relation** source and target node. Dominance relations are relations whose source and target are connected by **hasChild**, pointing relations are relations where source and target are not connected by **hasChild**. It is thus not necessary to distinguish pointing relations and dominance relations as separate concepts in the POWLA ontology.

Two basic subclasses of **Node** are distinguished: A **Terminal** is a **Node** that does not have a **hasChild** property. It corresponds to a PAULA terminal. A **Nonterminal** is a **Node** with at least one **hasChild** property. The differentiation between PAULA struct and markable can be inferred and is therefore not explicitly represented in the ontology: A struct is a **Nonterminal** that has another **Nonterminal** as its child, or that is connected to at least one of its children by means of a (dominance) **Relation**, any other **Nonterminal** corresponds to a PAULA markable. In this case, using OWL/DL to model linguistic data types allows us to *infer* the relevant distinction, the data model can thus be pruned from artifacts necessary for visualization, etc.

The concept **Root** was introduced for organizational reasons. It corresponds to a **Nonterminal** that does not have a parent (and may be either a **Terminal** or a **Nonterminal**). Roots play an important role in structuring annosets: A **DocumentLayer** (a **Layer** defined for one specific **Document**) can be defined as a collection of **Roots**, so that it is no longer necessary to link every **Node** with the corresponding **Layer**, but only the top-most **Nodes**. In ANNIS, **Roots** are currently calculated during runtime.

Both **Terminals** and **Nonterminals** are characterized by a string value (property **hasString**), and a particular position (properties **hasStart** and **hasEnd**) with respect to the primary data. **Terminals** are further connected with each other by means of **nextTerminal** properties. This is, however, a preliminary solution and may be revised. Further, **Terminals** may be linked to the primary data (strings) in accordance to the currently developed NLP Interchange Format (NIF) ²

The **POWLAElement Layer** corresponds to a layer in PAULA. It is characterized by an ID, and can be annotated with metadata. **Layer** refers to a *phenomenon*, however, not to one specific layer within a document (annoset). Within a document, the subconcept **DocumentLayer** is to be used, that is assigned all **Root** nodes associated with this particular layer (property **rootOfDocument**). A **Root** may have at most one **Layer**.

The **POWLAElement Document** corresponds to a PAULA document, i.e., an annoset, or annotation project that assembles all annotations of a body of text and its parts. An annoset may contain other annotation projects (**hasSubDocument**), if it does so, it represents a collection of documents (e.g., a subcorpus, or a pair

² <http://nlp2rdf.org/nif-1-0#toc-nif-recipe-offset-based-uris>

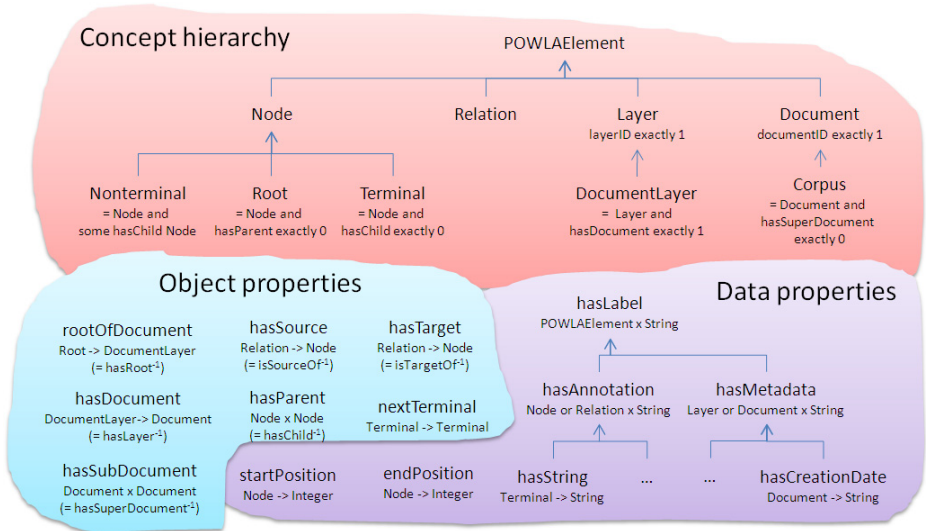


Fig. 2. The POWLA ontology (fragment)

of texts in a parallel corpus), otherwise, it contains the annotations of one particular text. In this case, it is a collection of different **DocumentLayers** (property **hasDocument**). A **Corpus** is a **Document** that is not a subdocument of another **Document**.

A diagram showing core components of the ontology is shown in Fig. 2.

2.3 POWLA ABox: Modelling Linguistic Annotations in POWLA

The POWLA ontology defines data types that can now be used to represent linguistic annotations. Considering the phrase *viele Kulturschätze* ‘many cultural treasures’ from the German sentence analyzed in Fig. 1, **Terminals**, **Nonterminals** and **Relations** are created as shown in Figs. 3 and 4.

Terminals *tok.51* and *tok.52* are the terminals *Viele* and *Kulturschätze*. The **Nonterminal** *nt.413* is the NP dominating both, the **Relation** *rel.85* is the relation between *nt.413* and *tok.51*. The properties **hasPos**, **hasCat** and **hasFunc** are subproperties of **hasAnnotation** that were created to reflect the **pos**, **cat** and **func** labels of nodes and edges in Fig. 1. **Relation** *rel.85* is marked as a dominance relation by the accompanying **hasChild** relation between its source and target.

As for corpus organization, the **Root** of the tree dominating *nt.413* is *nt.400* (the node with the label **TOP** in Fig. 1), and it is part of a **DocumentLayer** with the ID *tiger*. This **DocumentLayer** is part of a **Document**, etc., but for reasons of brevity, this is not shown here.

```

<powla:Terminal rdf:ID="tok.51">
  <powla:startPosition>434</powla:startPosition>
  <powla:endPosition>438</powla:endPosition>
  <powla:hasString>Viele</powla:hasString>
  <powla:hasPos>PIAT</powla:hasPos>
  <powla:nextTerminal rdf:about="#tok.52"/>
</powla:Terminal>
<powla:Terminal rdf:ID="tok.52">
  <powla:startPosition>439</powla:startPosition>
  <powla:endPosition>450</powla:endPosition>
  <powla:hasString>Kulturschätze</powla:hasString>
  <powla:hasPos>NN</powla:hasPos>
  ...

```

Fig. 3. Examples of Terminals in POWLA

```

<powla:Nonterminal rdf:ID="nt.413">
  <powla:hasChild rdf:about="#tok.51"/>
  <powla:hasChild rdf:about="#tok.52"/>
  <powla:startPosition>434</powla:startPosition>
  <powla:endPosition>450</powla:endPosition>
  <powla:hasCat>NP</powla:hasCat>
  <powla:isSourceOf rdf:about="#rel.85"/>
  ...
<powla:Relation rdf:ID="rel.85">
  <powla:hasSource rdf:about="#nt.413"/>
  <powla:hasTarget rdf:about="#tok.51"/>
  <powla:hasFunc>NK</powla:hasFunc>
  ...

```

Fig. 4. Examples of Nonterminals and Relations in POWLA

It should be noted that this representation in OWL/RDF is by no means complete. Inverse properties, for example, are missing. Using a reasoner, however, the missing RDF triples can be inferred from the information provided explicitly. A reasoner would also allow us to verify whether the necessary cardinality constraints are respected, e.g., every *Root* assigned to a *DocumentLayer*, etc.

Although illustrated here for syntax annotations only, the conversion of other annotation layers from PAULA to POWLA is similarly straight-forward. As sketched above, all PAULA data types can be modelled in OWL, and by *Root* and *DocumentLayer*, also PAULA namespaces (“tiger” for the example in Fig. 1) can be represented.

3 Corpora as Linked Data

With POWLA specifications as sketched above, linguistic annotations can be represented in RDF, with OWL/DL-defined data types. From the perspective of computational linguistics, this offers a number of advantages as compared to state-of-the-art solutions using standoff XML (i.e., a bundle of separate XML files that are densely interconnected with XLink and XPointer) as representation formalism and relational data bases as means for querying (e.g., [12] for PAULA XML, or [26,17] for GrAF):

1. Using OWL/DL reasoners, RDF data can be validated. (The semantics of XLink/XPointer references in standoff XML cannot be validated with standard tools, because XML references are *untyped*.)
2. Using RDF as representation formalism, multi-layer corpora can be directly processed with off-the-shelf data bases and queried with standard query languages. (XML data bases do not support efficient querying of standoff XML [18], relational data bases require an additional conversion step.)
3. RDF allows to combine information from different types of linguistic resources, e.g., corpora and lexical-semantic resources. They can thus be queried with the same query language, e.g., SPARQL. (To formulate similar queries using representation formalisms that are specific to either corpora or lexical-semantic resources like GrAF, or LMF [20], novel means of querying would yet have to be developed.)
4. RDF allows to connect linguistic corpora directly with repositories of reference terminology, thereby supporting the interoperability of corpora. (Within GrAF, references to the ISOcat data category registry [27] should be used for this purpose, but this recommendation does not make use of mechanisms that already have been standardized.)

The first benefit is sufficiently obvious not to require an in-depth discussion here, the second and the fourth are described in [11] and [10], respectively. Here, I focus on the third aspect, which can be more generally described as treating linguistic corpora as linked data.

The application of RDF to model linguistic corpora is sufficiently motivated from benefits (1) and (2), and this has been the motivation of several RDF/OWL formalizations of linguistic corpora [4,22,31,7]. RDF is, however, not only a way to represent linguistic data, but also, other forms of data, and in particular, to establish links between such resources. This is captured in the **linked data paradigm** [2] that consists of four rules: Referred entities should be designated by URIs, these URIs should be resolvable over http, data should be represented by means of standards such as RDF, and a resource should include links to other resources. With these rules, it is possible to follow links between existing resources, and thereby, to find other, related, data. If published as linked data, corpora represented in RDF can be linked with other resources already available in the Linked Open Data (LOD) cloud.³

³ <http://richard.cyganiak.de/2007/10/lod>

To this end, integrating corpora into the LOD cloud has not been suggested, probably mostly because of the gap between the linguistics and the Semantic Web communities. Recently, however, some interdisciplinary efforts have been brought forward in the context of the Open Linguistics Working Group of the Open Knowledge Foundation [13], an initiative of experts from different fields concerned with linguistic data, whose activities – to a certain extent – converge towards the creation of a Linguistic Linked Open Data (LLOD) (sub-)cloud that will comprise different types of linguistic resources, unlike the current LOD cloud also linguistic corpora. The following subsections describe ways in which linguistic corpora may be linked with other LOD (resp. LLOD) resources.

3.1 Grounding the POWLA Ontology in Existing Schemes

POWLA is grounded in Dublin Core (corpus organization), and closely related to the NLP Interchange Format NIF (elements of annotation).

In terms of Dublin Core, a POWLA Document is a `dctype:Collection` (it aggregates either different `DocumentLayers` or further `Documents`), a POWLA Layer is a `dctype:Dataset`, in that it provides data encoded in a defined structure. POWLA represents the primary data only in the values of `hasString` properties, hence, there is no `dctype:Text` represented here. Extending `Terminals` with string references as specified by NIF would allow us to point directly to the primary data (`dctype:Text`).

In Comparison to NIF, POWLA is more general (but also, less compact). Many NIF data structures can be regarded as specializations of POWLA categories, others are equivalent. For example, a NIF `String` corresponds to a POWLA `Node`, however, with more specific semantics, as it is tied to a stretch of text, whereas a POWLA `Node` may also be an empty element. The POWLA property `hasString` corresponds to NIF `anchorOf`, yet `hasString` is restricted to `Terminals`, whereas `anchorOf` is obligatory for all NIF `Strings`. Hence, both are not equivalent, however, it is possible to construct a generalization over NIF and POWLA that allows to define both data models as specializations of a common underlying model for NLP analyses and corpus annotations. The development of such a generalization and a transduction from NIF to POWLA is currently in preparation. NIF and POWLA are developed in close synchronization, albeit optimized for different application scenarios.

A key difference between POWLA and NIF is the representation of `Relations`, that correspond to object properties in NIF. Modeling edges as properties yields a compact representation in NIF (one triple per edge). In POWLA, it should be possible to assign a `Relation` to a `DocumentLayer`, i.e., a property with a `Relation` as subject. OWL/DL conformity requires to model `Relations` to be concepts (with `hasSource` and `hasTarget` at least 3 triples per edge). For the transduction from NIF to POWLA, such incompatibilities require more extensive modifications. At the moment, the details of such a transduction are actively explored by POWLA and NIF developers.

3.2 Linking Corpora with Lexical-Semantic Resources

So far, two resources have been converted using POWLA, including the NEGRA corpus, a German newspaper corpus with annotations for morphology and syntax [33], as well as coreference [32], and the MASC corpus, a manually annotated subcorpus of the Open American Corpus, annotated for a great band-width of phenomena [23]. MASC is represented in GrAF, and a GrAF converter has been developed [11].

MASC includes semantic annotations with FrameNet and WordNet senses [1]. WordNet senses are represented by sense keys as string literals. As sense keys are stable across different WordNet versions, this annotation can be trivially rendered in URIs references pointing to an RDF version of WordNet. (However, the corresponding WordNet version 3.1 is not yet available in RDF.)

FrameNet annotations in MASC make use of feature structures (attribute-value pairs where the value can be another attribute-value pair), which are not yet fully supported by the GrAF converter. However, reducing feature structures to simple attribute-value pairs is possible. The values are represented in POWLA as literals, but can likewise be transduced to properties pointing to URIs, if the corresponding FrameNet version is available. An OWL/DL version of FrameNet has been announced at the FrameNet site, it is, however, available only after registration, and hence, not strictly speaking an open resource.

With this kind of resources being made publicly available, it would be possible to develop queries that combine elements of both the POWLA corpus and lexical-semantic resources. For example, one may query for sentences about *land*, i.e., ‘retrieve every (POWLA) sentence that contains a (WordNet-)synonym of *land*’. Such queries can be applied, for example, to develop semantics-sensitive corpus querying engines for linguistic corpora.

3.3 Meta Data and Terminology Repositories

In a similar way, corpora can also be linked to other resources in the LOD cloud that provide identifiers that can be used to formalize corpus meta data, e.g., provenance information. Lexvo [15] for example, provides identifiers for languages, GeoNames [36] provides codes for geographic regions. ISOcat [28] is another repository of meta data (and other) categories maintained by ISO TC37/SC4, for which an RDF interface has recently been proposed [37].

Similarly, references to terminology repositories may be used instead of string-based annotations. For example, the OLiA ontologies [8] formalize numerous annotation schemes for morphosyntax, syntax and higher levels of linguistic description, and provide a linking to the morphosyntactic profile of ISOcat [9] with the General Ontology of Linguistic Description [19], and other terminology repositories. By comparing OLiA annotation model specifications with tags used in a particular layer in a particular layer annotated according to the corresponding annotation scheme, the transduction from string-based annotation to references to community-maintained category repository is eased. Using such a resource to describe the annotations in a given corpus, it is possible to abstract

from the surface form a particular tag and to interpret linguistic annotations on a conceptual basis.

Linking corpora with terminology and metadata repositories is thus a way to achieve **conceptual interoperability** between linguistic corpora and other resources.

4 Results and Discussion

This paper presented preliminaries for the development of a generic OWL/DL-based formalism for the representation of linguistic corpora. As compared to related approaches [4,22,31], the approach described here is not tied a restricted set of annotations, but applicable to any kind of text-based linguistic annotation, because it takes its point of departure from a generic data model known to be capable to represent any kind of linguistic annotation.

One concrete advantage of the OWL/RDF formalization is that it represents a standardized to represent heterogeneous data collections (whereas standard formats developed within the linguistic community are still under development): With RDF, a standardized representation formalism for different corpora is available, and with datatypes being defined in OWL/DL, the validity of corpora can be automatically checked (according to the consistency constraints posited by the POWLA ontology). POWLA represents a possible solution to the **structural interoperability** challenge for linguistic corpora [24]. In comparison to other formalisms developed in this direction (including ATLAS [3], NXT [6], GrAF and PAULA), it does, however, not propose a special-purpose XML standoff format, but rather, it employs existing and established standards with broad technical support (schemes, parsers, data bases, query language, editors/browsers, reasoners) and an active and comparably large community. Standard formats specifically designed for linguistic annotations as developed in the context of the ISO TC37/SC4 (e.g., GrAF), are, however, still under development.

As mentioned above, the development of POWLA as a representation formalism for annotated linguistic corpora is coordinated with the development of the NLP Interchange Format NIF [21]. Both formats are designed to be mappable, but they are optimized for different fields of application: POWLA is developed to represent annotated corpora with a high degree of genericity, whereas NIF is a compact and NLP-specific format for a restricted set of annotations. At the moment, NIF is capable to represent morphosyntactic and syntactic annotations only, the representation of more complex forms of annotation, e.g., alignment in a parallel corpus, has not been addressed so far. Another important difference is that NIF lacks any formalization of corpus structure. NIF is thus more compact, but the POWLA representation is more precise and more expressive, and both are designed to be mappable. This means that NIF annotations can be converted to POWLA representations, and then, for example, combined with other annotation layers.

PAULA is closely related to other standards: It is based on early drafts for the Linguistic Annotation Framework [25, LAF] developed by the ISO TC37/SC4.

Although it predates the official LAF linearization GrAF [26] by several years [16], it shares its basic design as an XML standoff format and the underlying graph-based data model. One important difference is, however, the treatment of segmentation [14]. While PAULA provides formalized terminal elements with XLink/XPointer references to spans in the primary data, GrAF describes segments by a sequence of numerical ‘anchors’. Although the resolution of GrAF anchors is comparable to that of `Terminals` in PAULA, the key difference is that anchor resolution is not formalized within the GrAF data model.

This has implications for the RDF linearizations of GrAF data: The RDF linearization of GrAF recently developed by [7] represents anchors as literal strings consisting of two numerical, space-separated IDs (character offsets) like in GrAF. This approach, however, provides no information how these IDs should be interpreted (the reference to the primary data is not expressed). In POWLA, `Terminals` are modeled as independent resources and information about the surface string and the original order of tokens is provided. Another difference is that this RDF linearization of GrAF is based on single GrAF files (i.e., single annotation layers), and that it does not build up a representation of the entire annotation project, but that corpus organization is expressed implicitly through the file structure which is inherited from the underlying standoff XML. It is thus not directly possible to formulate SPARQL queries that refer to the same annotation layer in different documents or corpora.

Closer to our conceptualization is [4] who used OWL/DL to model a multi-layer corpus with annotations for syntax and semantics. The advantages of OWL/DL for the representation of linguistic corpora were carefully worked out by the authors. Similar to our approach, [4] employed an RDF query language for querying. However, this approach was specific to a selected resource and its particular annotations, whereas POWLA, is a generic formalism for linguistic corpora based on established data models developed to the interoperable formalization of arbitrary linguistic annotations assigned to textual data.

As emphasized above, a key advantage of the representation of linguistic resources in OWL/RDF is that they can be published as linked data [2], i.e., that different corpus providers can provide their annotations at different sites, and link them to the underlying corpus. For example, the Prague Czech-English Dependency Treebank⁴, which is an annotated translation of parts of the Penn Treebank, could be linked to the original Penn Treebank. Consequently, the various and rich annotations applied to the Penn Treebank [30] can be projected onto Czech.⁵ Similarly, existing linkings between corpora and lexical-semantic resources, represented so far by string literals, can be transduced to URI references if the corresponding lexical-semantic resources are provided as linked

⁴ <http://www ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2004T25>

⁵ Unlike existing annotation projection approaches, however, this would not require that English annotations are directly applied to the Czech data – which introduces additional noise –, but instead, SPARQL allows us to follow the entire path from Czech to English to its annotations, with the noisy part (the Czech-English alignment) clearly separated from the secure information (the annotations).

data. An important aspect here is that corpora can be linked to other resources from the Linked Open Data cloud *using the same formalism*.

Finally, linked data resources can be used to formalize meta data or linguistic annotations. This allows, for example, to use information from terminology repositories to query a corpus. As such, the corpus can be linked to terminology repositories like the OLiA ontologies, ISocat or GOLD, and these community-defined data categories can be used to formulate queries that are independent from the annotation scheme, but use an abstract, and well-defined vocabulary. In this way, linguistic annotations in POWLA are not only structurally interoperable (they use the same representation formalism), but also conceptually interoperable (they use the same vocabulary).

References

1. Baker, C.F., Fellbaum, C.: WordNet and FrameNet as Complementary Resources for Annotation. In: Proceedings of the Third Linguistic Annotation Workshop, pp. 125–129 (August 2009)
2. Berners-Lee, T.: Design issues: Linked data (2006), <http://www.w3.org/DesignIssues/LinkedData.html>
3. Bird, S., Liberman, M.: A formal framework for linguistic annotation. *Speech Communication* 33(1-2), 23–60 (2001)
4. Burchardt, A., Padó, S., Spohr, D., Frank, A., Heid, U.: Formalising Multi-layer Corpora in OWL/DL – Lexicon Modelling, Querying and Consistency Control. In: Proceedings of the 3rd International Joint Conf. on NLP (IJCNLP 2008), Hyderabad (2008)
5. Carletta, J., Evert, S., Heid, U., Kilgour, J.: The NITE XML Toolkit: data model and query. *Language Resources and Evaluation Journal (LREJ)* 39(4), 313–334 (2005)
6. Carletta, J., Evert, S., Heid, U., Kilgour, J., Robertson, J., Voormann, H.: The NITE XML Toolkit: flexible annotation for multi-modal language data. *Behavior Research Methods, Instruments, and Computers* 35(3), 353–363 (2003)
7. Cassidy, S.: An RDF realisation of LAF in the DADA annotation server. In: Proceedings of ISA-5, Hong Kong (2010)
8. Chiarcos, C.: An ontology of linguistic annotations. *LDV Forum* 23(1), 1–16 (2008)
9. Chiarcos, C.: Grounding an ontology of linguistic annotations in the Data Category Registry. In: Workshop on Language Resource and Language Technology Standards (LR<S 2010), held in conjunction with LREC 2010, Valetta, Malta (May 2010)
10. Chiarcos, C.: Interoperability of corpora and annotations. In: Chiarcos, C., Nordhoff, S., Hellmann, S. (eds.) *Linked Data in Linguistics. Representing and Connecting Language Data and Language Metadata*, pp. 161–179. Springer, Heidelberg (2012)
11. Chiarcos, C.: A generic formalism to represent linguistic corpora in RDF and OWL/DL. In: 8th International Conference on Language Resources and Evaluation (LREC 2012) (accepted, 2012)
12. Chiarcos, C., Dipper, S., Götze, M., Leser, U., Lüdeling, A., Ritz, J., Stede, M.: A flexible framework for integrating annotations from different tools and tag sets. *Traitement Automatique des Langues* 49(2) (2009)

13. Chiarcos, C., Hellmann, S., Nordhoff, S.: The Open Linguistics Working Group of the Open Knowledge Foundation. In: Chiarcos, C., Nordhoff, S., Hellmann, S. (eds.) *Linked Data in Linguistics. Representing and Connecting Language Data and Language Metadata*, pp. 153–160. Springer, Heidelberg (2012)
14. Chiarcos, C., Ritz, J., Stede, M.: By all these lovely tokens... Merging conflicting tokenizations. *Journal of Language Resources and Evaluation (LREJ)* 46(1), 53–74 (2011)
15. De Melo, G., Weikum, G.: Language as a foundation of the Semantic Web. In: *Proceedings of the 7th International Semantic Web Conference (ISWC 2008)*, vol. 401 (2008)
16. Dipper, S.: XML-based stand-off representation and exploitation of multi-level linguistic annotation. In: *Proceedings of Berliner XML Tage 2005 (BXML 2005)*, Berlin, Germany, pp. 39–50 (2005)
17. Eckart, K., Riestler, A., Schweitzer, K.: A discourse information radio news database for linguistic analysis. In: Chiarcos, C., Nordhoff, S., Hellmann, S. (eds.) *Linked Data in Linguistics*, Springer, Heidelberg (2012)
18. Eckart, R.: Choosing an XML database for linguistically annotated corpora. *Sprache und Datenverarbeitung* 32(1), 7–22 (2008)
19. Farrar, S., Langendoen, D.T.: An OWL-DL implementation of GOLD: An ontology for the Semantic Web. In: Witt, A.W., Metzger, D. (eds.) *Linguistic Modeling of Information and Markup Languages: Contributions to Language Technology*, Springer, Dordrecht (2010)
20. Francopoulo, G., Bel, N., George, M., Calzolari, N., Monachini, M., Pet, M., Soria, C.: Multilingual resources for NLP in the Lexical Markup Framework (LMF). *Language Resources and Evaluation* 43(1), 57–70 (2009)
21. Hellmann, S.: The semantic gap of formalized meaning. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) *ESWC 2010. LNCS*, vol. 6089, pp. 462–466. Springer, Heidelberg (2010)
22. Hellmann, S., Unbehauen, J., Chiarcos, C., Ngonga Ngomo, A.: The TIGER Corpus Navigator. In: *9th International Workshop on Treebanks and Linguistic Theories (TLT-9)*, Tartu, Estonia, pp. 91–102 (2010)
23. Ide, N., Fellbaum, C., Baker, C., Passonneau, R.: The manually annotated subcorpus: A community resource for and by the people. In: *Proceedings of the ACL 2010*, pp. 68–73 (2010)
24. Ide, N., Pustejovsky, J.: What does interoperability mean, anyway? Toward an operational definition of interoperability. In: *Proceedings of the Second International Conference on Global Interoperability for Language Resources (ICGL 2010)*, Hong Kong, China (2010)
25. Ide, N., Romary, L.: International standard for a linguistic annotation framework. *Natural language engineering* 10(3–4), 211–225 (2004)
26. Ide, N., Suderman, K.: GrAF: A graph-based format for linguistic annotations. In: *Proceedings of The Linguistic Annotation Workshop (LAW) 2007*, Prague, Czech Republic, pp. 1–8 (2007)
27. Kemps-Snijders, M., Windhouwer, M., Wittenburg, P., Wright, S.: ISOcat: Corraling data categories in the wild. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco (May 2008)
28. Kemps-Snijders, M., Windhouwer, M., Wittenburg, P., Wright, S.: ISOcat: Remodelling metadata for language resources. *International Journal of Metadata, Semantics and Ontologies* 4(4), 261–276 (2009)

29. Marcus, M., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics* 19(2), 313–330 (1993)
30. Pustejovsky, J., Meyers, A., Palmer, M., Poesio, M.: Merging PropBank, NomBank, TimeBank, Penn Discourse Treebank and Coreference. In: *Proc. of ACL Workshop on Frontiers in Corpus Annotation* (2005)
31. Rubiera, E., Polo, L., Berrueta, D., El Ghali, A.: TELIX: An RDF-based Model for Linguistic Annotation. In: Simperl, E., et al. (eds.) *ESWC 2012. LNCS*, vol. 7295, pp. 195–209. Springer, Heidelberg (2012)
32. Schiehlen, M.: Optimizing algorithms for pronoun resolution. In: *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, Geneva, pp. 515–521 (August 2004)
33. Skut, W., Brants, T., Krenn, B., Uszkoreit, H.: A linguistically interpreted corpus of German newspaper text. In: *Proc. ESSLLI Workshop on Recent Advances in Corpus Annotation*, Saarbrücken, Germany (1998)
34. Stede, M.: The Potsdam Commentary Corpus. In: *Proceedings of the ACL Workshop on Discourse Annotation*, pp. 96–102, Barcelona, Spain (2004)
35. Stede, M., Bieler, H.: The MOTS Workbench. In: Mehler, A., Kühnberger, K.-U., Lobin, H., Lüngen, H., Storrer, A., Witt, A. (eds.) *Modeling, Learning, and Proc. of Text-Tech. Data Struct. SCI*, vol. 370, pp. 15–34. Springer, Heidelberg (2011)
36. Vatan, B., Wick, M.: GeoNames ontology, version 3.01 (February 2012), <http://www.geonames.org/ontology> (accessed March 15, 2012)
37. Windhouwer, M., Wright, S.E.: Linking to linguistic data categories in ISOcat. In: *Linked Data in Linguistics (LDL 2012)*, Frankfurt/M., Germany (accepted March 2012)
38. Windhouwer, M., Wright, S.E.: Linking to linguistic data categories in ISOcat. In: Chiarcos, C., Nordhoff, S., Hellmann, S. (eds.) *Linked Data in Linguistics. Representing and Connecting Language Data and Language Metadata*, pp. 99–107. Springer, Heidelberg (2012)

Representing Mereotopological Relations in OWL Ontologies with ONTOPARTS

C. Maria Keet¹, Francis C. Fernández-Reyes², and Annette Morales-González³

¹ School of Computer Science, University of KwaZulu-Natal, South Africa

² Instituto Superior Politécnico “José Antonio Echeverría” (CUJAE), Cuba

³ Advanced Technologies Application Center, CENATAV, Havana City, Cuba
keet@ukzn.ac.za, ffernandez@ceis.cujae.edu.cu, amorales@cenatav.co.cu

Abstract. Representing and reasoning over mereotopological relations (parthood and location) in an ontology is a well-known challenge, because there are many relations to choose from and OWL has limited expressiveness in this regard. To address these issues, we structure mereotopological relations based on the KGMT mereotopological theory. A correctly chosen relation counterbalances some weaknesses in OWL’s representation and reasoning services. To achieve effortless selection of the appropriate relation, we hide the complexities of the underlying theory through automation of modelling guidelines in the new tool ONTOPARTS. It uses, mainly, the categories from DOLCE [12], which avoids lengthy question sessions, and it includes examples and verbalizations. ONTOPARTS was experimentally evaluated, which demonstrated that selecting and representing the desired relation was done efficiently and more accurately with ONTOPARTS.

1 Introduction

Part-whole relations are essential for knowledge representation, in particular in terminology and ontology development in subject domains such as biology, medicine, GIS, and manufacturing. Usage of part-whole relations are exacerbated when part-whole relations are merged with topological or mereotopological relations, such as *tangential proper part* where the part touches the boundary of the whole it is part of; e.g., the FMA has 8 basic locative part-whole relations [14] and GALEN has 26 part-whole and locative part-whole relations [1]. It is also useful for annotating and querying multimedia documents and cartographic maps; e.g., annotating a photo of a beach where the area of the photo that depicts the sand *touching* the area that depicts the seawater so that, together with the knowledge that, among other locations, Varadero is a *tangential proper part of* Cuba, the semantically enhanced system can infer possible locations where the photo has been taken, or vv., propose that the photo may depict a beach scene.

Efforts have gone into figuring out which part-whole relations there are [24, 11], developing a logic language with which one can represent the semantics of the

¹ <http://www.opengalen.org/tutorials/crm/tutorial9.html>
up to [/tutorial16.html](#)

relations [2,8], and how to use the two together [20,25,26]. The representation of mereotopology in Description Logics (DL) has not been investigated, but related efforts in representing the Region Connection Calculus (RCC) in DLs have [17,9,21,23,6]. Currently, the advances in mereotopology are not directly transferrable to a Semantic Web setting due to the differences in languages and theories and they miss software support to make it usable for the ontology developer. Yet, ontologists require a way to effectively handle these part-whole relations during ontology development without necessarily having to become an expert in theories about part-whole relations, mereotopology, and expressive ontology languages. Moreover, structured and guided usage can prevent undesirable deductions and increase the amount of desirable deductions even without the need to add additional expressiveness to the language. For instance, instance classification: let NTPLI be a ‘non-tangential proper located in’ relation and $\text{EnclosedCountry} \equiv \text{Country} \sqcap \exists \text{NTPLI}.\text{Country}$, and instances $\text{NTPLI}(\text{Lesotho}, \text{South Africa}), \text{Country}(\text{Lesotho}), \text{Country}(\text{South Africa})$, then it will correctly deduce $\text{EnclosedCountry}(\text{Lesotho})$. With merely ‘part-of’, one would not have been able to obtain this result.

Thus, there are three problems: (i) the lack of oversight on plethora of part-whole relations, that include real parthood (mereology) and parts with their locations (mereotopology), (ii) the challenge to figure out which one to use when, and (iii) underspecified representation and reasoning consequences. To solve these problems we propose the ONTOPARTS tool to guide the modeller. To ensure a solid foundation, transparency, a wide coverage of the types of part-whole relations, and effectiveness during ontology development, we extend the taxonomy of part-whole relations of [11] with the novel addition of mereotopological relations, driven by the KGEMT mereotopological theory [22], resulting in a taxonomy of 23 part-whole relations. We describe the design rationale and trade-offs with respect to what has to be simplified from KGEMT to realise as much as possible in OWL so that ONTOPARTS can load OWL/OWL2-formalised ontologies, and, if desired, modify the OWL file with the chosen relation. To enable quick selection of the appropriate relation, we use a simplified OWL-ized DOLCE ontology for the domain and range restrictions imposed on the part-whole relations and therewith let the user take ‘shortcuts’, which reduces the selection procedure to 0-4 options based on just 2-3 inputs. The usability of ONTOPARTS and effectiveness of the approach was evaluated and shown to improve efficiency and accuracy in modelling.

In the remainder of the paper, we first describe the theoretical foundation of the mereotopological relations and trade-offs for OWL (Section 2). We describe the design, implementation, and evaluation of ONTOPARTS in Section 3, discuss the proposed solution in Section 4, and conclude in Section 5.

2 Mereotopology and OWL

Part-whole relations and the differentiation between different types of part-whole relations has its origins in cognitive science and conceptual data modelling [24,18]

and has been investigated also for Description Logics (e.g., [112]). There is a first important distinction between parthood versus a meronymic relation (‘part’ in natural language only), and, second, there is an additional aspect on parthood and location [11]. The second dividing characteristic is the domain and range of the relations (which are taken from the DOLCE foundational ontology [12] in [11]). Particularly relevant here are the containment and location axioms (Eqs. 1 and 2) [11], where $ED = \text{EnDurant}$ (enduring entity, e.g., a wall), $R = \text{Region}$ (e.g., the space that the wall occupies, the Alpine region), and has_2D and has_3D for surfaces and space are shorthand relations standing for DOLCE’s qualities and qualia; note that the domain and range is *Region* that has an object occupying it, hence, this does not imply that those objects are related also by structural parthood.

$$\begin{aligned} \forall x, y (\text{contained_in}(x, y) \equiv & \text{part_of}(x, y) \wedge R(x) \wedge R(y) \wedge \\ & \exists z, w (\text{has_3D}(z, x) \wedge \text{has_3D}(w, y) \wedge ED(z) \wedge ED(w))) \end{aligned} \quad (1)$$

$$\begin{aligned} \forall x, y (\text{located_in}(x, y) \equiv & \text{part_of}(x, y) \wedge R(x) \wedge R(y) \wedge \\ & \exists z, w (\text{has_2D}(z, x) \wedge \text{has_2D}(w, y) \wedge ED(z) \wedge ED(w))) \end{aligned} \quad (2)$$

Although logically strictly not necessary, there are strong arguments to distinguish between the surface and space notions [24, 11, 22, 18], which is especially relevant for geographic entities. For instance, a euro is *contained_in* a wallet, Paris is *located_in* France, and in the TBox of a DL-formalised ontology, e.g., $\text{City} \sqsubseteq \exists \text{locatedIn.Country}$.

2.1 Extension with Mereotopological Relations

Representation and reasoning with spatiality is very important in many applications areas such as geographical information systems, medical systems, and computer vision. Spatial relations proposed for such systems can be classified into three categories: topological, direction, and distance. Topological relations are considered to be the most important ones and several topological formalisms have been proposed in the literature. The 9-Intersection Method (9IM) is based on point-set topology, where the topological relations between two regions are characterized by the 9 intersections of interiors, boundaries and exteriors of the two regions [4]. The Region Connection Calculus (RCC) is a first order theory based on one primitive relation: the reflexive and symmetric *connection* predicate [19]. Using this primitive relation, one can define various collections of topological relations. The fundamental approach of RCC—and difference with point-set topology—is that it extends spatial entities to *regions of space* that are taken as primary rather than the points as the fundament in point-set topology. However, neither one considers the combination of the space region with the object that occupies it: this interaction is addressed by *mereotopology*, which focuses on spatial *entities*, not just regions. The challenge lies in how to realize the combination, given that no such ontology-informed categorisation for mereotopological relations exists. Concerning primitive relations [3, 22], one can

Table 1. Axiomatization of KGEMT (summarised from [22]). P: partof; PP: proper part of; O: overlap, C: connection; E: enclosure; EQ: indiscernibility; IPP: interior proper part; TPP: tangential proper part; SC: self-connected; c: closure; i: interior; e: exterior; +: sum; \sim : complement.

Core axioms and definitions			
$P(x, x)$	(t1)	$P(x, y) \wedge P(y, z) \rightarrow P(x, z)$	(t2)
$P(x, y) \wedge P(y, x) \rightarrow x = y$	(t3)	$\neg P(y, x) \rightarrow \exists z(P(z, y) \wedge \neg O(z, x))$	(t4)
$\exists w\phi(w) \rightarrow \exists z\forall w(O(w, z) \leftrightarrow \exists v(\phi(v) \wedge O(w, v)))$		(t5)	
$C(x, x)$	(t6)	$C(x, y) \rightarrow C(y, x)$	(t7)
$P(x, y) \rightarrow E(x, y)$	(t8)	$E(x, y) =_{df} \forall z(C(z, x) \rightarrow C(z, y))$	(t9)
$E(x, y) \rightarrow P(x, y)$	(t10)	$SC(x) \leftrightarrow \forall y, z(x = y + z \rightarrow C(y, z))$	(t11)
$\exists z(SC(z) \wedge O(z, x) \wedge O(z, y) \wedge \forall w(P(w, z) \rightarrow (O(w, x) \vee O(w, y)))) \rightarrow C(x, y)$		(t12)	
$z = \sum x\phi x \rightarrow \forall y(C(y, z) \rightarrow \exists x(\phi x \wedge C(y, x)))$		(t13)	
$P(x, cx)$	(t14)	$c(cx) = cx$	(t15)
$c(x + y) = cx + cy$	(t16)	$cx =_{df} \sim (ex)$	(t17)
$ex =_{df} i(\sim x)$	(t18)	$ix =_{df} \sum z\forall y(C(z, y) \rightarrow O(x, y))$	(t19)
Additional axioms, definitions, and theorems			
$PP(x, y) =_{df} P(x, y) \wedge \neg P(y, x)$	(t20)	$O(x, y) =_{df} \exists z(P(z, x) \wedge P(z, y))$	(t21)
$EQ(x, y) =_{df} P(x, y) \wedge P(y, x)$	(t22)	$TPP(x, y) =_{df} PP(x, y) \wedge \neg IPP(x, y)$	(t23)
$IPP(x, y) =_{df} PP(x, y) \wedge \forall z(C(z, x) \rightarrow O(z, y))$	(t24)		
$\neg PP(x, x)$	(t25)	$PP(x, y) \wedge PP(y, z) \rightarrow PP(x, z)$	(t26)
$PP(x, y) \rightarrow \neg PP(y, x)$	(t27)		

- define parthood, P , in terms of connection, C , (i.e., $P(x, y) =_{def} \forall z(C(z, x) \rightarrow C(z, y))$) so that topology is principal and mereology a subtheory,
- consider both P and C as primitive,
- introduce a ternary relation $CP(x, y, z)$, so that $P(x, y) =_{def} \exists z CP(x, z, y)$ and $C(x, y) =_{def} \exists z CP(x, y, z)$, or
- introduce topology as a sub-domain of mereology by introducing a sorted predicate to denote region (R) and define C in terms of overlapping regions [5] ($C(x, y) =_{def} O(x, y) \wedge R(x) \wedge R(y)$).

There are several problems with the first option, such as extensionality and identity [22] and that, implicitly, parthood is constrained to relating regions, because, by assumption, C is, even though normally an arbitrary part-whole relation should not be constrained to relate regions only. The third option requires a language with ternary relations, which OWL languages do not have, and it is not considered or used widely. The fourth option is, in spirit, close to Eqs. (11-2). However, adding complete definitions of both overlap and the connection is impossible in OWL; therefore, the second option with two primitives is the most appealing with the current Semantic Web languages and technologies. This fits nicely with the expressive Kuratowski General Extensional Mereotopology (KGEMT) [22], which provides both an ontological and logical underpinning of the mereotopology taxonomy. KGEMT consists of simpler mereological and topological theories. Formally, the axioms in KGEMT are as shown in Table 1, which is built up from T that is made up of (t6, t7), MT of (T + t8), M of (t1,

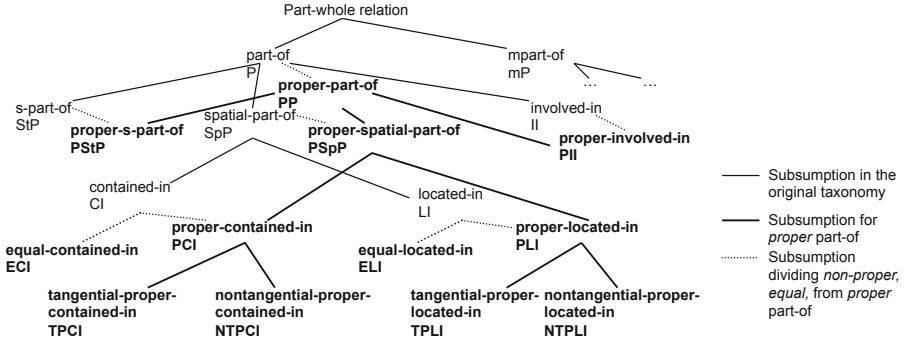


Fig. 1. Graphical depiction of the extension of the mereological branch of the basic taxonomy of part-whole relations with proper parthood and mereotopological relations (meronymic relations subsumed by *mpart-of* not shown)

t_2, t_3), GEM of ($M + t_4, t_5$), GEMT of ($MT + GEM + t_{10}, t_{12}, t_{13}$), and KGEMT of ($GEMT + t_{14}, t_{15}, t_{16}$). In addition, (t_{14} - t_{16}) require (t_{17} - t_{19}) and there are additional axioms and definitions (like t_{20} - t_{27}) that can be built up from the core ones.

Using Eqs. (12) and (t_{1} - t_{27}), we now extend the part-whole taxonomy of (11) with the mereotopological relations as defined in Eqs. (3-10), which is shown graphically in Fig. 1. The tangential and nontangential proper parthood relations are based on axioms 65 and 66 in [22], which are (t_{24}) and (t_{23}), respectively, in Table 1, and the same DOLCE categories are used as in Eqs. (12); see Fig. 1 and Table 1 for abbreviations.

$$\forall x, y (ECI(x, y) \equiv CI(x, y) \wedge P(y, x)) \quad (3)$$

$$\forall x, y (PCI(x, y) \equiv PPO(x, y) \wedge R(x) \wedge R(y) \wedge \exists z, w (has_3D(z, x) \wedge has_3D(w, y) \wedge ED(z) \wedge ED(w))) \quad (4)$$

$$\forall x, y (NTPCI(x, y) \equiv PCI(x, y) \wedge \forall z (C(z, x) \rightarrow O(z, y))) \quad (5)$$

$$\forall x, y (TPCI(x, y) \equiv PCI(x, y) \wedge \neg NTPCI(x, y)) \quad (6)$$

$$\forall x, y (ELI(x, y) \equiv LI(x, y) \wedge P(y, x)) \quad (7)$$

$$\forall x, y (PLI(x, y) \equiv PPO(x, y) \wedge R(x) \wedge R(y) \wedge \exists z, w (has_2D(z, x) \wedge has_2D(w, y) \wedge ED(z) \wedge ED(w))) \quad (8)$$

$$\forall x, y (NTPLI(x, y) \equiv PLI(x, y) \wedge \forall z (C(z, x) \rightarrow O(z, y))) \quad (9)$$

$$\forall x, y (TPLI(x, y) \equiv PLI(x, y) \wedge \neg NTPLI(x, y)) \quad (10)$$

Note that one also can use another foundational ontology, such as SUMO or GFO, provided it at least contains categories equivalent to the ones from DOLCE we use here. Concerning the interaction between this proposal for mereotopology and DOLCE: DOLCE includes the GEM mereological theory, of which KGEMT is an extension, and does not contain mereotopology; hence, our taxonomic categorisation of the mereotopological relations and additional KGEMT axioms do not lead to an inconsistency. Interaction with DOLCE's—or any other

foundational ontology’s—temporal parthood is unclear, as is a temporal extension to KGEMT; either way, they are orthogonal issues and it will become of practical interest only once there is a temporal OWL.

As with parthood and topology separately, the mereotopological relations are independent of the subject domain and thus may be applied across ontologies in domains such as medicine, manufacturing, and biology.

2.2 Limitations for Mereotopology in OWL

There are several aspects to consider with respect to ‘theoretical ideal’ from an ontological perspective, as described in the previous section, versus ‘realistic implementation’ in a Semantic Web setting, which are: (i) what to represent from mereotopology, (ii) how to represent the ontological aspects in a DL language, (iii) what does one lose with the OWL 2 languages, and (iv) how does this affect reasoning. Due to space limitations, we discuss only a selection of the issues.

Representation Issues. Only a subset of the KGEMT axioms can be represented in OWL, which is summarised in Table 2. The differences for the OWL species have to do with the characteristics of the object properties, and in particular transitivity, (ir)reflexivity and (a)symmetry (see also the OWL and OWL 2 specifications [13,16,15]), which are exactly the features that affect negatively the basic reasoning scenarios. This has the consequence that some theorems in KGEMT can only be asserted as axioms, due to its partial representation. However, even its limited version is still more appealing than only RCC8 for OWL because of the notion of spatial entity and relation and the inferences with the property characteristics that is important for several subject domains [11,22,18,5] compared to region only. In addition, in some implementations, the RCC8 relations are modified into a set-based approach where regions are non-empty closed sets [17] or where concepts are generated for each RCC8 constructor and the named concepts instantiated with individuals [9,21]. While conflating sets, regions, and part-whole relations might be appealing from a logic perspective as a sufficient approximation, it is not the same neither in detail nor from an ontological or modelling viewpoint such that other trade-offs in the representation are preferred (e.g., [22,6,23]), and in particular to keep the relations as such compared to a concept-based encoding that obscures the semantics and thereby complicates its usage [11,6,23]. Put differently, KGEMT is the best (or least worst) from a modeller point of view.

Reasoning Issues. We illustrate some of the inferences we miss out on with an extended African Wildlife Ontology, which includes the knowledge that South Africa (SA) is connected with Botswana (B), Mapungubwe National Park (MNP) (home to giraffes, elephants, etc.) is tangentially proper located in (TPLI) South Africa and bordering Botswana, and in Botswana, the park is called Northern Tuli Game Reserve (NTGR). So, to this end, there are axioms in our ontology in the TBox and the ABox, among others: $\text{NationalPark} \sqsubseteq \exists \text{LI.Country}$, $\text{ContinentalCountry} \sqsubseteq \text{Country}$, $\text{ContinentalCountry} \sqsubseteq \exists \text{C.Continental}$

Table 2. Subsets of KGEMT that can be represented in the OWL species; t9, t20, t21, t22, t23, t24 can be simplified and added as primitives to each one

OWL species	Subsets of KGEMT axioms
OWL 2 DL	(t1, t2, t6, t7, t8, t10, t26) or (t1, t2, t6, t7, t8, t10, t27) or (t1, t2, t6, t7, t8, t10, t25)
OWL DL	t2, t7, t8, t10, t26
OWL Lite	t2, t7, t8, t10, t26
OWL 2 RL	t2, t7, t8, t10, t26
OWL 2 EL	t1, t2, t6, t8, t10, t26
OWL 2 QL	t1, t6, t7, t8, t10

Country, ContinentalCountry(SA), ContinentalCountry(B), NationalPark(MNP), TPLI(MNP, SA), C(MNP, NTGR), TPLI(NTGR, B). Then a query “Which South African national park is connected with Northern Tuli Game Reserve?” has as answer MNP in a language where symmetry can be asserted, and is empty in the absence of the symmetry property assertion (i.e., in OWL 2 EL). In addition, with TPLI being asymmetric, an erroneous addition of, say, TPLI(SA, MNP) to the ontology yields an inconsistent ontology, whereas without the option to represent asymmetry, the error (with respect to the subject domain semantics) would go undetected.

Parthood and connection are reflexive, which cannot be represented in OWL 2 RL. This does not really cause any major problems, other than that in certain cases its inclusion communicates to the modeller she has to be more precise. For instance, with Leaf being a StP (structural part of) Twig and StP reflexive, the automated reasoner will deduce that Leaf is a StP of Leaf, and with aforementioned axioms, that MNP is C (connected with) and LI (located in) MNP: the former is a rather odd deduction, and, in fact, should be structural *proper* part—which is irreflexive—of Twig, and the latter two are uninteresting, and, in fact, TPLI. Also, if a user queries “Which park is connected with Northern Tuli Game Reserve?”, then one would want to retrieve all parks *other than* NTGR.

Which of the characteristics of the object properties of mereotopological relations is, or are, more important depends on the desired inference scenarios; thus far, transitivity, symmetry, asymmetry, and irreflexivity appear to be the more interesting ones, which—within a Semantic Web setting—means giving precedence to OWL 2 DL and OWL 2 RL.

3 Design and Implementation of ONTOPARTS

Given the theoretical assessment on representing part-whole and mereotopological relations and feasibility to simplify it for a Semantic Web setting, we can now proceed toward the design and implementation of the ONTOPARTS tool. The tool, additional files, and demo videos can be consulted in the online supplementary material at <http://www.meteck.org/files/ontopartssup/supindex.html>.

3.1 Requirements, Design, and Core Functionality

The main requirement of the software is to hide the logic involved in the formal definition of the 23 part-whole relations, and translate it to a set of intuitive steps and descriptions that will guide the user to make the selection and decision effortlessly. The selection procedure for the 23 possible relations should be made as short as possible and present only a small relevant subset of suggestions from which the user can select the one that best fit the situation. A set of (top-level) categories should be proposed to quickly discriminate among relations since the user may be more familiar with the categories' notions for domain and range than with the relations' definitions, therewith standardizing the criteria for selecting the relations. Simple examples must be given for each relation and category. Last, the user must have the possibility also to save the selected relation to the ontology file from where the classes of interest were taken.

Given these basic functional requirements, some design decisions were made for ONTOPARTS. From a generic perspective, a separate tool is an advantage, because then it can be used without binding the ontologist to a single ontology editor. Another consideration is usability testing. We chose to use a rapid way of prototyping to develop the software to quickly determine whether it is really helpful. Therefore, we implemented a stand-alone application that works with OWL files. We also chose to use the DOLCE top-level ontology categories for the standardization of the relationships' decision criteria.

To structure the selection procedure in a consistent way in the implementation, we use activity diagrams to describe the steps that a user has to carry out to interact with ONTOPARTS and to select the appropriate relation. An activity diagram for the selection process of the mereotopological relations is available in the online supplementary material. The selection of the appropriate relation incorporates some previous ideas of a decision diagram and topological principles as an extension of mereological theories [10,25,26], and questions and decision points have been added to reflect the extended taxonomy. For the mereotopological relations considered, in principle, the decision for the appropriate one can be made in two separate ways: either find mereotopological relations and then asking to distinguish between *located_in* and *contained_in*, or vv. In the ONTOPARTS interface, we have chosen to reduce the sequence of questions to a single question (check box) that appears only when the domain and range are regions, which asks whether the classes are geographical entities.

Concerning the most expressive OWL 2 DL species and KGEMT, then antisymmetry (t3), the second order axioms (t5, t13), and the closure operators (t14-t19) are omitted, and definitions of relations are simplified to only domain and range axioms and their position in the hierarchy (recollect Table 2). In addition, OWL's IrreflexiveObjectProperty and AsymmetricObjectProperty can be used only with simple object properties, which runs into problems with a taxonomy of object properties and transitivity (that are deemed more important), therefore also (t25, t27) will not appear in the OWL file. The combination of the slimmed KGEMT and extended taxonomy of part-whole object properties together with a DOLCE ultra-ultra-light is included in the online supplementary

material as `MereoTopoD.owl`. This OWL file contains the relations with human-readable names, as included in the taxonomy of [11] and the mereotopological extension depicted in Fig. 1, as it is generally assumed to be better workable to write out abbreviations in the domain ontology as it increases readability and understandability of the ontology from a human expert point of view.

Observe that at the class-level, we have the so-called “all-some” construction for property axioms, and if the modeller wants to modify it with a min-, max-, or exact cardinality (e.g., ‘each spinal column is a proper part of exactly one human’), then it goes beyond OWL 2 DL because the `properPartOf` object property is not simple. Further, transitivity is a feature of OWL-DL, OWL Lite, OWL 2 DL, DL, EL and RL, but not QL. Because one cannot know upfront the setting of the ontology, we keep the hierarchy of relations but do not add the relational properties when writing into the `.owl` file, but the user can add them afterward.

3.2 Description of the Selection Procedure

ONTOPARTS guides the user in the process of making the decision in a very intuitive way. To select the correct part-whole relation between two classes, one first has to load the ontology into the tool and select the class that represent the part and the class that represent the whole, upon which the selection procedure commences. To not overload the user at once with a choice of 23 possible relations or having to answer 22 questions first, ONTOPARTS narrows down the set of available relations when the user specifies the category of the domain and of the range involved in the relation, which are categories from DOLCE. If the user selects regions, then she enters the branch of mereotopological relations where the interface contains one decision point to distinguish between the 2D and 3D case. Upon finishing the above steps, the user clicks the button to let the tool suggest appropriate relation(s). All possible relations are computed using the DOLCE categories and the optional check box for geographic entities. Each proposed relation is verbalized in a pseudo-natural language sentence with the selected part-class and whole-class and an example as an additional guide to make the relation more understandable. Once a user has analysed each proposed relation, she can select one and proceed to save it in the OWL file by clicking on the save button. The selection procedure is illustrated in Example 1.

Example 1. Suppose a modeller or domain expert is developing a tourism ontology and has to figure out the relation between the classes `Campground` and `RuralArea`. She loads the OWL file in ONTOPARTS, and proceeds to select two entities to be related, as shown in Fig. 2. Then, she selects the categories for each entity from the ones provided by the software, depicted in Fig. 3. Examples for each DOLCE category are shown by hovering the pointer over the terms in the taxonomy; e.g., hovering over `Process` provides “Ex: Running, Writing”. If the selected categories are regions (or any of its subtypes), then the software provides the option to specify whether the regions correspond to geographical entities. After all the required information is selected, she clicks on the button “Suggest relationships”. The amount of relations suggested depends on the chosen categories; in the example there are still four options (Fig. 4), but if we

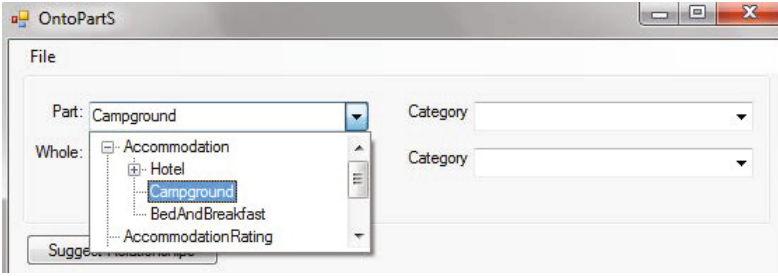


Fig. 2. Selecting the part and whole classes in ONTOPARTS from the loaded OWL file

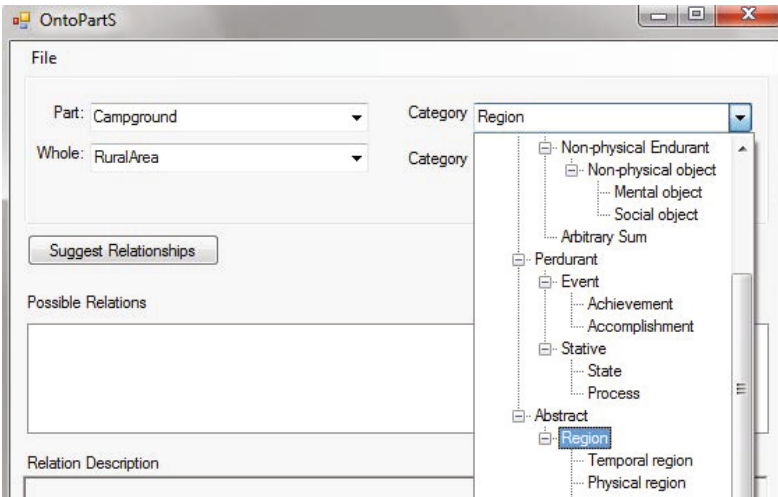


Fig. 3. Selecting the categories for each entity

would have had two classes that are both, say, processes, then there is only one option (*involved_in*, as ONTOPARTS includes the taxonomy of [11]). The software provides the suggestions, a verbalization of the possible relationship(s), e.g., “Campground coincides exactly with RuralArea”, and typical examples, as can be seen in Fig. 4. Once she selects the desired relation from the ones proposed by the software, she can choose to add this relationship to the OWL file by simply clicking the button labelled “Save relationship to file” (Fig. 4 bottom) and continue either with other classes and selection of a part-whole relation or with developing the ontology in the ontology development environment of choice. ◇

3.3 Preliminary Experimental Assessment of ONTOPARTS

The main objectives of the experiments are to assess usability of the tool and to validate the hypothesis that the use of automated guidelines assists with representation of part-whole relations between classes during the ontology design

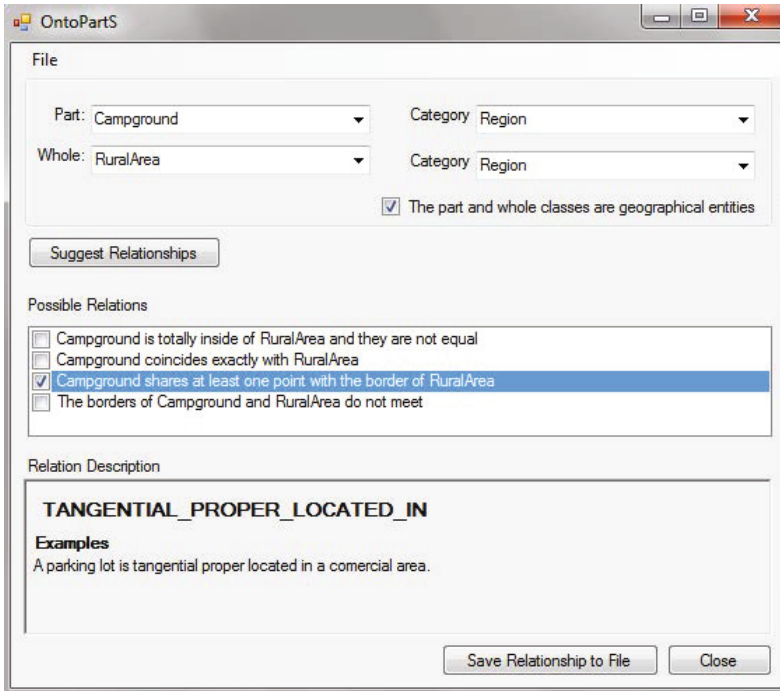


Fig. 4. Relationships suggested by ONTOPARTS

phase such that it can be done more efficiently and with less errors. To this end, a qualitative and two preliminary quantitative evaluations have been carried out.

Materials & Methods. 17 third year students in Informatics Engineering at the Instituto Superior Politécnico “José Antonio Echeverría” participated in the first test, who are well-versed in logics and knowledge representation with frames, nets and rules systems, but not with ontologies. The methodology for this experiment was as follows. The students receive an overview of foundational ontologies and part-whole relations (30 minutes). Then, from the provided computer ontology (designed by us) and ONTOPARTS, they must select the most appropriate DOLCE category for each subject domain class and the relation that holds between them. The evaluation is performed by assessing the OWL files, relations detected, selected DOLCE categories, the errors made, and why.

A second experiment was carried out with 6 honours students in Computer Science at the University of KwaZulu-Natal, with the main difference the group’s longer introduction into foundational ontologies (2h) and part-whole relations (2h), the division into two groups, one using ONTOPARTS, the other not, and the limitation of 40 minutes. The third group for qualitative evaluation consisted of 4 colleagues from the Free University of Bozen-Bolzano, two of whom experts in RCC8, part-whole relations, and logic.

The materials used for the experiments were the DOLCE taxonomy, the taxonomy with the 23 part-whole relations, and the beta version of the ONTOPARTS tool. The domain ontology about computers was developed using Protege 4.0, which was divided into two versions, one with and one without part-whole relations.

Results. The students in the first experiment asserted 380 part-whole relations among the classes in the domain ontology (37 mereotopological), of which 210 were correct, i.e., on average, about 12.4 correct assertions/participant (variation between 5 and 37); for the second experiment, the numbers are, respectively, 82 (22 mereotopological), 58, and an average of 9.7 (with variation between 0 and 27). Given the controlled duration of the second experiment, this amounts to, on average, a mere 4 minutes to choose the correct relation with ONTOPARTS.

Evaluating the mistakes made by the participants revealed that an incorrect selection of part-whole relation was due to, mainly, an incorrect association of a domain ontology class to a DOLCE category. This was due to the late discovery of the tool-tip feature in ONTOPARTS by some participants and the lack of an “undo” button (even though a user could have switched back to the ontology development environment and deleted the assertion manually). Several errors were due to the absence of inverses in the beta version of the ONTOPARTS tool, leading some participants to include `Metal constitutes some Hardware`. 83% of the errors in the second experiment were made by those who did not use ONTOPARTS, which looks promising for ONTOPARTS.

The responses in the qualitative evaluation was unanimous disbelief that selection could be made this easy and quickly, and the desire was expressed to consult the formal and ontological foundations. As such, ONTOPARTS stimulated interest for education on the topic along the line of “the tool makes it easy, then so the theory surely will be understandable”.

Overall, it cannot be concluded that modelling of part-whole relations with ONTOPARTS results in statistically significant less errors—for this we need access to more and real ontology developers so as to have a sufficiently large group whose results can be analysed statistically. Given the speed with which correct relations were selected, the automated guidelines do assist with representation of part-whole relations such that it can be done more efficiently and quickly. The experimentation also aided in improving ONTOPARTS’s functionality and usability, so that it is now a fully working prototype.

4 Discussion

Despite the representation and reasoning limitations with the DL-based OWL 2 species, there are several modelling challenges that can be addressed with the mereotopological and part-whole relation taxonomy together with ONTOPARTS and they solve the three problems identified in the introduction.

Mitigating Representation Limitations with the Taxonomy. ONTOPARTS sensitizes the modeller to part-whole relations, which thereby already prevents

the is-a vs. part-of confusion—i.e., using the is-a relation where a part-of relation should have been used—common among novice ontology developers, and no such errors were encountered during our experiments either (recollect Section 3.3). By making part-whole relations easily accessible without the need for immediate in-depth investigation into the topic, it is expected that this type of error may also be prevented without any prior substantial training on the topic.

The fine-grained distinctions between the parthood relation enables, among others, proper instance classification like mentioned in the introduction for `EnclosedCountry`, thanks to being able to select the right relation and thereby capturing the intended semantics of `EnclosedCountry`. If, on the other hand, the modeller would have known only about proper part-of but not proper located in, then she could only have asserted that Lesotho is a proper part of South Africa, which holds (at best) only *spatially* but not *administratively*. Not being able to make such distinctions easily leads to inconsistencies in the ontology or conflicts in ontology import, alignment, or integration. OntoClean [7] helps with distinguishing between geographic and social entity, and, in analogy, ONTOPARTS aids *relating* the entities with the appropriate relation.

Solving the Problems. In the introduction, we identified three problems that were in need of a solution to improve modelling of part-whole relations. The first problem regarding the plethora of part-whole relations has been solved by providing logic-based definitions of the relations and structuring them in a taxonomy. In so doing, we extended the taxonomy of [11] that already makes a clear distinction between parthood (mereology) versus other so-called part-whole relations (meronymy), so that the parts with their locations (mereotopology) could be integrated in the parthood branch of that taxonomy. Their position in the taxonomy follows directly from the definitions (Eqs. 3.10), which, in turn, are based on the KGEMT mereotopology and the common modeller’s preference to distinguish between 3D parthood (containment) and 2D parthood (location).

The challenge to figure out which part-whole relations to use when has been addressed by ONTOPARTS. Determining which part-whole relation fits best for the given classes requires only the input from the modeller about the categories and, depending on the category, if they are geographic entities. Upon this input, ONTOPARTS suggests 1-4 possible relations, which is considerably less than the full 23 together.

Implementing part-whole relations with its underlying KGEMT mereotopology in OWL requires making concessions, which concern principally the choice to represent relations as relations, and the impossibility to include definitions of relations and some of the characteristics of the part-whole object properties. The latter is practically important, because the OWL species differ in language features in this regard, and its consequences for reasoning were analysed. Transitivity, symmetry, asymmetry, and irreflexivity are the more relevant ones with respect to the deductions one gains or loses. This means giving precedence to OWL 2 DL and OWL 2 RL over the other OWL species when modelling part-whole relations.

5 Conclusions

We have introduced mereotopology from a modeller's perspective and integrated it into the taxonomy of part-whole relations that is founded on the KGEMT mereotopological theory and basic categories of the DOLCE foundational ontology so as to disambiguate the relations. To make this practically usable, we transformed the mappable KGEMT axioms into OWL, added a DOLCE ultra-ultra-light and extended the taxonomy of part-whole OWL object properties, designed an additional activity diagram for the newly added mereotopological relations, added examples, and implemented this in the ONTOPARTS application. ONTOPARTS ensures that the complexities of the underlying theories and languages are hidden from the modeller by means of the automated modelling guidelines and adds the new axioms to the OWL file with a simple one-click button. This ontology-inspired part-whole relation selection tool, ONTOPARTS, can be used with different OWL languages and different ontology development tools. ONTOPARTS was evaluated with modellers, which was found to simplify the task, was performed more accurately, and generated desire to learn more about the theoretical details.

ONTOPARTS and the OWLized hierarchy of relations are freely available in the online material (<http://www.meteck.org/files/ontopartssup/supindex.html>). Current and future work pertains to adding features, such as better interaction with property chaining and language choices in conjunction with the object property characteristics, and handling ABox assertions.

References

1. Artale, A., Franconi, E., Guarino, N., Pazzi, L.: Part-whole relations in object-centered systems: An overview. *DKE* 20(3), 347–383 (1996)
2. Bittner, T., Donnelly, M.: Computational ontologies of parthood, componenthood, and containment. In: *Proc. of IJCAI 2005*, pp. 382–387. AAAI Press, Cambridge (2005)
3. Cohn, A.G., Renz, J.: Qualitative spatial representation and reasoning. In: *Handbook of Knowledge Representation*, ch. 13 p. 551–596. Elsevier (2008)
4. Egenhofer, M.J., Herring, J.R.: Categorizing binary topological relations between regions, lines, and points in geographic databases. Tech. Rep. 90-12, National Center for Geographic Information and Analysis, University of California (1990)
5. Eschenbach, C., Heydrich, W.: Classical mereology and restricted domains. *International Journal of Human-Computer Studies* 43, 723–740 (1995)
6. Grütter, R., Bauer-Messmer, B.: Combining OWL with RCC for spatioterminological reasoning on environmental data. In: *Proc. of OWLED 2007* (2007)
7. Guarino, N., Welty, C.: An overview of OntoClean. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*, pp. 151–159. Springer (2004)
8. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible *SRTOIQ*. In: *Proc. of KR 2006*, pp. 452–457 (2006)
9. Katz, Y., Cuenca Grau, B.: Representing qualitative spatial information in OWL-DL. In: *Proc. of OWLED 2005*, Galway, Ireland (2005)

10. Keet, C.M.: Part-Whole Relations in Object-Role Models. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2006 Workshops. LNCS, vol. 4278, pp. 1116–1127. Springer, Heidelberg (2006)
11. Keet, C.M., Artale, A.: Representing and reasoning over a taxonomy of part-whole relations. *Applied Ontology* 3(1-2), 91–110 (2008)
12. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A.: *Ontology library. WonderWeb Deliverable D18 (ver. 1.0)* (December 31, 2003), <http://wonderweb.semanticweb.org>
13. McGuinness, D.L., van Harmelen, F.: *OWL Web Ontology Language Overview. W3C Recommendation (2004)*, <http://www.w3.org/TR/owl-features/>
14. Mejino, J.L.V., Agoncillo, A.V., Rickard, K.L., Rosse, C.: Representing complexity in part-whole relationships within the foundational model of anatomy. In: *Proc. of the AMIA Fall Symposium*, pp. 450–454 (2003)
15. Motik, B., Grau, B.C., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: *OWL 2 Web Ontology Language Profiles. W3c recommendation, W3C (October 27, 2009)*, <http://www.w3.org/TR/owl2-profiles/>
16. Motik, B., Patel-Schneider, P.F., Parsia, B.: *OWL 2 web ontology language structural specification and functional-style syntax. W3c recommendation, W3C (October 27, 2009)* <http://www.w3.org/TR/owl2-syntax/>
17. Nutt, W.: On the Translation of Qualitative Spatial Reasoning Problems into Modal Logics. In: Burgard, W., Christaller, T., Cremers, A.B. (eds.) *KI 1999. LNCS (LNAI)*, vol. 1701, pp. 113–124. Springer, Heidelberg (1999)
18. Odell, J.J.: *Advanced Object-Oriented Analysis & Design using UML*. Cambridge University Press, Cambridge (1998)
19. Randell, D.A., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. In: *Proc. of KR 1992*, pp. 165–176. Morgan Kaufmann (1992)
20. Schulz, S., Hahn, U., Romacker, M.: Modeling anatomical spatial relations with description logics. In: *AMIA 2000 Annual Symposium*, pp. 779–783 (2000)
21. Stocker, M., Sirin, E.: Pelletspatial: A hybrid RCC-8 and RDF/OWL reasoning and query engine. In: *Proc. of OWLED 2009, Chantilly, USA, October 23-24. CEUR-WS*, vol. 529 (2009)
22. Varzi, A.: Spatial reasoning and ontology: parts, wholes, and locations. In: *Handbook of Spatial Logics*, pp. 945–1038. Springer, Heidelberg (2007)
23. Wessel, M.: Obstacles on the way to qualitative spatial reasoning with description logics: some undecidability results. In: *Proc. of DL 2001, Stanford, CA, USA, August 1-3. CEUR WS*, vol. 49 (2001)
24. Winston, M.E., Chaffin, R., Herrmann, D.: A taxonomy of partwhole relations. *Cognitive Science* 11(4), 417–444 (1987)
25. Yang, W., Luo, Y., Guo, P., Tao, H., He, B.: A Model for Classification of Topological Relationships Between Two Spatial Objects. In: Wang, L., Jin, Y. (eds.) *FSKD 2005. LNCS (LNAI)*, Part II, vol. 3614, pp. 723–726. Springer, Heidelberg (2005)
26. Zhong, Z.-N., et al.: Representing topological relationships among heterogeneous geometry-collection features. *J. of Comp. Sci. & Techn.* 19(3), 280–289 (2004)

Evaluation of the Music Ontology Framework

Yves Raimond¹ and Mark Sandler²

¹ BBC R&D

yves.raimond@bbc.co.uk

² Queen Mary, University of London

mark.sandler@eecs.qmul.ac.uk

Abstract. The Music Ontology provides a framework for publishing structured music-related data on the Web, ranging from editorial data to temporal annotations of audio signals. It has been used extensively, for example in the DBTune project and on the BBC Music website. Until now it hasn't been systematically evaluated and compared to other frameworks for handling music-related data. In this article, we design a 'query-driven' ontology evaluation framework capturing the intended use of this ontology. We aggregate a large set of real-world music-related user needs, and evaluate how much of it is expressible within our ontological framework. This gives us a quantitative measure of how well our ontology could support a system addressing these real-world user needs. We also provide some statistical insights in terms of lexical coverage for comparison with related description frameworks and identify areas within the ontology that could be improved.

1 Introduction

The Music Ontology [19] was first published in 2006 and provides a framework for distributing structured music-related data on the Web. It has been used extensively over the years, both as a generic model for the music domain and as a way of publishing music-related data on the Web.

Until now the Music Ontology has never been formally evaluated and compared with related description frameworks. In this paper, we perform a quantitative evaluation of the Music Ontology framework. We want to validate the Music Ontology with regards to its intended use, and to get a list of areas the Music Ontology community should focus on in further developments.

As more and more BBC web sites are using ontologies [20], we ultimately want to reach a practical evaluation methodology that we can apply to other domains. Those ontologies are mainly written by domain experts, and we would need to evaluate how much domain data they can capture. We would also need to identify possible improvements in order to provide relevant feedback to domain experts.

We first review previous work on ontology evaluation in §2. We devise our evaluation methodology in §3, quantifying how well real-world user-needs fit within our Music Ontology framework. We perform in §4 the actual evaluation, and compare several alternatives for each step of our evaluation process. We discuss the results and conclude in §5.

2 Techniques for Ontology Evaluation

Brewster et al. argue that standard information retrieval or information extraction evaluation methodologies, using the notion of *precision* and *recall*, are not appropriate for ontology evaluation [7]. We need different evaluation methodologies to evaluate knowledge representation frameworks. Ultimately, we need an ontology evaluation metric in order to easily assess ontologies and to track their evolution [23]. In this article, we design such an evaluation metric and apply it to the Music Ontology framework. We review different ontology evaluation methodologies in §2.1, §2.2 and §2.3 and explain why they are not suitable for evaluating our Music Ontology framework. We focus on two evaluation paradigms in §2.4 and §2.5 that constitute the basis of our evaluation methodology.

2.1 Qualitative Evaluation

One way to qualitatively evaluate an ontology is to take a set of users and ask them to rate the ontology according to a number of criteria. The OntoMetric evaluation methodology [15] includes a number of such qualitative metrics. Zhang and Li [24] evaluate two multimedia metadata schemes by asking diverse groups of users to rate usefulness of individual metadata fields according to each generic user task defined by the Functional Requirements for Bibliographic Records (FRBR [17]): finding, identifying, selecting and obtaining.

Qualitative ontology evaluations have value especially when evaluating against intended use but raise several problems. It is difficult to choose the right set of users (they could be ontologists, end-users or domain experts), and it is difficult to find an actual scale on which to rate particular criteria of the ontology (what do we mean by a model being “good”?). We also want our ontology evaluation methodology to be as automatable as possible in order to integrate continuous evaluation within our development and publishing workflow, as suggested in [13]. Each ontology release needs to have a positive impact on the evaluation results. For these reasons we do not consider performing a qualitative evaluation of our Music Ontology framework.

2.2 Structural and Ontological Metrics

A number of ontology evaluation metrics can be derived automatically. Amongst these we distinguish between *structural* and *ontological* metrics [23].

Structural Metrics. Web ontologies are defined through an RDF graph. This graph can be analysed to derive evaluation metrics. These metrics, evaluating the structure of the graph defining the ontology but not the ontology itself, are called structural metrics. For example the AKTiveRank system [1] includes a metric quantifying the average amount of edges in which a particular node corresponding to a concept is involved. This metric therefore gives an idea of how much detail a concept definition in the evaluated ontology holds. Another set of examples are the structural ontology measures defined in [9], including maximum

and minimum depth and breadth of the concept hierarchy. Such metrics do not capture the intended use of the evaluated ontology. We therefore do not consider using structural metrics in our evaluation.

Ontological Metrics. Ontological metrics evaluate the actual models instead of their underlying graph structure. The OntoClean methodology [11] evaluates modelling choices in ontologies from a philosophical stand-point. It defines a number of criteria that need to be satisfied. For example a subsumption relationship cannot be drawn between concepts that have different identity criteria—a time interval cannot be a sub-concept of a time duration. OntoClean relates more to ontology engineering than ontology evaluation [23]. It can be seen as a set of ontology design guidelines. These guidelines were used when designing the Music Ontology and the underlying ontologies [19].

2.3 Similarity to a “Gold-Standard”

If we have access to a “gold-standard” (a canonical model of a particular domain) we can evaluate other ontologies of that domain by measuring their similarities to that canonical model. A set of measures for describing the similarity of different ontologies (both at the lexical and at the conceptual level) is proposed in [16]. We do not have such a gold-standard ontology, so this approach can be dismissed for evaluating our Music Ontology framework.

2.4 Task-Based Evaluation

Another way of evaluating an ontology is to measure its performance on a specific task [18]. A given task is chosen, as well as a corresponding gold-standard for perfect performance. Then, we consider the following errors when trying to fulfill that task in a particular ontology-driven application:

- Insertion errors (some terms in the ontology are not necessary);
- Deletion errors (missing terms);
- Substitution errors (ambiguous or ill-defined terms).

2.5 Data-Driven Ontology Evaluation

Brewster et al. provide a data-driven approach for ontology evaluation [7]. They use a corpus of text within the domain modelled by the ontology. They extract terms from it and try to associate them with terms in the ontology to evaluate, which leads to a measure for the domain coverage of the ontology. In order to evaluate the structure of the ontology, they cluster the extracted terms and quantify the extent to which terms in the same cluster are closer in the ontology than terms in different clusters. Elhadad et al. use a similar methodology to evaluate an ontology in the movies domain against a corpus of movie reviews [8], although they focus on the coverage of ontology instances.

3 A Query-Driven Ontology Evaluation Methodology

We now devise our methodology for evaluating our Music Ontology framework, based on the the data-driven and the task-based evaluation methodologies described in §2.4 and §2.5. We want this evaluation methodology to allow us to validate our ontology with regards to real-world information-seeking behaviours.

We consider evaluating our knowledge representation framework against a dataset of verbalised music-related user needs. We isolate a set of music-related needs drawn from different sets of users, and we measure how well a music information system backed by our knowledge representation frameworks could handle these queries. Our evaluation methodology involves the following steps:

3.1 Step 1 - Constructing a Dataset of Verbalised User Needs

We start by constructing a dataset of verbalised user needs. We perform a similar evaluation process on several datasets of verbalised user queries available online. We can distinguish amongst several communities of users, and our Music Ontology framework might perform differently for each of them. We want to evaluate our ontology for each of these communities.

3.2 Step 2 - Extracting Query Features

We analyse these needs to extract a set of *features* — recurrent patterns used to describe the information need, e.g. “the name of the artist was X” or “the lyrics mentioned Y”. We consider several alternatives for extracting features from our dataset.

- We can use the results of previous works in extracting query features from similar datasets;
- We can extract features from the dataset by following a statistical approach;
- We can manually extract features from a random sample of the dataset.

We also consider extracting a weight w_f for each feature f , capturing the relative importance of f within the dataset. Moreover, these weights are normalised so that their sum is equal to one.

$$w_f = \frac{\text{number of queries that contain the feature } f}{\sum_g \text{number of queries that contain the feature } g} \quad (1)$$

3.3 Step 3 - Computing the Ontology Fit

We now evaluate how well these features map to our knowledge representation framework. The corresponding measure captures the *ontology fit*. The Music Ontology was designed to not duplicate terms that could be borrowed from other web ontologies (for example, `foaf:Person`, `dc:title` or `po:Broadcast`). We take into account this design choice. In the last step of our evaluation process

we therefore also consider terms from FOAF¹, Dublin Core² and the Programmes Ontology³.

We develop an ontology fit measure capturing how well the extracted features can be mapped to our ontology. For a query feature f , we define δ as follows.

$$\delta(f) = \begin{cases} 1 & f \text{ is expressible within the ontology} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Our ontology fit measure for a set of verbalised queries Q is then the weighted sum of the $\delta(f)$ for each feature f extracted from Q .

$$\Delta = \sum_f w_f \cdot \delta(f) \quad (3)$$

The ontology fit measure Δ therefore captures how possible it is to map a set of user needs to queries expressed within our Music Ontology framework. The closer Δ is to one, the more our ontology can be used to express the dataset of user queries. We use the ontology fit measure to validate our ontology with regards to real-world user needs.

3.4 Discussion

This ‘query-driven’ evaluation methodology corresponds to a particular kind of a task-based evaluation methodology, where the task is simply to be able to answer a set of musical queries and the evaluation metric focuses on coverage (the insertion and substitution errors are not considered). The gold-standard associated with this task is that such queries are fully expressed in terms of our knowledge representation framework — the application has a way to derive accurate answers for all of them. This evaluation methodology also corresponds to a particular kind of data-driven evaluation. We start from a corpus of text, corresponding to our dataset of verbalised user needs, which we analyse and try to map to our knowledge representation framework.

A similar query-driven evaluation of an ontology-based music search system is performed by Baumann et al. [4]. They gather a set of 1500 verbalised queries issued to their system, which they cluster manually in five different high-level categories (requests for artists, songs, etc.) in order to get insights on the coverage of their system. We use a similar methodology, although we define a quantitative evaluation measure which takes into account much more granular query features. We also consider automating steps of this evaluation process.

4 Evaluation of the Music Ontology Framework

We want to evaluate our representation framework against a large dataset, holding musical needs drawn from a wide range of users. We consider two main

¹ <http://xmlns.com/foaf/spec/>

² <http://dublincore.org/>

³ <http://www.bbc.co.uk/ontologies/programmes>

categories of users: casual users and users of music libraries. We derive an ontology fit measure for each of these categories.

4.1 Casual Users

We first consider verbalised queries drawn from casual users. We measure how well these queries can be expressed within our Music Ontology framework using our ontology fit measure. We consider the three alternatives mentioned in §3.2 for extracting features from a dataset of verbalised user queries.

Evaluating Against Previous Studies of User Needs. We consider evaluating our knowledge representation framework using previous analysis of casual user needs. Such analysis leads to the categorisation of the query *type* (e.g. queries aiming at identifying a particular musical item, queries aiming at researching a particular aspect of a musical work), of the query *context* (the intended use for the requested information) and of the query *features* (recurrent patterns used to describe the information need). We are especially interested in the categorisation of query features as it leads directly to the results of the second step of our evaluation methodology.

Bainbridge et al. [2] analyse 502 queries gathered from Google Answers⁴. Google Answers allows users to post a particular question, which others can answer. Lee et al. [14] analyse 566 queries from the same source, restricting themselves to queries aiming at identifying a particular recording or a particular artist. Both extract a set of recurrent features in such queries. The extracted features along with their correspondences to Music Ontology terms are summarised in table 4.1.

The corresponding ontology fit Δ is 0.828 for Lee’s analysis and 0.975 for Bainbridge’s analysis.

Such ontology fit measures are arguable. The proper term to choose within the Music Ontology framework for one of these features is highly dependent on its actual context. It might be the case that one of these features is expressible within our framework in one context, but not in another. For example for the “related work” feature “it was a cover of *a*” is expressible, but “it was in the charts at the same time as *a*” is not. The features reported in these studies are too general to provide a solid basis for deriving an ontology fit measure.

Corpus-driven Evaluation. We now perform an evaluation inspired by the data-driven evaluation proposed by Brewster et al. [7] and reviewed in §2.5. We use a statistical analysis on a dataset of user queries to derive information features, and we try to map the results of such an analysis onto Music Ontology terms.

We sample verbalised user needs from both Google Answers and Yahoo Questions⁵. We aggregated the whole Google Answers archive in the music category

⁴ Google Answers archives are available at <http://answers.google.com/>, as the service is no longer running.

⁵ Yahoo Questions is available at <http://answers.yahoo.com/>

Table 1. Comparison of the features identified in [2] and in [14] along with corresponding Music Ontology terms

Features used in queries	Music Ontology term	% of queries containing the feature	
		Bainbridge et al. [2]	Lee et al. [14]
Lyrics	mo:Lyrics	28.9	60.6
Date	event:time	31.9	59.2
Media	mo:Medium	-	44.0
Genre	mo:genre	32.7	35.5
Uncertainty	-	-	30.7
Lyrics description	-	-	30.0
Used in movie/ad	po:track	-	30.0
Gender of artist	foaf:gender	-	20.5
Musical style	event:factor ^a	-	19.8
Artist Name	foaf:name	55.0	19.3
Orchestration	mo:Orchestration	13.5	16.8
Related work	mo:MusicalWork	-	15.9
Lyrics topic	dc:subject	2.6	15.4
Where heard	event:place	24.1	14.7
Affect/Mood	-	2.4	14.0
Musical Work	mo:MusicalWork	35.6	13.6
Used in scene	mo:Signal	-	13.3
Audio/Video example	-	4.4	10.8
Similar	musim:Similarity	4.6	9.2
Tempo	mo:tempo	2.4	7.6
Nationality of music/artist	fb:nationalityNoun	12.5	4.2
Related event	event:Event	-	4.2
Language	dc:language	2.0	3.7
Record	mo:Record	12.2	2.7
Melody description	so:Motif	-	0.7
Label	mo:Label	5.4	0.1
Link	foaf:page	2.9	-

^a To express that a particular performance has a given stylistic influence, we add this influence as a factor of the performance.

(3318 verbalised user needs) and a subset of Yahoo Questions (4805 verbalised user needs). Most user queries include editorial information (artist name, track name etc.), as spotted in previous analyses of similar datasets. When including some information about a musical item this information will most likely be related to vocal parts: singer, lyrics etc. The three most cited musical genres are “rock”, “classical” and “rap”. The queries often include information about space and time (e.g. when and where the user heard about that song). They also include information about the access medium: radio, CD, video, online etc. A large part of the queries include personal feelings, illustrated by the terms “love” or “like”. Finally, some of them include information about constituting parts of a particular musical item (e.g. “theme”).

We could consider the words occurring the most in our dataset as query features and their counts as a weight. However, the same problem as in the ontology fit derived in §4.1 also arises. The Music Ontology term corresponding to one of these features is highly context-dependent. There are two ways to overcome these issues. On the one hand, we can keep our evaluation purely on a *lexical* level. We are particularly interested in such an evaluation because it allows us to include other music-related representation frameworks which are not ontologies but just specifications of data formats, therefore providing some insights for comparison. On the other hand, we can extract underlying topics from our corpus of verbalised user needs, and consider these topics as query features. We therefore move our evaluation to the *conceptual* level.

Evaluation at the Lexical Level. We now derive a measure of the lexical coverage of our ontology. We first produce a vector space representation of these verbalised user needs and of labels and comments within the Music Ontology specification. We first remove common stop words. We then map the stemmed terms to vector dimensions and create vectors for our dataset and our ontology using `tf-idf`. We also include in our vector space other music-related representation frameworks. We finally compute cosine distances between pairs of vectors, captured in table 2.

We first note that the results in this table are not comparable with the ontology fit results derived in the rest of this article. They are not computed using the same methodology as defined in §3. We note that our Music Ontology framework performs better than the other representation framework — it is closer to the dataset of user queries. These results are due to the fact that our ontology encompasses a wider scope of music-related information than the others, which are dedicated to specific use-cases. For example XSPF is specific to playlists, iTunes XML and hAudio to simple editorial metadata, Variations3 to music libraries and AceXML to content-based analysis and machine learning. The lexical coverage of the Music Ontology framework is therefore higher. Of course this measure is very crude and just captures the lexical overlap between specification documents and our dataset of user queries. It can serve for comparison purposes, but not to validate our framework against this dataset.

Evaluation at the conceptual level. We now want to go beyond this lexical layer. We try to extract from our dataset a set of underlying *topics*. We then consider these topics as our query features and compute an ontology fit measure from them by following the methodology described in §3.

We consider that our corpus of musical queries reflects the underlying set of topics it addresses. A common way of modelling the contribution of these topics to the i^{th} word in a given document (in our case a musical query) is as follows.

$$P(w_i) = \sum_{j=1}^T P(w_i|z_i = j) \cdot P(z_i = j) \quad (4)$$

where T is the number of latent topics, z_i is a latent variable indicating the topic from which the i^{th} word was drawn, $P(w_i|z_i = j)$ is the probability of the word

Table 2. Cosine similarities between vectorised specification documents and the casual users dataset. We use labels and descriptions of terms for web ontologies and textual specifications for other frameworks.

Ontology	Similarity
Music Ontology	0.0812
ID3 version 2.3.0	0.0526
hAudio	0.0375
Musicbrainz	0.0318
XSPF	0.026
ACE XML	0.0208
iTunes XML	0.0182
ID3 version 2.4.0	0.0156
Variations3 FRBR-based model, phase 2	0.0112
FRBR Core & Extended	0.0111
MODS	0.0055
MPEG-7 Audio	0.0013

w_i under the j^{th} topic, and $P(z_i = j)$ is the probability of choosing a word from the j^{th} topic in the current document. For example in a corpus dealing with performances and recording devices, $P(w|z)$ would capture the content of the underlying topics. The performance topic would give high probability to words like venue, performer or orchestra, whereas the recording device topic would give high probability to words like microphone, converter or signal. Whether a particular document concerns performances, recording devices or both would be captured by $P(z)$.

The Latent Dirichlet Allocation (LDA) [6] provides such a model. In LDA, documents are generated by first picking a distribution over topics from a Dirichlet distribution which determines $P(z)$. We then pick a topic from this distribution and a word from that topic according to $P(w|z)$ to generate the words in the document. We use the same methodology as in [10] to discover topics.

We use an approximate inference algorithm via Gibbs sampling for LDA [12]. We first pre-process our dataset of musical queries by stemming terms, removing stop words and removing words that appear in less than five queries. Repeated experiments for different number of topics (20, 50, 100 and 200) suggest that a model incorporating 50 topics best captures our data. We reach the set of topics illustrated in table 4.1.

We consider these topics as our query features. For each topic, we use its relative importance in the dataset as a feature weight. We manually map each topic to terms within our Music Ontology framework to derive the ontology fit measure described in §3.3. The corresponding ontology fit measure is 0.723.

However, this measure of the ontology fit is still arguable. Some of the topics inferred are not sufficiently precise to be easily mapped to Music Ontology terms. A subjective mapping still needs to be done to relate the extracted topics with a set of ontology terms. Moreover, some crucial query features are not captured within the extracted topics. For example a lot of queries include an implicit

Table 3. Top words in the first six topics inferred through Latent Dirichlet Allocation over our dataset of musical queries

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6
band	album	song	music	play	live
rock	track	singer	piece	piano	concert
metal	artist	female	classical	note	perform
member	cover	sung	sheet	chord	tour
punk	release	male	composition	key	date
drummer	title	lead	instrument	tuning	ticket
guitarist	name	vocalist	score	scale	opera
quit	purchase	chorus	piano	melody	stage
beatles	bought	artist	orchestra	major	show
zeppelin	obtain	sound	choral	minor	play

notion of uncertainty (such as “I think the title was something like Walk Away”), which is not expressible within our ontology.

A possible improvement to the evaluation above would be to use a Correlated Topic Model [5], which also models the relationships between topics. This would allow us to not only evaluate the coverage of concepts within our ontology, but also the coverage of the relationships between these concepts. It remains future work to develop an accurate measure for evaluating how the inferred graph of topics can be mapped to an ontological framework. Another promising approach for ontology evaluation is to estimate how well a generative model based on an ontology can capture a set of textual data.

Manual Evaluation of the Ontology Fit. We now want to derive a more accurate ontology fit measure. In order to do so, we manually extract the underlying logical structure of these queries and see how well these logical structures can be expressed within our representation framework.

We consider a random sample of 40 queries drawn from the dataset of user needs used in §4.1, corresponding to 0.5% of the queries available within the Google Answers archive.

We manually pre-process every verbalised query q in this sample to extract a set $\alpha(q)$ of query features. These features are recurring logical sentences encoding the queries. In order to minimise the bias in this manual step, we use the following methodology. We use predicates defined within the Music Ontology framework when they exist. When encountering unknown features, we define new predicates using the same FRBR and event-based model as used by the Music Ontology. For example a query holding the sentence “the composer of the song is Chet Baker” would lead to the following two features:

$$\alpha(q) = \{\text{composer}(S, P), \text{name}(P, \text{'Chet Baker'})\}$$

We do not follow exactly the manual data analysis methodology used by Lee et al. [14], which partially structures the original queries by delimiting the parts that correspond to a particular recurrent feature. Indeed, it is important for our

Table 4. Predominant query features in a random sample of the Google Answers dataset, along with their weights and corresponding terms in the Music Ontology framework

Feature	Weight	Corresponding term
title(Item, Title)	0.085	dc:title
maker(Item, Maker)	0.058	foaf:maker
lyrics(Item, Lyrics)	0.054	mo:Lyrics
time(Item, Time)	0.042	dc:date
uncertain(Statement)	0.042	-
similar(Item1, Item2)	0.035	musim:Similarity
based_near(Person, Place)	0.035	foaf:based_near
place(Event, Place)	0.031	event:place

purpose that we extract the whole logical structure of the query. This will lead to a more accurate ontology fit measure (but derived from a smaller dataset) than in the previous sections.

Once these queries have been pre-processed, we assign a weight for each distinct feature. Such weights are computed as described in §3.2. We give the main query features, as well as the corresponding weight and the corresponding Music Ontology term, in table 4. We then compute our ontology fit measure as described in §3.3. We find an ontology fit measure of 0.749.

Discussion. The different variants of our query-driven evaluation made in this section all lead to a similar ontology fit measure. Around 70% of the information held within a dataset of casual user queries is expressible within our Music Ontology framework.

Over the different evaluations made in this section, we found that the main features that are not expressible within our framework are the following.

- Uncertainty - e.g. “I don’t remember if the song had drums in it”;
- Partial characterisation of the lyrics - e.g. “One part of the lyrics was ‘put your hands up’ ”⁶;
- Emotions related to the music itself - e.g. “This song was really sad”;
- Description of related media - e.g. “In the music video, the band was playing in a forest and the singer was trapped under ice” or “The artist was on the cover of that magazine”;
- Other cultural aspects, such as the position of a track in the charts.

Future work on the Music Ontology should therefore focus on these points.

4.2 Users of Music Libraries

Gathering a dataset of music library user needs is difficult. We therefore adapt our approach to cope with the lack of publicly available datasets.

⁶ This particular point is certainly made important by the bias our dataset has towards English-speaking users. For example Baumann [3] reports the case of a user stating “I am not interested in lyrics in general, because my English is too bad to understand something”.

Methodology. Saxton and Richardson [21] present an evaluation methodology for reference services in libraries based on the sampling of real-world questions and on the evaluation of the corresponding transactions on a number of dimensions, including completeness, usefulness, user satisfaction and accuracy. Sugimoto [22] isolates such reference questions in order to evaluate the performance of music libraries. He then analyses the corresponding transactions for a number of music libraries in the United States.

We evaluate our representation framework using a similar methodology. We consider a reference set of queries covering a wide range of possible query types. We then manually extract query features, and follow the process described in §3 to derive an ontology fit measure. We therefore evaluate the performance of the ontology by quantifying how an ontology-backed system would perform if it were occupying the role of the librarian. Such a methodology is similar to the methodology we adopted in §4.1, except that we filter the dataset to leave a small sample of representative questions prior to the actual evaluation instead of using a random sample of questions. The accuracy of such an evaluation is arguable as it does not include information about the predominance of a query feature in a real-world dataset. However, it gives us a measure of how well a representative set of query features is covered by our ontology.

Dataset of User Queries. In order to cope with the lack of data availability for this category of users, we consider re-using the questions selected in Sugimoto’s study [22] from a binder of recorded reference questions asked at the University of North Carolina Chapel Hill Music Library between July 15, 1996 and September 22, 1998. These questions were chosen to cover a typical range of possible questions asked in a music library. These questions are:

1. What is the address for the Bartok Archive in NY?
2. Can you help me locate Civil War flute music?
3. I am a percussion student studying the piece “Fantasy on Japanese Wood Prints” by Alan Hovhaness. I wondered if there was any information available about the actual Japanese wood prints that inspired the composer. If so, what are their titles, and is it possible to find prints or posters for them?
4. Do you have any information on Francis Hopkinson (as a composer)?
5. What are the lyrics to “Who will Answer”? Also, who wrote this and who performed it?

Ontology Fit We start by extracting features from these five queries as in §4.1. We reach a set of query features and associated weights leading to an ontology fit measure of 0.789. Our ontology therefore performs slightly better for this particular dataset than for the casual users dataset. Almost 80% of the information is expressible within our Music Ontology framework. These results can be explained by the diversity of the queries drawn from casual users. For example one query analysed within §4.1 describes in great levels of detail a music video in order to get to the name of a track. Such descriptions are not expressible within our framework and lower the ontology fit.

5 Conclusion

In this article we devised a query-driven evaluation process for music ontologies based on the data-driven and task-based ontology evaluation methodologies. We created a dataset of user queries and measure how well these queries fit within our knowledge representation framework. We end up quantifying how well a system based on our representation framework could help answering these queries.

A number of alternatives can be used for each step of such an evaluation process. First there are several categories of users which are interesting to handle separately as our ontology may perform differently for each. Then there are several ways of performing an analysis of user queries. We summarise in table 5 the results obtained in this article, investigating different alternatives for each of these steps. Our ontology covers more than 70% of the different datasets considered. We identified the main features that are lacking from our ontology in §4.1.

Table 5. Summary of the ontology fit results described in this article

Dataset	Evaluation method	Section	Ontology fit (Δ)
Casual users	Using results of previous analysis	§4.1	0.828 for 14
	Statistical analysis	§4.1	0.975 for 2
	Manual analysis	§4.1	0.723
Music library users	Manual analysis	§4.1	0.749
		§4.2	0.789

We also performed a lexical comparison of different music representation frameworks. This comparison captured how lexically close a particular representation framework is from a dataset of casual user queries. We found that our Music Ontology framework performs better than the others according to this metric.

All the results described in this article evaluate a particular characteristic of our ontology: its coverage of real-world user needs. However, a number of other characteristics would be interesting to capture as well. For example we might want to evaluate the *verbosity* of the ontology – how many ontology terms are needed to express a particular information feature. We might also want to evaluate the *popularity* of the ontology – how many documents reusing Music Ontology terms are available on the Web.

Future work includes using a similar methodology to evaluate other ontologies used within the BBC web site. As those ontologies are mostly built by domain experts we are planning on evaluating how much domain data they can actually capture and use the results of this evaluation to identify possible improvements.

References

1. Alani, H., Brewster, C.: Metrics for ranking ontologies. In: Proceedings of the 4th Int. Workshop on Evaluation of Ontologies for the Web (2006)
2. Bainbridge, D., Cunningham, S.J., Downie, S.J.: How people describe their music information needs: A grounded theory analysis of music queries. In: Proceedings of the 4th International Conference on Music Information Retrieval (2003)
3. Baumann, S.: A music library in the palm of your hand. In: Proceedings of the Contact Forum on Digital libraries for musical audio (Perspectives and tendencies in digitalization, conservation, management and accessibility), Brussels (June 2005)
4. Baumann, S., Klüter, A., Norlien, M.: Using natural language input and audio analysis for a human-oriented MIR system. In: Proceedings of Web Delivery of Music (WEDELMUSIC) (2002)
5. Blei, D.M., Lafferty, J.D.: A correlated topic model of. *The Annals of Applied Statistics* 1(1), 17–35 (2007)
6. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. *The Journal of Machine Learning Research* 3(3), 993–1022 (2003)
7. Brewster, C., Alani, H., Dasmahapatra, S., Wilks, Y.: Data driven ontology evaluation. In: Proceedings of the International Conference on Language Resources and Evaluation, Lisbon, Portugal, pp. 164–168 (2004)
8. Elhadad, M., Gabay, D., Netzer, Y.: Automatic Evaluation of Search Ontologies in the Entertainment Domain using Text Classification. In: Applied Semantic Technologies: Using Semantics in Intelligent Information Processing. Taylor and Francis (2011)
9. Fernández, M., Overbeeke, C., Sabou, M., Motta, E.: What Makes a Good Ontology? A Case-Study in Fine-Grained Knowledge Reuse. In: Gómez-Pérez, A., Yu, Y., Ding, Y. (eds.) ASWC 2009. LNCS, vol. 5926, pp. 61–75. Springer, Heidelberg (2009)
10. Griffiths, T., Steyvers, M.: Finding scientific topics. *Proceedings of the National Academy of Sciences* (2004)
11. Guarino, N., Welty, C.: Evaluating ontological decisions with ONTOCLEAN. *Communications of the ACM* 45(2), 61–65 (2002)
12. Heinrich, G.: Parameter estimation for text analysis. Technical report, University of Leipzig & vsonix GmbH, Darmstadt, Germany (April 2008)
13. Lavbic, D., Krisper, M.: Facilitating ontology development with continuous evaluation. *Informatica* 21(4), 533–552 (2010)
14. Ha Lee, J., Stephen Downie, J., Cameron Jones, M.: Preliminary analyses of information features provided by users for identifying music. In: Proceedings of the International Conference on Music Information Retrieval (2007)
15. Lozano-Tello, A., Gomez-Perez, A.: ONTOMETRIC: A method to choose the appropriate ontology. *Journal of Database Management* 15(2), 1–18 (2004)
16. Maedche, A., Staab, S.: Measuring Similarity between Ontologies. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS (LNAI), vol. 2473, pp. 251–263. Springer, Heidelberg (2002)
17. IFLA Study Group on the Functional Requirements for Bibliographic Records. Functional requirements for bibliographic records - final report. UBCIM Publications - New Series, vol.19 (September 1998), <http://www.ifla.org/VII/s13/frbr/frbr1.htm> (last accessed March 2012)
18. Porzel, R., Malaka, R.: A task-based approach for ontology evaluation. In: Proceedings of the ECAI Workshop on Ontology Learning and Population (2004)

19. Raimond, Y., Abdallah, S., Sandler, M., Giasson, F.: The music ontology. In: Proceedings of the International Conference on Music Information Retrieval, pp. 417–422 (September 2007)
20. Raimond, Y., Scott, T., Oliver, S., Sinclair, P., Smethurst, M.: Use of Semantic Web technologies on the BBC Web Sites. In: *Linking Enterprise Data*, pp. 263–283. Springer (2010)
21. Saxton, M.L., Richardson, J.V.: *Understanding reference transactions*. Academic Press (May 2002)
22. Sugimoto, C.R.: *Evaluating reference transactions in academic music libraries*. Master's thesis, School of Information and Library Science of the University of North Carolina at Chapel Hill (2007)
23. Vrandečić, D., Sure, Y.: How to Design Better Ontology Metrics. In: Franconi, E., Kifer, M., May, W. (eds.) *ESWC 2007*. LNCS, vol. 4519, pp. 311–325. Springer, Heidelberg (2007)
24. Zhang, Y., Li, Y.: A user-centered functional metadata evaluation of moving image collections. *Journal of the American Society for Information Science and Technology* 59(8), 1331–1346 (2008)

The Current State of SKOS Vocabularies on the Web

Nor Azlinayati Abdul Manaf^{1,2}, Sean Bechhofer¹, and Robert Stevens¹

¹ School of Computer Science, The University of Manchester, United Kingdom
{abdulman, seanb, robert.stevens}@cs.man.ac.uk

² MIMOS Berhad, Technology Park Malaysia, 57000 Kuala Lumpur, Malaysia

Abstract. We present a survey of the current state of Simple Knowledge Organization System (SKOS) vocabularies on the Web. Candidate vocabularies were gathered through collections and web crawling, with 478 identified as complying to a given definition of a SKOS vocabulary. Analyses were then conducted that included investigation of the use of SKOS constructs; the use of SKOS semantic relations and lexical labels; and the structure of vocabularies in terms of the hierarchical and associative relations, branching factors and the depth of the vocabularies. Even though SKOS concepts are considered to be the core of SKOS vocabularies, our findings were that not all SKOS vocabularies published explicitly declared SKOS concepts in the vocabularies. Almost one-third of the SKOS vocabularies collected fall into the category of *term lists*, with no use of any SKOS semantic relations. As concept labelling is core to SKOS vocabularies, a surprising find is that not all SKOS vocabularies use SKOS lexical labels, whether `skos:prefLabel` or `skos:altLabel`, for their concepts. The branching factors and maximum depth of the vocabularies have no direct relationship to the size of the vocabularies. We also observed some common modelling slips found in SKOS vocabularies. The survey is useful when considering, for example, converting artefacts such as OWL ontologies into SKOS, where a definition of typicality of SKOS vocabularies could be used to guide the conversion. Moreover, the survey results can serve to provide a better understanding of the modelling styles of the SKOS vocabularies published on the Web, especially when considering the creation of applications that utilize these vocabularies.

1 Introduction

We present a survey of Simple Knowledge Organization System (SKOS) vocabularies on the Web. The aim of this survey is to understand what a typical SKOS vocabulary looks like, especially in terms of the shape, size and depth of the vocabulary structure. We are also interested in determining the extent of usage of SKOS constructs. The results of this survey will equip us with a better understanding of the modelling styles used in the SKOS vocabularies published on the Web. This may be important when considering the creation of an application that utilizes these vocabularies, for example, when converting artefacts such as

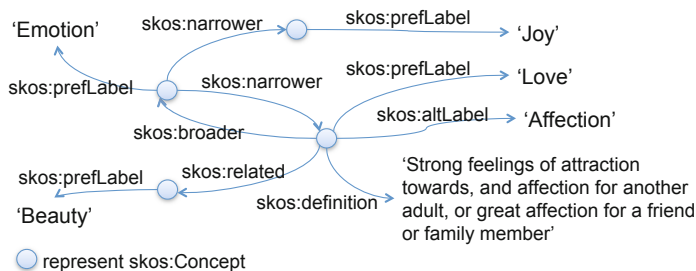


Fig. 1. An example of SKOS constructs usage

OWL ontologies into SKOS, where some typicality of SKOS vocabularies may be useful to guide the conversion.

SKOS¹, accepted as a W3C Recommendation in August 2009, is one of a number of Semantic Web knowledge representation languages. Other such languages include the Resource Description Framework (RDF)², the RDF Schema (RDFS)³, and the Web Ontology Language (OWL)⁴. SKOS is a language designed to represent traditional knowledge organization systems whose representation has weak semantics that are used for simple retrieval and navigation. Such representation includes thesauri, subject headings, classification schemes, taxonomies, glossaries and other structured controlled vocabularies.

The basic element in SKOS is the *concept* which can be viewed as a ‘unit of thought’; ideas, meanings or objects, that is subjective and independent of the term used to label them [1]. These concepts can be semantically linked through hierarchical and associative relations.

Figure 1 shows an example of the usage of SKOS constructs. There are four `skos:Concept` in the figure, representing the concept of *Emotion*, *Love*, *Joy* and *Beauty*. The `skos:broader` and `skos:narrower` properties are used to show that the concepts are hierarchically arranged, while the `skos:related` property is used to show the associative relations between the concepts. SKOS provides three properties for associating lexical labels to conceptual resources; `skos:prefLabel`, `skos:altLabel` and `skos:hiddenLabel`. SKOS documentation properties such as `skos:definition` are used to provide additional textual information regarding the concept.

One of SKOS’ aims is to provide a bridge between different communities of practice within the Library and Information Sciences and the Semantic Web communities. This is accomplished by transferring existing models of knowledge organization systems to the Semantic Web technology context [1]. Knowledge organization system (KOS) is a general term, referring to the tools that present the organized interpretation of knowledge structures [2]. This includes a variety

¹ <http://www.w3.org/2004/02/skos/>

² <http://www.w3.org/RDF/>

³ <http://www.w3.org/2001/sw/wiki/RDFS>

⁴ <http://www.w3.org/2001/sw/wiki/OWL>

of schemes that are used to organize, manage and retrieve information. There are several types of KOS. Hodge [3] groups them into three general categories:

Term lists (flat vocabularies): emphasize lists of terms, often with definitions; e.g., authority files, glossaries, dictionaries, gazetteers, code lists;

Classifications and categories (multi-level vocabularies): emphasize the creation of subject sets; e.g., taxonomies, subject headings, classification schemes; and

Relationship lists (relational vocabularies): emphasize the connections between terms and concepts; e.g., thesauri, semantic networks, ontologies.

We wish to find how many SKOS vocabularies are publicly available for use on the Web and how many fall into one of the listed categories. Additionally, we are interested in learning what the SKOS vocabularies look like in terms of size, shape and depth of the vocabulary structure. We are interested in understanding which of the SKOS constructs listed in the SKOS Reference document [1] are actually being used in the SKOS vocabularies and how often these constructs are used.

Related Work. While research has attempted to characterize Semantic Web documents such as OWL ontologies and RDF(S) documents on the Web [4,5,6], to the authors knowledge there is no attempt at characterizing SKOS vocabularies. Our approach is similar to the work produced by Wang et. al. [4], which focused on OWL ontologies and RDFS documents.

2 Materials and Methods

The steps carried out in this survey are:

1. Preparing a candidate SKOS vocabulary corpus.
2. Identifying SKOS vocabularies.
3. Collecting survey data.
4. Filtering out multiple copies of the same SKOS vocabularies.
5. Analysing the corpus of vocabularies.

Apparatus. All experiments were performed on a 2.4GHz Intel Core 2 Duo MacBook running Mac OS X 10.5.8 with a maximum of 3 GB of memory allocated to the Java virtual machine. Two reasoners were used: JFaCT, which is a Java version of the FaCT++ reasoner, and Pellet. We used the OWL API version 3.2.4⁵ for handling and manipulating the vocabularies.

Preparing a candidate SKOS vocabulary corpus. We employed several methods to gather the candidate SKOS vocabularies to be included in our corpus. First, we gathered vocabularies from two dedicated collections, which are the SKOS

⁵ <http://owlapi.sourceforge.net/>

Implementation Report⁶ and the SKOS/Datasets⁷. We chose the collections as a primary source because the vocabularies listed in these collections were compiled by the SKOS Working Group through its community call. We manually downloaded the vocabularies listed in these collections and stored them locally.

In the second method we utilised Semantic Web search engines such as Swoogle⁸ and Watson⁹. Collecting vocabularies from these sources may enable us to gain some insights into the use of SKOS vocabularies in the community. We made use of the API provided by both search engines to programmatically gather the results from the relevant search. For both search engines, we used `thesaurus`, `skos` and `concept` as search terms. At this point, we collected the URIs of the vocabularies as our analysis tools will retrieve the documents from the Web given the URIs. We also used the Google search engine, using search keywords “skos” and relevant filetypes, which are “.skos”, “.owl”, “.rdf”, “.ttl”, “.nt”, “.n3” and “.xml”. We considered these terms as keywords in identifying SKOS vocabularies because some of them are the terms used as construct names in the SKOS data model.

Our third method used a Web crawler. We used an off-the-shelf web crawler called Blue Crab web crawler¹⁰, which could be configured to crawl based on user specific settings.

Identifying SKOS vocabularies. For the purposes of this survey, we used the following definition of SKOS vocabulary. A SKOS vocabulary is a vocabulary that at the very least contains SKOS concept(s) used directly, or SKOS constructs that indirectly infer the use of a SKOS concept, such as use of SKOS semantic relations.

Each candidate SKOS vocabulary was screened in the following way to identify it as a SKOS vocabulary:

1. Check for existence of direct instances of type `skos:Concept`; if Yes, then accept the vocabulary as a SKOS vocabulary.
2. Check for existence of implied instances of `skos:Concept` due to domain and range restrictions on SKOS relationships (for example the subject of a `skos:broader`, `skos:narrower` or `skos:related` relationship is necessarily a `skos:Concept`); if Yes, then accept the vocabulary as a SKOS vocabulary.
3. Otherwise, do not accept this vocabulary as a SKOS vocabulary.

Consider the following vocabulary snippets written in Manchester Syntax¹¹. Vocabulary 1 and Vocabulary 2 are accepted as SKOS vocabularies based on tests in Step 1 and Step 2, respectively. Meanwhile, Vocabulary 3 is not

⁶ <http://www.w3.org/2006/07/SWD/SKOS/reference/20090315/implementation.html>

⁷ <http://www.w3.org/2001/sw/wiki/SKOS/Datasets>

⁸ <http://swoogle.umbc.edu/>

⁹ <http://kmi-web05.open.ac.uk/WatsonWUI/>

¹⁰ <http://www.limit-point.com/products/bluecrab/>

¹¹ <http://www.w3.org/TR/owl2-manchester-syntax/>

accepted as a SKOS vocabulary according to our definition, even though this vocabulary uses SKOS constructs such as `skos:prefLabel` and `skos:altLabel`.

Vocabulary 1:	Vocabulary 2:	Vocabulary 3:
Individual: Emotion	Individual: Love	Individual: Love
Types:	Types:	Types:
Concept	Thing	Thing
Individual: Love	Facts:	Facts:
Types:	broader Emotion	prefLabel "Love",
Concept		altLabel "Affection"
Individual: Beauty	Individual: Emotion	
Types:	Types:	
Concept	Thing	

Collecting survey data. Since we are interested in both the asserted and inferred version of the SKOS vocabularies, we performed the data recording in two stages; with and without invocation of an automatic reasoner such as Pellet or JFaCT. The data collected without invocation of an automatic reasoner may suggest the actual usage of SKOS constructs in those vocabularies. As for the rest of the analysis, such as identifying the shape and structure of the vocabularies, we need to collect the data from the inferred version of the vocabularies, hence the use of an automatic reasoner.

In the first stage of data collection, no automatic reasoner is invoked, since we are interested in the asserted version of the SKOS vocabularies. For each candidate SKOS vocabulary, we count and record the number of instances for all SKOS constructs listed in the SKOS Reference [1]. We also record the IRI of all *Concept Scheme* present in each SKOS vocabulary.

In the second stage of data collection, we applied a reasoner, and collected and recorded the following for each SKOS vocabulary:

1. Number of SKOS concepts.
2. Depth of each SKOS concept and maximum depth of the concept hierarchy.
3. Total number of links for `skos:broader`, `skos:narrower` and `skos:related` properties.
4. Total number of loose singleton concepts (concepts that are not connected to any other concepts).
5. Total number of root concepts (concepts with only `skos:narrower` relation, but no `skos:broader` relation).
6. Maximum number of `skos:broader` property.

Filtering out multiple copies of the same SKOS vocabularies. We use the recorded information in the previous stage to filter structurally identical SKOS vocabularies. We compare the *Concept Scheme* IRI to search for duplicate vocabularies. For two or more vocabularies having the same *Concept Scheme* IRI, we compare the record for each SKOS construct count. We make a pairwise comparison between each SKOS construct count, taking two vocabularies at a time.

1/ If the two vocabularies have identical records, we then check the content of these vocabularies. This is done by making a pairwise comparison between the instances of `skos:Concept` in one vocabulary to the other. If the two vocabularies have the same instances of `skos:Concept`, then one copy of these vocabularies is kept and the duplicate vocabulary is removed. Otherwise, follow the next step.

2/ If the two vocabularies do not have identical records or identical instances of `skos:Concept`, we assume that one vocabulary is a newer version of the other. We check, if the two vocabularies belong to the same category (either Thesaurus, Taxonomy, or Glossary), then we keep the latest version of the vocabulary and remove the older version. Otherwise, both vocabularies are kept.

Analysing the survey results. In analysing the collected data, we found that some of the vocabularies that are known to be SKOS vocabularies, fail in Step 2 (to be identified as a SKOS vocabulary). We manually inspected these vocabularies. We found several patterns of irregularity in the vocabulary representation and considered them as *modelling slips* made by ontology engineers when authoring the vocabularies. For each type of modelling slip, we decide whether the *error* is intentional or unintentional, and if *fixing* the *error* would change the content of the vocabulary. If the *error* is unintentional and *fixing* the error does not change the content of the vocabulary, then we can apply *fixing* procedures to correct the modelling slips. All fixed vocabularies were included in the survey for further analysis.

We calculated the mode, mean, median and standard deviation for the occurrence of each construct collected from the previous process. The analysis focused on two major aspects of the vocabularies; the usage of SKOS constructs and the structure of the vocabularies. In terms of the usage of SKOS constructs, we analysed which constructs were most used in the SKOS vocabularies. As for the structural analysis of the SKOS vocabulary, we introduced a SKOS metric, \mathcal{M} , with eight tuples as follows:

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{D}, \mathcal{L}, \mathcal{R}, \mathcal{MAX}_B, \mathcal{F}_H, \mathcal{B}_H, \mathcal{F}_A \rangle \quad (1)$$

where \mathcal{S} is the size of vocabulary (represented by the number of SKOS *concepts*), \mathcal{D} is the maximum depth of vocabulary structure, \mathcal{L} is the number of *loose singleton concepts* in the vocabulary, \mathcal{R} is the number of *root nodes* of the vocabulary structure, \mathcal{MAX}_B is the maximum `skos:broader` relation for each *concept* in the vocabulary, \mathcal{F}_H is the average hierarchical forward branching factor, \mathcal{B}_H is the average hierarchical backward branching factor and \mathcal{F}_A is the average associative forward branching factor.

According to [7], *branching factor* is the measure of the number of links coming in to or going out from a particular node. For a directed graph, there are two types of *branching factor*, namely **forward branching factor (FBF)** and **backward branching factor (BBF)**. The *FBF* is the number of arcs or links going out from a node. The *BBF* is the number of arcs or links coming into a node.

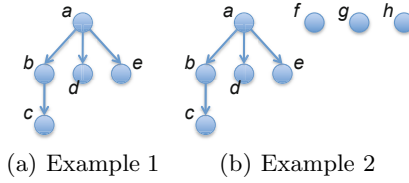


Fig. 2. Example graphs for determining the structure of vocabulary

The *FBF* for hierarchical relations is calculated based on the number of `skos:narrower` relations of a particular concept. Whereas the *BBF* for hierarchical relations is calculated based on the number of `skos:broader` relations of a particular concept. As for associative relations, both *FBF* and *BBF* are calculated based on the number of `skos:related` relations of a particular concept. Since the `skos:related` relation is symmetric, both *FBF* and *BBF* for the associative relation of a particular vocabulary is the same.

We suspect that the *branching factor* values for each concept in a particular SKOS vocabulary are non-uniform. Therefore, we calculated the *average FBF* and *average BBF* for both hierarchical and associative relations. Note that we ignored the *loose singleton concepts* when calculating the *average branching factors*. The *average hierarchical FBF*, F_H , *average hierarchical BBF*, B_H and *average associative FBF*, F_A are given by the following equations:

$$\mathcal{F}_H = \frac{n}{T_n}, \quad \mathcal{B}_H = \frac{b}{T_b}, \quad \mathcal{F}_A = \frac{r}{T_r} \tag{2}$$

where n , b , and r are the total number of `skos:narrower`, `skos:broader` and `skos:related` relations, respectively, and T_n, T_b and T_r are the total number of concepts with `skos:narrower`, `skos:broader` and `skos:related` relations, respectively.

Figure 2 shows two graphs illustrating two different structures of example SKOS vocabulary. Each circle represents a SKOS concept and each directed link between two circles represents a `skos:narrower` relation. The SKOS metric for Figure 2(a) and 2(b) are given by:

\mathcal{M}	\mathcal{S}	\mathcal{D}	\mathcal{L}	\mathcal{R}	\mathcal{MAX}_B	\mathcal{F}_H	\mathcal{B}_H	\mathcal{F}_A
\mathcal{M}_A	5	2	0	1	1	2	1	0
\mathcal{M}_B	8	2	3	1	1	2	1	0

Note that even though the \mathcal{F}_H and \mathcal{B}_H for both examples are the same because each example has the same `skos:narrower` and `skos:broader` relations, the structure of both example is different. However, by looking at the $\mathcal{S}, \mathcal{L}, \mathcal{R}$, we may distinguish the structure of Example 1 from Example 2.

Following the categories of KOS discussed in Section 1, we also defined some rules using the SKOS metric, \mathcal{M} , to categorise the vocabularies in our corpus:

- If all $\mathcal{D}, \mathcal{F}_{\mathcal{H}}, \mathcal{B}_{\mathcal{H}}, \mathcal{F}_{\mathcal{A}} > 0$, then this vocabulary is categorised as a ***Thesaurus***.
- If all $\mathcal{D}, \mathcal{F}_{\mathcal{H}}, \mathcal{B}_{\mathcal{H}} > 0$ and $\mathcal{F}_{\mathcal{A}} = 0$, then this vocabulary is categorised as a ***Taxonomy***.
- If all $\mathcal{D}, \mathcal{F}_{\mathcal{H}}, \mathcal{B}_{\mathcal{H}}, \mathcal{F}_{\mathcal{A}} = 0$, then this vocabulary is categorised as a ***Glossary***.
- If the vocabulary does not belong to any of the above category, then this vocabulary is categorised as ***Others***. For example, the vocabulary uses only associative relation but not hierarchical relations.

3 Result and Observation

We collected 303 candidate SKOS vocabularies from the first method of corpus collection¹². As for the second method, we collected 4220 URIs¹³. We collected 2296 URIs of candidate SKOS vocabularies from the third method¹⁴. This gives a total of 6819 candidate SKOS vocabularies.

After the SKOS vocabulary identification stage, 1068 vocabularies were identified as SKOS vocabularies according to our definition of SKOS vocabulary. The filtering of structurally identical SKOS vocabularies resulted in the exclusion of 603 identical SKOS vocabularies and 11 older versions of SKOS vocabularies, which gave us 454 SKOS vocabularies for further analysis.

Typology of modelling slips Based on our analysis of the collected result, we determined three types of modelling slips as follows:

Type 1: Undeclared property type. Each property used in the vocabulary is not explicitly typed as any of the property types such as `owl:objectProperty`, `owl:dataProperty`, `owl:annotationProperty`, etc. An example of this modelling slip is the use of `skos:broader` or `skos:narrower` properties in the vocabulary without explicit declarations as `owl:ObjectProperty`; such as through an `owl:import` of SKOS core vocabulary. This can cause tooling like the OWLAPI to treat the property as `owl:annotationProperty`. The *fixing* procedure for this type of modelling slip was to add the declarations for SKOS related properties. Applying the *fixing* procedure fixed 18 SKOS vocabularies.

Type 2: Inconsistent. We found 20 SKOS vocabularies were inconsistent. We classify the reasons for inconsistent vocabularies into:

- unintentionally typing an individual that is supposed to be an instance of `skos:ConceptScheme` class to also be an instance of `skos:Concept` class. The *fixing* procedure for this type of modelling slip was to remove the declaration of `skos:Concept` class from the individual. Applying the *fixing* procedure fixed 5 SKOS vocabularies.
- unintentionally using `skos:narrower` construct to relate between a member and its collection. We found that the `skos:member` property was declared in the

¹² This figure is valid as at 8 December 2010.

¹³ as at 2 March 2011.

¹⁴ Note that the web crawler runs for approximately three months, ended on 17 May 2011.

Table 1. Summary results

Stages	vocabs
Corpus preparation:	6819
- 1st method	303
- 2nd method	4220
- 3rd method	2296
SKOS vocabularies identification	1068
- 1st method	143
- 2nd method	432
- 3rd method	21
Structurally identical filtering	454
Fixing modelling slips:	24
- Type 1	18
- Type 2	6
Total SKOS vocabularies	478

Table 2. Exclusions

Exclusion type	vocabs
Plain HTML pages/blogs/forum	2986
Ontologies that are not SKOS vocabularies	1152
File not found	632
Connection refused	511
Network is unreachable	331
Connection timed out	284
Parsing error	266
Actual SKOS Core vocabulary	93
Failed to load import ontology	63
Modelling slips (Type 2 & Type 3)	23 (14 & 9)
Total exclusion	6341

vocabulary but never used. The *fixing* procedure for this type of modelling slip was to replace the `skos:narrower` property with the `skos:member` property to show the relationship between a member and its collection. Applying the *fixing* procedure fixed 1 SKOS vocabulary.

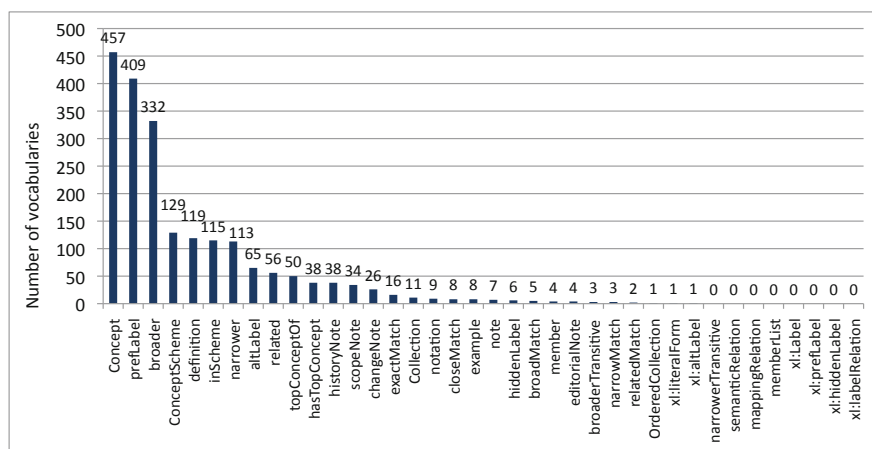
- invalid datatype usage such as invalid `dateTime` datatype, invalid integer datatype, invalid string datatype, etc. This mistake was considered unfixable *errors*. There were 14 SKOS vocabularies excluded from the survey.

Type 3: Use of unsupported datatype. This was issued by the reasoner when encountering user-defined datatype. Note that this is not exactly a modelling slip instead a result from some limitations of the reasoner, thus, hindered us from getting the required data. To deal with this case, we first checked whether the user-defined datatype was actually in use to type the data in the vocabulary. If the datatype was not in use, we excluded the datatype from the datatype list and reclassified the vocabulary. There were 9 vocabularies excluded from the survey.

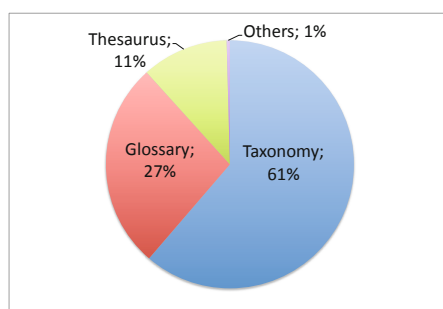
Fixing the modelling slips resulted in 24 additional SKOS vocabularies included in the corpus, which gave us the final number of 478 SKOS vocabularies. The summary figures and reasons for exclusion are presented in Tables 1 and 2. The full results and analysis can be found at <http://www.myexperiment.org/packs/237>.

Figure 3(a) shows the percentage of the SKOS construct usage. For each SKOS construct, a SKOS vocabulary was counted as using the construct if the construct is used at least once in the vocabulary. In this stage, we only counted the asserted axioms in each of the SKOS vocabularies.

Of all the SKOS constructs that are made available in the SKOS Recommendation 1 `skos:Concept`, `skos:prefLabel`, and `skos:broader` are the three most used constructs in the vocabularies, with 95.6%, 85.6% and 69.5%,



(a) Usage



(b) Usage

Fig. 3. Construct Usage and Vocabulary Types

respectively. 28 out of 35 SKOS constructs were used in less than 10% of the vocabularies. There were eight SKOS constructs that were not used in any of the vocabularies.

The rules in Section 2 were used to categorise the vocabularies following the types of KOS as described in Section 1. Figure 3(b) shows a chart representing different types of SKOS vocabulary. As shown in this figure, 61% or 293 of the vocabularies are categorised as *Taxonomy*. The second largest type is *Glossary* with 27% or 129 vocabularies. 11% or 54 vocabularies fell into the *Thesaurus* category.

The remaining 1% or 2 vocabularies were categorised as *Others*. Further inspection revealed that the two vocabularies in the *Others* category are a *Glossary* with 4 `skos:related` properties on 2 pairs of the *concepts* (out of 333 *concepts*) and a snippet of a real SKOS vocabulary intended to show the use of `skos:related` property. We decided to reclassify these two vocabularies into the *Glossary* category for the first one and the *Thesaurus* category for the other.

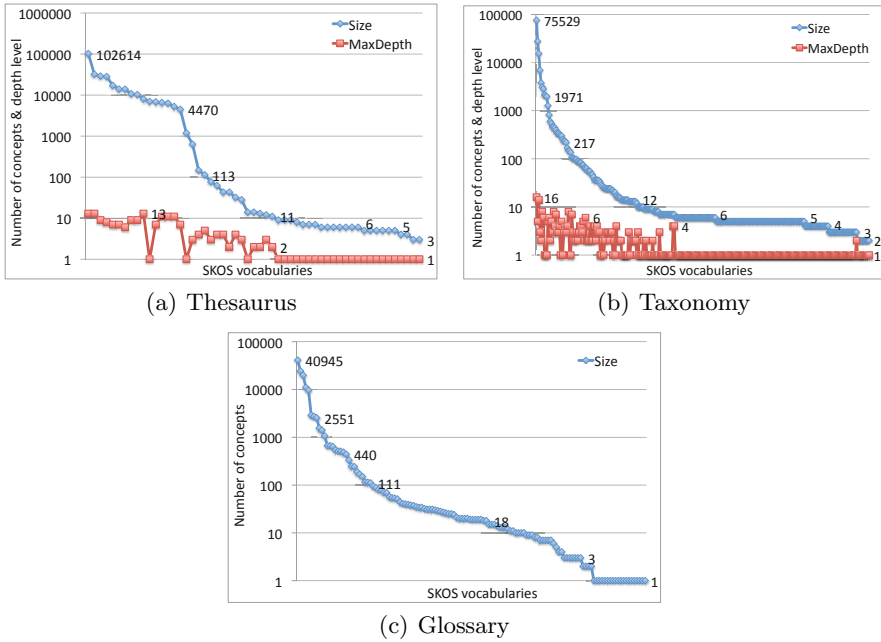


Fig. 4. Size of vocabulary and its maximum depth of vocabulary structure

Figure 4 plots the size of SKOS vocabularies and the maximum depth of the vocabulary structure. Each subgraph represents different categories of SKOS vocabulary; *Thesaurus*, *Taxonomy* and *Glossary*. Within each category, the vocabularies are sorted according to their size in descending order. The size of SKOS vocabulary was calculated based on the number of SKOS concepts in the vocabulary. The maximum depth of vocabulary was calculated based on hierarchical relations; `skos:broader` and `skos:narrower`; in the vocabulary. Figure 4(a) shows that the smallest size of vocabulary for the *Thesaurus* category is 3 concepts and the largest is 102614 concepts. The maximum depth of the vocabulary structure ranged from 1 to 13 levels. For the *Taxonomy* category as shown in Figure 4(b), the smallest size of vocabulary was 2 concepts and the largest was 75529 concepts. The maximum depth of the vocabulary structure ranged from 1 to 16 levels. Figure 4(c) plots size of vocabularies from the *Glossary* category. The smallest size of vocabulary is 1 concept and the largest is 40945 concepts. The maximum depth of the vocabulary structure for the *Taxonomy* category ranged from 1 to 16 levels. Note that there was no maximum depth for the *Glossary* category because there are no hierarchical relations were present in the vocabularies of this category.

Figure 5(a) and 5(b) show the number of loose concepts, root concepts and maximum `skos:broader` relations for each vocabulary structure. For the *Thesaurus* category, the loose concepts ranged from 0 concept (which means all concepts are connected to at least another concept) to 4426 concepts. The root concepts ranged from 1 root to 590 root concepts. The maximum `skos:broader`

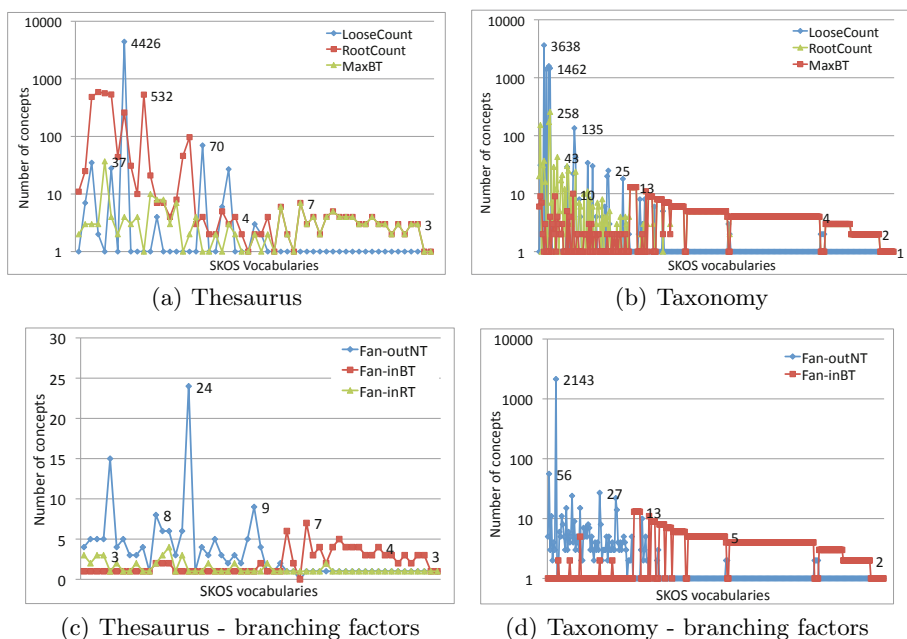


Fig. 5. (a) Number of loose concepts, root concepts, maximum `skos:broader` relations, hierarchical branching factors, and associative branching factors of vocabulary structure

relations ranged from 1 relation to 37 relations. Only 11 out of 54 vocabularies had a single parent (mono-hierarchy) and the rest had at least 2 parent concepts.

As for the *Taxonomy* category, the loose concepts ranged from 0 concept to 3638 concepts. The root concepts ranged from 1 concept to 258 concepts. The maximum `skos:broader` relations ranged from 1 relation to 13 relations. 223 out of 293 vocabularies had more than 1 `skos:broader` relations, which meant that these were poly-hierarchy graphs.

Figure 5(c) and 5(d) show the number of hierarchical and associative branching factor for the *Thesaurus* and *Taxonomy* category. For the *Thesaurus* category, the smallest hierarchical FBF was 1 branch and the largest was 24 branches. The smallest hierarchical BBF was 1 branch and the largest was 7 branches. As for the associative FBF, the smallest was 1 branch and the largest was 4 branches. For the *Taxonomy* category, the smallest hierarchical FBF was 1 branch and the largest was 2143 branches. The smallest hierarchical BBF was 1 branch and the largest was 13 branches.

4 Discussion

As shown in Figure 3(a), of all the SKOS constructs that are available in the SKOS standards, only 3 out of 35 SKOS constructs are used in more than 60%

of the vocabularies. These constructs are `skos:Concept`, `skos:prefLabel` and `skos:broader`. Note that the `skos:Concept` construct usage is not 100%, due to not all vocabularies explicitly typing their individuals as `skos:Concept` in their vocabularies. However, these vocabularies use other SKOS constructs such as semantic relations that infer the individuals as `skos:Concept` due to its domain and range constraints. In some SKOS applications available on the Web like SKOS Reader¹⁵, being able to recognise SKOS concepts is the key to display a SKOS vocabulary. Sometimes, these applications are not equipped with automatic reasoner that could infer the SKOS concepts. Therefore, not explicitly typing the resources as SKOS concepts in a SKOS vocabulary could prevent certain SKOS applications like SKOS Reader from behaving properly.

We found that `skos:broader` relations are used more frequently compared to `skos:narrower` relations, with 69.5% over 23.6% of vocabularies. Note that `skos:narrower` is an inverse relation of `skos:broader`. In those vocabularies where the ontology engineers only specify either one of these relations we think they may be taking advantage of its semantic property to infer the inverse relation. However, we found that 77 vocabularies specified both relations, `skos:broader` and `skos:narrower`, for any pair of concepts that had either relation. 7 of these vocabularies were originated from traditional KOS via some form of conversion process. Various works on converting from traditional KOS to SKOS can be found in [8,9,10,11].

Note also that the `skos:prefLabel` construct is not used in all of the vocabularies. Further analysis revealed that some vocabularies use `rdfs:label` for labelling their concepts. There are 25 out of 35 SKOS constructs used in less than 10% of the vocabularies. Some of these constructs are SKOS mapping properties, which are expected to be used in vocabularies that define alignment to other vocabularies. Other constructs that fell into this portion are mainly the SKOS documentation constructs and SKOS extended (SKOS-XL) constructs. The result also showed that more than 1000 vocabularies excluded from the survey used some of the SKOS constructs such as lexical labelling and documentation constructs.

From the results shown in Figure 3(b), the largest category of 61% of the vocabularies in our corpus are categorised as a *Taxonomy*. All the three categories, *Thesaurus*, *Taxonomy* and *Glossary*, are consistent with the KOS categories.

The results we found in this survey corresponded to some extent to the result of the Controlled Vocabulary Survey¹⁶ conducted by the Semantic Web Company published in June 2011. They conducted an online survey, involving 158 participants which aimed to investigate and learn more about some aspects related to controlled vocabularies. Amongst the foci of interests are; i) preferred knowledge models, ii) main application areas, iii) importance of standards, and iv) trends in organization sizes. The result of their survey revealed that taxonomies and ontologies seem to be the preferred knowledge models. Semantic

¹⁵ <http://labs.mondeca.com/skosReader.html>

¹⁶ http://poolparty.punkt.at/wp-content/uploads/2011/07/Survey_Do_Controlled_Vocabularies_Matter_2011_June.pdf

search, data integration and structure for content navigation are the main application areas for controlled vocabularies. Standards like SKOS have gained greater awareness amongst the participants, which shows that the web-paradigm has entered the world of controlled vocabularies.

The result shows that there is no direct relationship between size of vocabulary and its maximum depth of vocabulary structure. We can see from the graph that the maximum depth of small vocabularies is almost similar to the maximum depth of large vocabularies for both the *Thesaurus* and *Taxonomy* categories.

For the *Thesaurus* category, 43 out of 55 or 80% of the vocabularies have maximum `skos:broader` relations more than one. This means that at least one of the concepts in these vocabularies has more than one *broader concept*, which make them poly-hierarchy graphs. 93% of the vocabularies have more than one root concept, which means that these vocabularies have shape of multi-trees. As for the *Taxonomy* category, 76% of the vocabularies have maximum `skos:broader` relations more than one and 81% of the vocabularies have more than one *root concept*.

As for the hierarchical and associative branching factors result, we found one anomaly to the *Taxonomy* category where one of the vocabularies had depth, \mathcal{D} of 1 and only one *root concept*. One particularly noteworthy value of the metric is that the hierarchical FBF, $\mathcal{F}_{\mathcal{H}}$ is 2143, which is one less than the total vocabulary size, \mathcal{S} , which is 2144. This vocabulary is an outlier, with a a single root node and a very broad, shallow hierarchy. If we were to exclude this vocabulary, the range of hierarchical FBF for the *Taxonomy* category is between 1 and 56. The hierarchical FBF alongside hierarchy depth are important in determining the indexing and search time for a particular query [12].

5 Conclusion

Our method for collecting and analysing SKOS vocabularies has enabled us to gain an understanding of the type and typicality of those vocabularies. We found out that all but two of the SKOS vocabularies that we collected from the Web fell into one of the categories listed by the traditional KOS; flat vocabularies (*Glossary*), multi-level vocabularies (*Taxonomy*) and relational vocabularies (*Thesaurus*). In the future, we plan to select several SKOS vocabularies from each category and study them in more detail in terms of the use and function of the vocabularies in applications.

Based on the results of this survey, a typical taxonomy looks like a polyhierarchy that is 2 levels deep, with a $\mathcal{FBF}_{\mathcal{H}}$ of 10 concepts and a $\mathcal{BF}_{\mathcal{H}}$ of 3 concepts. A typical thesaurus also looks like a polyhierarchy that is 6 level deep, with a $\mathcal{FBF}_{\mathcal{H}}$ of 3 concepts and a $\mathcal{BF}_{\mathcal{H}}$ of 2 concepts, additionally having associative relationships, $\mathcal{FBF}_{\mathcal{A}}$ of 1 concept.

In this survey, we collected 478 vocabularies that according to our definition are SKOS vocabularies. Three years after becoming a W3C Recommendation, the use of SKOS remains low. However, our total of SKOS vocabularies may be artificially low, with some being hidden from our collection method. The

reasons for some of these vocabularies not being publicly accessible by an automated process could be due to confounding factors such as proprietary issue, vocabularies stored within SVN, etc. However, we are confident that we have done our best to deploy various methods in collecting SKOS vocabularies that are publicly available on the Web.

Acknowledgements. Nor Azlinayati Abdul Manaf is receiving the scholarship from Majlis Amanah Rakyat (MARA), an agency under the Malaysian Government, for her doctorate studies. Many thanks to the reviewers who gave insightful comments and suggestion to improve this paper.

References

1. Miles, A., Bechhofer, S.: SKOS simple knowledge organization system reference. W3C recommendation, W3C (2009)
2. Zeng, M.L., Chan, L.M.: Trends and issues in establishing interoperability among knowledge organization systems. *JASIST* 55(5), 377–395 (2004)
3. Hodge, G.: *Systems of Knowledge Organization for Digital Libraries: Beyond Traditional Authority Files*. Council on Library and Information Resources (2000)
4. Wang, T.D., Parsia, B., Hendler, J.: A Survey of the Web Ontology Landscape. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) *ISWC 2006*. LNCS, vol. 4273, pp. 682–694. Springer, Heidelberg (2006)
5. Ding, L., Finin, T.W.: Characterizing the Semantic Web on the Web. 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. 242–257. Springer, Heidelberg (2006)
6. Tempich, C., Volz, R.: Towards a benchmark for semantic web reasoners - an analysis of the DAML ontology library. In: *EON* (2003)
7. Poole, D.L., Mackworth, A.K.: *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press, New York (2010)
8. van Assem, M., Malaisé, V., Miles, A., Schreiber, G.: A Method to Convert Thesauri to SKOS. In: Sure, Y., Domingue, J. (eds.) *ESWC 2006*. LNCS, vol. 4011, pp. 95–109. Springer, Heidelberg (2006)
9. Panzer, M., Zeng, M.L.: Modeling classification systems in SKOS: some challenges and best-practice recommendations. In: *Proceedings of the 2009 International Conference on Dublin Core and Metadata Applications, Dublin Core Metadata Initiative*, pp. 3–14 (2009)
10. Summers, E., Isaac, A., Redding, C., Krech, D.: LCSH, SKOS and linked data. In: *Proceedings of the 2008 International Conference on Dublin Core and Metadata Applications (DC 2008)*, Dublin Core Metadata Initiative, pp. 25–33 (2008)
11. Binding, C.: Implementing Archaeological Time Periods Using CIDOC CRM and SKOS. 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. 273–287. Springer, Heidelberg (2010)
12. Roszkowski, M.: Using taxonomies for knowledge exploration in subject gateways. In: *Proceedings of the 17th Conference on Professional Information Resources, INFORUM 2011* (2011)

The ISOcat Registry Reloaded

Claus Zinn, Christina Hoppermann, and Thorsten Trippel

Department of Linguistics, University of Tübingen, Germany
{claus.zinn,christina.hoppermann,thorsten.trippel}@uni-tuebingen.de

Abstract. The linguistics community is building a metadata-based infrastructure for the description of its research data and tools. At its core is the ISOcat registry, a collaborative platform to hold a (to be standardized) set of data categories (*i.e.*, field descriptors). Descriptors have definitions in natural language and little explicit interrelations. With the registry growing to many hundred entries, authored by many, it is becoming increasingly apparent that the rather informal definitions and their glossary-like design make it hard for users to grasp, exploit and manage the registry's content. In this paper, we take a large subset of the ISOcat term set and reconstruct from it a tree structure following the footsteps of `schema.org`. Our ontological re-engineering yields a representation that gives users a hierarchical view of linguistic, metadata-related terminology. The new representation adds to the precision of all definitions by making explicit information which is only implicitly given in the ISOcat registry. It also helps uncovering and addressing potential inconsistencies in term definitions as well as gaps and redundancies in the overall ISOcat term set. The new representation can serve as a complement to the existing ISOcat model, providing additional support for authors and users in browsing, (re-)using, maintaining, and further extending the community's terminological metadata repertoire.

1 Introduction

The linguistics community has accumulated a tremendous amount of research data over the past decades. It has also realized that the data, being the back-bone of published research findings, deserves equal treatment in terms of archiving and accessibility. For the sustainable management of research data, archiving infrastructures are being built, with metadata-based issues taking center stage. Metadata schemes need to be defined to adequately describe the large variety of research data. The construction of schemas and the archiving of resources will be conducted locally, usually in the place where the research data originated.

To ensure the interoperability of all descriptive means, the ISOcat data category registry has been constructed (see <http://www.isocat.org>). The registry, implementing ISO12620:2009 [4], aims at providing a set of data categories for the description of concepts and resources in various linguistic disciplines (syntax, semantics, *etc.*), but also features a section on metadata terms, which is our primary concern in this paper. Linguists giving a metadata-based description of their research data are solicited to only use metadata descriptors from the

registry. When the registry lacks an entry, researchers are encouraged to extend it by defining new data categories. The registry has a governing body to ensure the quality and merit of all entries submitted for standardization. It is hoped that its grass-root nature helps defining a sufficiently large set of metadata descriptors of which a standardized subset reflects a consensus in a large user base. While the grass-roots approach is appealing, the organization of the registry's content as a glossary of descriptors with little structure is problematic. With the metadata term set now containing 450+ entries, with new entries added regularly, it becomes increasingly hard to browse and manage its content. To address this issue, we propose to re-organise the rather flat knowledge structure into a more systematic, formal and hierarchical representation, following the footsteps of schema.org. The new structure can serve as a complement to the existing one, giving users an alternative and more accessible entry point to the registry.

The remainder of this paper is structured as follows: Sect. 2 gives an account of the ISOcat registry. In Sect. 3, we describe our ontological reconstruction and re-engineering to represent the contents of the glossary by a hierarchically-structured concept scheme. Sect. 4 discusses ontology engineering issues and sketches future work, and Sect. 5 concludes.

2 The ISOcat Data Category Registry

2.1 Specification of Data Categories

The ISOcat data category registry is an implementation of ISO 12620:2009 [4] and accessible by a web-based interface (see <http://www.isocat.org>). The registry's content is currently partitioned into 14 thematic domain groups (TDGs) such as “Metadata”, “Morphosyntax”, “Terminology”, and “Lexical Semantics”, as well as additional project-related work spaces. Each TDG is governed by a decision-making body that considers all requests for the standardization of a data category. Note that the work reported herein is exclusively concerned with the TDG “Metadata”. This group has 458 data categories (DC) [5].

All users have read access to the public parts of the registry. Creators of metadata schemes can make reference to an ISOcat DC by using the entry's persistent identifier. Registered users gain write access to the registry; they can define new entries (becoming owner of the entry) and also modify them at a later stage. DCs owned by other users cannot be modified. New entries get the registration status “private”, but can be proposed for standardization. The ownership of standardized entries is transferred to the TDG's governing body. This policy concentrates curation efforts on the original creators, or the governing bodies; users wanting changes to data categories they do not own have to contact the DC's owners, or add their own, suitable defined, data category to the registry.

A specification of a data category consists of three parts: an administrative part, a descriptive part, and a conceptual domain. The administrative part includes, among others, the DC's registration status (private, candidate, standard),

¹ Accessed December 12, 2011 at <http://www.isocat.org>

origin (name of creator), versioning information (creation date, last change), an English mnemonic identifier, and a persistent identifier. The description section gives a natural language definition of the DC in English. Optionally, it can be complemented by other language sections to give for instance, a DC a French name and a French *definiens*. When more than one DC name is given, one of the names needs to be identified as “preferred name”. Moreover, it is also possible to complement an existing definition section with another one. Thus a DC can be associated with multiple, presumably semantically similar, definitions. The third part gives the conceptual domain of a (non-simple) data category. A DC must take one of four types: complex/closed, complex/open, complex/constrained, and simple.² DCs of type complex/closed have a conceptual domain entirely defined in terms of an enumerated set of values, where each value must be defined as a DC of type simple. DCs of type complex/constrained have a conceptual domain restricted by a constraint given in some constraint language, and a DC of type complex/open can take any value. DCs of type simple are values, and thus do not have a conceptual domain. Each conceptual domain has a mandatory data type, in accordance with those defined by W3C XML Schema. The default data type is “string”, which is also the datatype that is used most in the TDG metadata. Complex/open DCs specify only the datatype as conceptual domain.

Fig. 1 shows an excerpt of the specification of the complex/open DC `/corpusType/`. Its description section shows the DC’s data element name “corpusType”, its English name “corpus type” and its natural-language definition (“Classification of the type of a corpus.”); the DC’s conceptual domain, or value range, is given as a closed set of simple DCs. While `/corpusType/` belongs only to TDG Metadata, a data category can, in general, belong to more than one profile.³

2.2 On Data Category Relationships and Definitions

There is a limited notion of “authorized” relationship in the ISOcat registry. A simple data category (e.g., `/specialisedCorpus/`) can be member of the value domain of a complex/closed data category (e.g., `/corpusType/`). ISO12620:2009 also specifies the possibility that simple data categories can be related to each other via a (single-inheritance) IS-A subsumption relation.⁴ Example entries in the ISOcat registry include, for instance, `/technicalTranslation/ IS-A /translation/`, and `/translation/ IS-A /languageMediation/`.

Relationships between complex data categories, however, are not stored in the DCR. In [2, Slide 26], it is argued that “[R]elation types and modeling strategies for a given data category may differ from application to application” and that the “[M]otivation to agree on relation and modeling strategies will be stronger at individual application level”. It is concluded that the “[I]ntegration of multiple relation structures in DCR itself” (in addition to the ones already present) could lead to “endless ontological clutter”. For the expression of rich

² A fifth type of DCs called “container” is rarely used.

³ In this case, a conceptual domain can be specified for each profile.

⁴ Also see diagram at http://www.isocat.org/12620/model/DCR_data_model_3.svg

corpus type - 1:0	
Key	3822
PID	http://www.isocat.org/datcat/DC-3822
Type	complex/closed
Owner	Hoppermann, Christina
Scope	public
1. Administration Information Section	
1.1 Administration Record	
Identifier	corpusType
Version	1:0
Registration Status	private
Administration Status	private
Justification	Common metadata data category
1.1.1 Creation	
Creation Date	2010-11-26
Change Description	Creation of a new data category.
1.1.2 Last Change	
Last Change Date	2011-06-20
Change Description	Changed source of data element name.
2. Description Section	
Profile	Metadata
2.1 Data Element Name Section	
Data Element Name	corpusType
Source	CMDI
[-] 2.2 English Language Section	
Language	English (en)
2.2.1 Name Section	
Name	corpus type
Name Status	preferred name
2.2.2 Definition Section	
Definition	Classification of the type of a corpus.
Source	NaLiDa
[+] 2.3 German Language Section	
3. Conceptual Domain	
Data Type	string
Profile	Metadata
Value	/comparableCorpus/ (comparable corpus)
Value	/generalCorpus/ (general corpus)

Fig. 1. Excerpt of the ISOcat entry for /corpusType/

relational structures, a *relation registry* should be used. To provide evidence for these claims, the authors of [8] give a modeling example where the data category “noun” is placed in two different conceptualizations.

Naturally, a definition establishes a relation between the term being defined, *i.e.*, its *definiendum*, and its *definiens*. Users of the ISOcat registry are encouraged to follow guidelines when defining new data categories. The DCR Style Guidelines [1, page 3] make reference to ISO-704 [5], and list the following:

- They should consist of a single sentence fragment;
- They should begin with the superordinate concept, either immediately above or at a higher level of the data category concept being defined;
- They should list critical and delimiting characteristic(s) that distinguish the concept from other related concepts.

Since this encodes the notion of *genus-differentia* definitions, it is clear that any set of (related) definitions induces a concept system. Notwithstanding, the DCR Style Guidelines also point out that “concept systems, such as are implied here by the reference to broader and related concepts, should be modeled in Relation Registries outside the DCR.” In line with the policy to disallow (formal) relationships between complex data categories, the DCR guidelines continue saying

“Furthermore, different domains and communities of practice may differ in their choice of the immediate broader concept, depending upon any given ontological perspective. Harmonized definitions for shared DCs should attempt to choose generic references insofar as possible.”

This policy can induce quite some tension or confusion. While the definition of a DC must reference a superordinate concept, it should reference a rather generic than a rather specific superordinate concept. Moreover, superordinate concepts in the *definiens* are referenced with natural-language expressions rather than with formal references (say, by pointing to existing terms of the registry).

In the sequel, we will present a reconstruction of a concept system from the many hundred data category entries and their definitions. This concept system then makes formally explicit the relationships between ISOCat terms and the concepts they denote. The concept system could then be seen as a more formal (and complementary) account of the ISOCat registry; the system could, in fact, be understood as the possible content of a relation registry making explicit all relations between the ISOCat Metadata data categories.

3 Expression of ISOCat.org Using Schema.org

3.1 Building the Skeleton

By December 2011, the TDG Metadata consisted of more than 200 simple DCs, more than 200 complex/open DCs, and less than 50 complex/closed entries. To construct the ontology’s skeleton, we studied the explicitly given relation structures present between complex/closed DCs and its members of the value range, as well as the existing IS-A relations present between simple DCs. We use OWL for all subsequent modeling.

The use of IS-A constructs in the TDG Metadata gives a mixed picture. Fig. 2 shows the IS-A context of the DC `/translation/`. While most of the relationships are subsumptions (*e.g.*, relating `/gisting/` with `/translation/`), others are not (*e.g.*, relating `/projectManagement/` or `/postProjectReview/` with `/translation/`). For the construction of our skeleton, we have only taken the correct use of IS-A relations into account, anticipating that the standardization process of the ISOCat registry will address the issue of its incorrect uses.

The modeling of the relationships between a complex/closed DC and its value range can be modeled by an OWL class description of the “enumeration” type. Reconsider the ISOCat entry `/corpusType/`, which in OWL can be expressed as:

- | | |
|---|---|
| 1. sub DC: <u>/adaptation-1:0/</u> | 16. sub DC: <u>/projectManagement-1:0/</u> |
| 2. sub DC: <u>/alignedText-1:0/</u> | 17. sub DC: <u>/proofreading-1:0/</u> |
| 3. sub DC: <u>/backTranslation-1:0/</u> | 18. sub DC: <u>/scientificTranslation-1:0/</u> |
| 4. sub DC: <u>/computerAssistedTranslation-1:0/</u> | 19. sub DC: <u>/sightTranslation-1:0/</u> |
| 5. sub DC: <u>/editedTranslation-1:0/</u> | 20. sub DC: <u>/sourceText-1:0/</u> |
| 6. sub DC: <u>/gisting-1:0/</u> | 21. sub DC: <u>/specialLanguage-1:0/</u> |
| 7. sub DC: <u>/globalization-1:0/</u> | 22. sub DC: <u>/targetText-1:0/</u> |
| 8. sub DC: <u>/internationalization-1:0/</u> | 23. sub DC: <u>/technicalTranslation-1:0/</u> |
| 9. sub DC: <u>/literaryTranslation-1:0/</u> | 24. sub DC: <u>/terminography-1:0/</u> |
| 10. sub DC: <u>/localization-1:0/</u> | 25. sub DC: <u>/transcreation-1:0/</u> |
| 11. sub DC: <u>/machineTranslation-1:0/</u> | 26. sub DC: <u>/translationEditing-1:0/</u> |
| 12. sub DC: <u>/medicalTranslation-1:0/</u> | 27. sub DC: <u>/translationMemory-1:0/</u> |
| 13. sub DC: <u>/pivotLanguageTranslation-1:0/</u> | 28. sub DC: <u>/translationMemoryTool-1:0/</u> |
| 14. sub DC: <u>/postProjectReview-1:0/</u> | 29. sub DC: <u>/translationQualityAssessment-1:0/</u> |
| 15. sub DC: <u>/pre-translation-1:0/</u> | 30. super DC: <u>/languageMediation-1:0/</u> |

Fig. 2. Subsumption hierarchy for the simple DC `/translation/`

```

<owl:Class rdf:ID="Corpus">
  <rdfs:subClassOf rdf:resource="#Resource"/>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#ComparableCorpus"/>
    <owl:Thing rdf:about="#ParallelCorpus"/>
    <owl:Thing rdf:about="#Treebank"/>    [...]
  </owl:oneOf>
</owl:Class>

```

with the individuals `ComparableCorpus`, `ParallelCorpus`, `Treebank` all being instances of the class. The class `Corpus`, thus, is defined by exhaustively enumerating its instances (and its subclass relationship with `Resource`).

Alternatively, we can model complex/closed DCs using a union construct:

```

<owl:Class rdf:ID="Corpus">
  <rdfs:subClassOf rdf:resource="#Resource"/>
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#ComparableCorpus"/>
    <owl:Class rdf:about="#ParallelCorpus"/>
    <owl:Class rdf:about="#Treebank"/>    [...]
  </owl:unionOf>
</owl:Class>

```

where `ComparableCorpus`, `ParallelCorpus`, `Treebank` are all to be modeled as classes. The latter option is suited when there are relations specific to subclasses of `Corpus`, *i.e.*, with subclasses of `Corpus` as domain or range.

To account for relations implicit in DC definitions, we have been using ISOcat’s search functionality to return all DCs where “type” or “class” occurred in either a DC’s name (26 DCs returned) or its natural-language definition (55 DCs). From those, the DCs given in Table 1 are a good starting point to bootstrap the

Table 1. Central data categories indicating class hierarchy

Data Category	Definition	Example Section
DC-3806 <code>/resourceClass/</code>	Indication of the class, i.e. the type, of a resource.	corpus, lexicon, experiment, tool, grammar, etc.
DC-3822 <code>/corpusType/</code>	Classification of the type of corpus.	Value range: <code>/comparableCorpus/</code> , <code>/generalCorpus/</code> , <code>/subcorpus/</code> , <code>/specialisedCorpus/</code> , <code>/other/</code> , <code>/learnerCorpus/</code> , <code>/monitorCorpus/</code> , <code>/unknown/</code> , <code>/parallelCorpus/</code> , <code>/treebank/</code> , <code>/referenceCorpus/</code>
DC-2487 <code>/lexiconType/</code>	A description of the type of the lexicon.	word list, monolingual dictionary, thesaurus, bilingual dictionary, glossary term base
DC-3871 <code>/experimentType/</code>	Specification of the design type used for the elicitation of experimental data within a research study, especially in the field of psychology.	experimental design, quasi-experimental design, within-subjects design, between-subjects design, mixed design, pretest-posttest design, laboratory experiment, field experiment, etc.
DC-3810 <code>/toolType/</code>	Indication of the type of a tool.	annotation tool, lemmatizer, chunker, segmentation tool, corpus manager, editor, concordancer, tagger, etc.
DC-3900 <code>/writtenResourceType/</code>	The type of the written resource.	primary text, annotation, ethnography, study, etc.
DC-3901 <code>/writtenResourceSubType/</code>	The subtype of the written resource.	dictionary, terminology, wordlist, lexicon, etc. (if written resource type is <code>LexicalAnalysis</code>).

class hierarchy of linguistic resources, despite the fact that their definitions fail to follow the DCR Style Guidelines promoting *genus-differentia* definitions.⁵

Fig. 3 depicts our initial class skeleton that we derived from the DCs given in the table. Its top class (just below **Thing**) stems from the complex/open DC `/resourceClass/`. The elements cited in its example section, however, should

⁵ Two entries have explanation sections adding to their definitions. The explanation section of DC-3900 mentions “type[s] of written resource such as Text, Annotation, Lexical research, Transcription etc”, whereas the respective section of DC-3901 mentions that “[d]ifferent types of written resources have different controlled vocabularies for SubType: the type ‘Lexical research’ has as SubType vocabulary {dictionary, terminology, wordlist, lexicon,...}. In case the Written Resource Type is Annotation the SubType specifies the type of annotation such as phonetic, morphosyntax etc.”

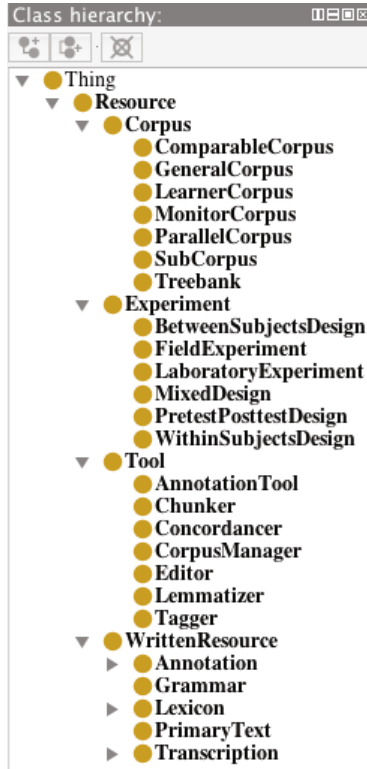


Fig. 3. Class hierarchy for linguistic resources (excerpt)

not always be taken as *direct* subclasses of “Resource”, as the definition of `/writtenResourceType/` and `/writtenResourceSubType/` indicate. In fact, all definitions, when taken together, do not give a clear hierarchical picture. The string “lexicon”, *e.g.*, is mentioned as type of resource in `/resourceClass/`, and also as a subclass to “WrittenResource” (`/writtenResourceSubType/`). Note that “lexicon” and many other strings appearing in the example sections of the complex DCs are not explicitly defined as a DC. And while the related term “LexicalAnalysis” is mentioned in the definition of `/writtenResourceSubType/` as type of written resource, it is not mentioned elsewhere. The situation is similar for the DCs related to experiments. There is no DC with name “experiment”, although `/experimentType/` includes a possible definition. None of the strings in its example section has a corresponding DC. Moreover, there is the DC `/blindExperiment/`, but for technical reasons it is not part of the enumeration.

The situation becomes more complex when attaching properties to the class hierarchy. As the ISOcat registry does not make a distinction between concepts and properties, and given the rather fuzzy definition of many DCs, there are no hard criteria other than modeling expertise to follow (see Sect. 3.3).

```

| Resource: accessProtocol, approach, availability, characterencoding, characterSet, creationdate, creationtool, creatorFullName, completionyear,
deliveryFormat, deploymentTool, derivationDate, derivationMode, derivationTool, derivationType, derivationWorkflow, dialect, distributionType, domain,
dominantLanguage, endposition, geographiccoverage, geocoordinates, harvestingDate, languageid, languageName, languagescript, lastUpdate, license,
licenseType, locationAddress, locationContinent, locationCountry, locationRegion, mainScript, mediaType, medium, metadataCreationDate, metadataCreator,
metadataLanguage, mimetype, modalities, noLanguages, originalSource, positionType, price, publicationDate, region, relationType, resourceClass,
resourceName, resourceTitle, size, sizePerLanguage, sizePerRepLevel, sizeUnit, socialFamilyContext, startPosition, startYear, structuralUnits, subStructureName,
subStructureType, timecoverage, task, temporalClassification, topic, updatefrequency, validation, validationLevel, validationMode, validationType, version,
vocabularysize
| | | Corpus: corpusType, durationOfEffectiveSpeech, durationOfFullDatabase
| | | | ComparableCorpus
| | | | GeneralCorpus
| | | | LearnerCorpus
| | | | MonitorCorpus
| | | | ParallelCorpus
| | | | ReferenceCorpus
| | | | SpecialisedCorpus
| | | | TreebankCorpus
| | | | Experiment: controlGroup, dataRejected, elicitationInstrument, elicitationMethod, elicitationModel, elicitationSoftware, elicitationTimeframe,
elicitationType, experimentInvestigator, experimentName, experimentTitle, experimentType, experimentParadigm, experimentSetting, noParticipants,
population, samplingMethod, surveyPeriod, surveyType, testType, treatmentGroup, variableName, variableType
| | | | BlindExperiment
| | | | | CrossSectionalStudy
| | | | | LongitudinalStudy
| | | | | WizardOfOzExperiment
| | | | Recording: condition, compression, duration, quality, environment, numberOfSpeakers, modalities, recordingPlatformHardware,
recordingPlatformHardware, sampleRate
| | | | | AudioRecording: audioFileFormat
| | | | | VideoRecording
| | | | Tool: api, applicationType, classificationType, displayType, executionLocation, harddiskMin, implementationLanguage, inputParameter,
inputResource, inputType, openSource, outputType, operatingSystem, orchestrator, prerequisiteName, replacesInput, runningEnvironment, schema,
workingMemoryMin
| | | | | AccessTool
| | | | | AnalysisTool
| | | | | AnnotationTool
| | | | | ArchivingTool
| | | | | CreationTool
| | | | | DisplayTool

```

Fig. 4. The ISOcat registry, following the design of `schema.org` (excerpt)

3.2 Re-representation of ISOcat.org (Initial Version)

Fig. 4 depicts one possible hierarchical representation of ISOcat data categories in the proposed new form. It uses the design of `schema.org`, which we found appealing for both its expressive power and simplicity. The new representation gives a structural account of the ISOcat terms. Each concept, relation or instance is linked to the original entry in the ISOcat registry using the DC's persistent identifier. The structure currently accounts for over two thirds of DCs of the TDG Metadata. Those DCs that are not yet included in the hierarchy often have a rather fuzzy definition, not permitting a clear-cut inclusion in the hierarchy.

The benefits of the new representation are rather obvious. Readers get an immediate overview of the different types of linguistic resources and the properties that can be attributed to them. The new representation makes explicit a class hierarchy that is only implicitly present – and scattered – in the ISOcat registry. It differentiates between classes and relations, and attaches the latter to the former. Our re-representation of the TDG Metadata of the ISOcat registry into hierarchical form is preliminary. In the following, we describe the lessons we learned. It is on the maintainers and owners of the data categories to take the lessons into account, and to render their entries more precise and concise.

3.3 Lessons Learned

Rearranging the terms of a glossary into a hierarchy helps to better understand the notion of each term. We hope that our concept scheme informs renewed efforts into achieving a high-quality glossary of linguistic terms.

Lack of Structure. In knowledge representation systems that make use of class hierarchies and inheritance, attributes should be as generic as possible and as specific as required. With the ISOcat registry being designed as a glossary rather than a concept system, information that is usually inherited from superclasses needs to be encoded by extra data category entries. Consider, for instance, the many data categories defined for naming different kinds of entities:

Data Category	Definition
DC-2536 /projectName/	A short name or abbreviation of the project that led to the creation of the resource or tool/service.
DC-2544 /resourceName/	A short name to identify the language resource.
DC-2577 /participantName/	The name of the person participating in the content of the recording as it is used by others in the transcription.
DC-2512 /creatorFullName/	The name of the person who was participating in the creation project.

Similar cases are /experimentName/ (DC-3861), /scriptName/ (DC-3809), /substructureName/ (DC-3820), /countryName/ (DC-3792), /continentName/ (DC-3791), /variableName/ (DC-3880), /prerequisiteName/ (DC-3805), /participantFullName/ (DC-2556), /languageName/ (DC-2484), and /contactFullName/ (DC-2454). While these entries, and others, bear no formal relation to each other in a glossary, from a “concept system” point of view, they should. Following our re-engineering of the ISOcat glossary into an ontology, we would – similar to `schema.org` – attach a general-purpose relation `name` to the top class. With this modeling, the aforementioned DCs can be formally related to the general-purpose `name` relation, defining a property hierarchy:⁶

```
<owl:DatatypeProperty rdf:id="personName">
  <rdfs:subPropertyOf rdf:resource="#name"/>
</owl:DatatypeProperty>
```

DCs related to dates and sizes can be dealt with similarly. The DC /creationDate/ could be made a sub-property of a (new) DC /date/, and in turn, the existing DCs /derivationDate/, /metadataCreationDate/, /publicationDate/ *etc.* can be made sub-properties of /creationDate/. All properties of this type should inherit from /date/ its range `Date` from the XML Schema datatype standard. Similarly, if the DC /size/ (DC-2580) is regarded as general-purpose relation that can be associated with any type of resource, then /vocabularySize/ (DC-2504), /sizePerLanguage/ (DC-2581) and /sizePerRepresentativeLevel/ (DC-2582) could be defined as its sub-properties.

⁶ The DC /description/ (DC-2520) is similar in generality to `name`; it is defined as “a description in general prose text of the issues that are indicated by the context. The description field can occur at many different places in a component and profile.”

Lack of Precision. An analysis of the ISOCat content shows a sloppy use of language when defining DC entries.

Naming Policies. The TDG Metadata of the ISOCat registry has many DCs to name entities. While some of these DC terms carry “name” in their name (*e.g.*, `/personName/`, `/projectName/`), others do not:

Data Category	Definition
DC-3793 <code>/cooperationPartner/</code>	Naming of the cooperation partner of a research project.
DC-2522 <code>/funder/</code>	Name of the funder of the project.

Naming is also an issue when considering other DC subsets such as the pair DC-2568 `/environment/` (“description of the environmental conditions under which the recording was created.”) and DC-2696 `/recordingenvironment/` (“the environment where the recording took place”). Here, the name for DC-2568 should be made more specific, say “recordingCondition”. Also, for usability reasons, a general policy for naming DCs is advisable and needs to be enforced.

Non-adherence to genus-differentia definitions. The entries given in Table [II](#) and the many other examples we have given, show that many authors fail to give definitions in line with ISOCat’s advocated policy. It is advisable that a TDG’s governing body sensitize newly registered users to the importance of good definitions. Sometimes, a user will have difficulty to choose from a pair of semantically similar DCs: while the DC `/address/` is given as “the address of an organization that was/is involved in creating, managing and accessing resource or tool/service” the DC `/locationAddress/` is defined as “the address where the resource was created or originated”. Both DCs follow the principle of *genus-differentia*, but the language used is too imprecise to draw a distinction.

Typing. The complex/closed DC-2548 `/anonymizationFlag/` is of datatype boolean; however its value range comprises `/true/` (DC-2952), `/false/` (DC-2953), `/unspecified/` (DC-2592), and `/unknown/` (DC-2591), whereas the XML Schema boolean datatype can only take the values “true” and “false”.

Incompleteness. With the class hierarchy giving a birds-eye view on the ISOCat glossary, several gaps can be easily spotted. For many of the main classes, there are no corresponding entries in the ISOCat registry. Entries are missing, for instance, for “resource”, “lexicon”, “corpus”, “experiment” *etc.* although references are made to them in the definition and example sections of many DCs.

There are many minor gaps. There is, for instance, the DC-2689 `/audiofileformat/`, but there are no corresponding DCs for “videoFileFormat”, “documentFileFormat” *etc.* Moreover, there are DCs of type complex/open but their type could be complex/closed. DC-2516 `/derivationMode/`, for instance, could easily be closed by adding the simple DC “semi-automatic” to the existing values “manual” and “automatic”.

There are cases where a data category's association with its profiles is incomplete. There is, for instance, DC-2008 `/languageCode/`; it is only associated with the TDG Morphosyntax. Instead of also associating this DC with the TDG Metadata, users have created yet another, but conceptually identical DC, namely `/languageId/` (DC-2482), and have associated it with the TDG Metadata.

Usage of Existing Standards. There is a fair share of data categories that refer to general metadata-related concepts rather than specific linguistic ones. Mapping these DCs to a hierarchy shows that it could share substantial parts with `schema.org`. This includes descriptors that are widely used across many domains, such as metadata about persons, organizations, and places, but also software applications.⁷ Take the class <http://schema.org/PostalAddress>, for instance. It serves as an anchor point to address-related properties, most of which with near equivalents in the ISOcat registry: `/locationAddress/` (DC-2528), `/locationRegion/` (DC-2533), `/locationCountry/` (DC-2532), `/locationContinent/` (DC-2531), `/email/` (DC-2521) and `/faxNumber/` (DC-2455). The DC `/address/` (DC-2505) could then be seen as relation with domain `Person` or `Organization`, and range `PostalAddress`.

The designers of `schema.org` propagate the usage of ISO standards whenever possible. While the ISOcat registry already advises to use language (ISO 639-X) and country codes (ISO 3166-1), it could also profit from the inclusion of additional, and widely known, standards. For linguistic resources, the ISO standard ISO-8601 [3] on dates and times is particularly interesting for the description of segments and their duration (intervals) from recordings, transcriptions, annotations *etc.* The current term set on experimental data would profit from consulting (and referring to) an existing ontology for scientific experiments [7]. The ISOcat entries related to media types, file formats, programming languages, software platforms should make reference to existing knowledge sources such as the IANA registry for MIME media types⁸, the Wikipedia list on file formats⁹, programming languages¹⁰, and operation systems¹¹.

Moreover, the Semantic Web community, including `schema.org`, is encouraged to give an RDF representation for those lists with persistent identification.

Actions to be Taken. The re-representation of the ISOcat registry of metadata terms unveiled a number of issues that need to be addressed. The most pressing issue is the DC authors' ignorance of recommended good practise when defining entries. Naming conventions and the use of *genus-differentia* need to be enforced. An enforcement will prompt users to add those entries to the registry that we highlighted as gaps, and others. Much can be gained by grouping together DCs that are not explicitly linked together by ISOcat's subsumption relation or the

⁷ For the description of software in `schema.org`, please see

<http://www.google.com/support/webmasters/bin/answer.py?answer=1645432>.

⁸ See <http://www.iana.org/assignments/media-types/index.html>

⁹ See http://en.wikipedia.org/wiki/List_of_file_formats

¹⁰ See http://en.wikipedia.org/wiki/Lists_of_programming_languages

¹¹ See http://en.wikipedia.org/wiki/List_of_operating_systems

relationships between complex/closed DCs and the simple DCs of their value range. Moreover, there is ample opportunity to connect to existing ontologies instead of inventing terminology anew.

We believe that our re-representation addresses these issues. It has the potential to serve the goals of the ISOcat user community; it adds to the precision of the ISOcat metadata-related content and groups together entries that are semantically related; its hierarchical structure gives users a birds-eye view to better access and manage a large repertoire of expert terminology.

4 Discussion

4.1 Expert Vocabulary: From Glossary to Ontology

The ISOcat registry is designed as a glossary of terms, and this design can quickly be understood by a large user base without expertise in knowledge representation. Users can easily define a data category whenever they believe such an entry is missing. ISOcat's ease-of-use is also its fundamental shortcoming, however. The definition of a DC is given in natural language, and hence, is inherently vague and often open to multiple interpretations. Also, ISOcat entries vary in style and quality, given the collaborative authoring effort. The increasing size of the TDG Metadata, now containing more than 450 terms, its glossary-like organization, the current data curation policy of the registry – authors can only modify the entries they own – may prompt users to rashly define their own data category instead of identifying and re-using an appropriate existing one. Nevertheless, it is hoped that a standardization process, once set in motion, will lead to an expert vocabulary most linguists agree upon.

It is clear that the definitions of the ISOcat metadata terms spawn a concept system. Simple DCs are related to complex DCs because they appear in the value range of the latter, and it is also possible to define subsumption relations between simple DCs. Moreover, *genus-differentia* definitions relate to each other *definiendum* and *definiens*. The non-adherence of authors to good practise when defining, potentially prompted by a policy that disallows formal relationships between complex DCs, is responsible for many of the weaknesses identified.

It is argued that relationships between complex DCs should be represented in a *relation registry* [8]. DCR authors are encouraged to keep the definitions of their entries deliberately vague so that this vagueness can then be addressed – in varying manner – externally by using the relation registry. While the relation registry is currently used for the mapping of terms from *different* TDGs or vocabularies (using SKOS-like relation types such as *broader* and *narrower*, see [6]), we find it questionable whether this is a viable approach for intra-vocabulary mapping within the TDG Metadata. It would be unclear, *e.g.*, how to draw the line between explicitly and implicitly defined relations in the ISOcat data category registry and those defined in the relation registry, and the possible confusion it creates when the registries' content is in contradiction to each other.

In fact, the concept scheme we derived from our analysis could be seen as an incarnation of the relation registry. But in light of the previous discussion, it must be an officially sanctioned one, aiming at giving an adequate account of ISOcat metadata-related content.

4.2 Impact on Existing Metadata Infrastructure

The concept scheme can serve as a tool to better browse and manage the ISOcat term registry for metadata. It can inform curation efforts to render precise the definition of existing entries, or to create new entries to fill the gaps made obvious by our ontological reengineering. For this, the concept scheme and the ISOcat registry need to be synchronized. This can be achieved by enforcing the policy that authors of new DCs must somehow provide anchor points that link a DC to a node in the hierarchy. Reconsider the entry `/resourceClass/` (cf. Table II, page 7). It could be “semantically enriched” by making explicit the class hierarchy that is only implicitly given in the informal language definition of the entry: `Resource is a class. Corpus is a subclass of Resource. Lexicon is a subclass of Resource etc.`

The semantic enrichment of the DC’s definition could then prompt users to create entries for “corpus”, “lexicon”, “experiment” *etc.* Alternatively, and more in line with common usage in many dictionaries, users could be encouraged to associate the term being defined with broader, narrower, or related terms.

We hope that our concept scheme serves as a starting, reference and entry point to the content of the ISOcat metadata-related vocabulary. For this, it needs to be “in sync” but also officially sanctioned to better reflect, at any given time, the content of the ISOcat registry. Our concept scheme, when understood as a “relation registry”, has the advantage that – by following schema.org and its OWL version (see <http://schema.org/docs/schemaorg.owl>) – it is based on existing, open, and widely-used W3C standards. Future work will address how to best profit from this technology in terms of sharing vocabulary with schema.org and distributing metadata about linguistic resources using microformats.

5 Conclusion

The ISOcat registry has taken a central role in those parts of the linguistics community that care about metadata. Its low-entry barrier allows users to contribute towards a set of terms for the description of linguistic resources. The ISOcat registry will continue to serve this role, but the registry and its users can profit from the provisions we have outlined. With the re-representation of the ISOcat metadata registry into a hierarchical structure, we have gained a birds-eye view of its content. Our work unveiled current shortcomings of the ISOcat registry from a knowledge representation perspective, where class hierarchies are often constructed centrally and in a systematic and top-down manner.

Many of the problems that we have highlighted are typical for distributed work on a lexicographic resource; here, contributors often take a local stance asking

whether a glossary contains a certain term suitable for some given application of the term, or not. With a glossary growing to many hundred entries, it is not surprising that there will be two or more entries denoting the same concept (synonymy), or two entries sharing the same data category name having different (homonymy) or only partially overlapping (polysemy) meanings, *etc.*

A large part of our critique could be addressed by pointing out the “private” nature of the DCs in the TDG Metadata. Once the standardization process of the data categories gains traction, many of the issues can be addressed and solved. We believe that our ontological approach would greatly support this process. DCs owners are encouraged to consult our formalization and check whether their entries can be improved by the birds-eye view now at their disposition. The standardization body is encouraged to take our ontology to identify “important” DCs and schedule them for standardization. For users of the registry, it serves as efficient access method complementing existing search and browse functionality.

Our hierarchy is one of possibly many interpretations of the ISOCat metadata registry. With on-going work on the registry, it will need to be revised accordingly. Note that we do not seek to replace the TDG Metadata with the ontology we have reconstructed by interpreting its content. It is intended to support existing workflows in order to obtain an ISOCat-based metadata repertoire that progresses towards the completeness and high-quality of its entries.

The url <http://www.sfs.uni-tuebingen.de/nalida/isocat/> points to the current version of the hierarchy. Feedback is most welcome!

References

1. DCR Style Guidelines. Version “2010-05-16”, <http://www.isocat.org/manual/DCRGuidelines.pdf> (retrieved December 5, 2011)
2. Data Category specifications. Clarin-NL ISOCat workshop (May 2011), <http://www.isocat.org/manual/tutorial/2011/ISOCat-DC-specifications.pdf> (retrieved December 5, 2011)
3. Int’l Organization of Standardization. Data elements and interchange formats – Information interchange – Representation of dates and times (ISO-8601), Geneva (2009)
4. Int’l Organization of Standardization. Terminology and other language and content resources - Specification of data categories and management of a Data Category Registry for language resources (ISO-12620), Geneva (2009)
5. Int’l Organization of Standardization. Terminology work – Principles and methods (ISO-704), Geneva (2009)
6. Schuurman, I., Windhouwer, M.: Explicit semantics for enriched documents. What do ISOCat, RELcat and SCHEMACat have to offer? In: Proceedings of Supporting Digital Humanities, SDH 2011 (2011)
7. Soldatova, L.N., King, R.D.: An ontology of scientific experiments. Journal of the Royal Society Interface 3(11), 795–803 (2006)
8. Wright, S.E., Kemps-Snijders, M., Windhouwer, M.A.: The OWL and the ISOCat: Modeling Relations in and around the DCR. In: LRT Standards Workshop at LREC 2010, Malta (May 2010)

SCHEMA - An Algorithm for Automated Product Taxonomy Mapping in E-commerce

Steven S. Aanen, Lennart J. Nederstigt, Damir Vandić, and Flavius Fräsincár

Erasmus Universiteit Rotterdam

P.O. Box 1738, NL-3000 DR

Rotterdam, The Netherlands

{steve,lennart}@student.eur.nl, {vandic,frasincar}@ese.eur.nl

Abstract. This paper proposes SCHEMA, an algorithm for automated mapping between heterogeneous product taxonomies in the e-commerce domain. SCHEMA utilises word sense disambiguation techniques, based on the ideas from the algorithm proposed by Lesk, in combination with the semantic lexicon WordNet. For finding candidate map categories and determining the path-similarity we propose a node matching function that is based on the Levenshtein distance. The final mapping quality score is calculated using the Damerau-Levenshtein distance and a node-dissimilarity penalty. The performance of SCHEMA was tested on three real-life datasets and compared with PROMPT and the algorithm proposed by Park & Kim. It is shown that SCHEMA improves considerably on both recall and F₁-score, while maintaining similar precision.

Keywords: schema mapping, e-commerce, lexical matching, word sense disambiguation.

1 Introduction

In recent years the Web has increased dramatically in both size and range, playing an increasingly important role in our society and world economy. For instance, the estimated revenue for e-commerce in the USA grew from \$7.4 billion in 2000 to \$34.7 billion in 2007 [10]. Furthermore, a study by Zhang et al. [25] indicates that the amount of information on the Web currently doubles in size roughly every five years. This exponential growth also means that it is becoming increasingly difficult for a user to find the desired information.

To address this problem, the Semantic Web was conceived to make the Web more useful and understandable for both humans and computers, in conjunction with usage of ontologies, such as the GoodRelations [9] ontology for products. Unfortunately, as it stands today, the vast majority of the data on the Web has not been semantically annotated, resulting in search failures, as search engines do not understand the information contained in Web pages. Traditional keyword-based search cannot properly filter out irrelevant Web content, leaving it up to the user to pick out relevant information from the search results.

Search failures manifest themselves in e-commerce as well [23]. In addition, more than half of the surveyed users in the aforementioned study on online shopping in the USA [10], have encountered various frustrations when shopping online. Due to the absence of Web-wide faceted product search, it is difficult to find the product which satisfies the user's needs best. Users switch between Web-wide keyword-based search results and price comparison tools to find the 'best' product. As this is a time-consuming process, prices are often the determining factor for a purchase. This is an unwanted situation for both buyer and seller: the buyer might like a more expensive product, because it suits his needs better, whereas the seller would like to be able to differentiate his offering on other characteristics than pricing alone. The solution would be to realise a uniform presentation of Web product information, which requires the data to be annotated and structured. A method for the aggregation of data from Web stores is to use the existing hierarchical product category structure: the product taxonomy. By matching the product taxonomies from different Web stores, it becomes easier to compare their products. This should contribute towards solving the search problems encountered by users when shopping online.

In this paper we introduce the *Semantic Category Hierarchy for E-commerce Mapping Algorithm* (SCHEMA), to be used for mapping between heterogeneous product taxonomies from multiple sources. It employs *word sense disambiguation* techniques, using WordNet [17], to find synonyms of the correct sense for the category name. Furthermore, it uses lexical similarity measures, such as the *Levenshtein distance* [14], together with structural information, to determine the best candidate category to map to. In order to evaluate SCHEMA, its performance is compared on recall and precision with PROMPT [19] and the algorithm proposed by Park & Kim [20].

The structure of the paper is as follows. In Sect. 2 related work is reviewed. Section 3 presents SCHEMA, our framework for taxonomy mapping. Section 4 discusses the evaluation results of SCHEMA, compared to existing approaches. Last, in Sect. 5, we give our conclusions and suggest future work.

2 Related Work

The field of taxonomy or schema mapping has generated quite some interest in recent years. It is closely related to the field of ontology mapping, with one important difference: whereas for matching of taxonomies (hierarchical structures), and schemas (graph structures), techniques are used that try to guess the meaning implicitly encoded in the data representation, ontology mapping algorithms try to exploit knowledge that is explicitly encoded in the ontologies [22]. In other words, due to the explicit formal specification of concepts and relations in an ontology, the computer does not need to guess the meaning. In order to interpret the meaning of concepts in an ontology or schema, algorithms often exploit the knowledge contained in generalised *upper ontologies*, such as SUMO [18] or WordNet [17]. In this way the semantic interoperability between different ontologies is enhanced, facilitating correct matching between them. The

semantic lexicon WordNet plays a specifically important role in many mapping algorithms, helping to overcome the ambiguity occurring in natural language, often in combination with word sense disambiguation approaches, such as the approach of Lesk [213]. In addition to the usage of upper ontologies for producing the mappings between ontologies and schemas, lexical similarity measures are also often used. Using lexical similarity measures helps algorithms to deal with slight lexical variations in words. The Levenshtein distance [14] is known as the edit distance, and has been augmented to allow for transposition of characters in the Damerau-Levenshtein distance [4], both utilised in our algorithm.

In their algorithm for product taxonomy mapping, Park & Kim [20] propose to use a disambiguation technique in combination with WordNet to obtain synonyms for a category name, in order to find candidate paths for matching. The candidate paths are assessed using co-occurrence and order consistency, which evaluate the overlap and the order of the categories between the source and candidate path, respectively. While specifically targeted at e-commerce, some phenomena that occur frequently in product taxonomies are neglected, such as composite categories, in which multiple concepts are combined. Various other (database) schema matching approaches exist. SimilarityFlooding [16] uses the similarity between adjacent elements of schema entities to score possible mappings, but does not take the frequently occurring terminological variations, applicable to e-commerce, into account. COMA++ [1] provides a collection of simple matching algorithms and combinations of these. Some approaches use class attribute data for matching, such as S-Match [8] and CUPID [15]. A good overview of existing approaches has been made in recent surveys for schema matching [5,21,22].

PROMPT [19] is a general-purpose ontology mapping tool, which uses predefined (automatic or manual) mappings, called *anchors*, as guidelines for the mapping of similar nodes. However, due to its emphasis on mapping ontologies in general, it fails in matching many categories when employed for product taxonomy mapping. H-Match [3] uses WordNet for determining the correct contextual and linguistic interpretation of concepts, combined with semantic ontology data. Yu et al. [24] propose to use an upper ontology, in order to create a semantic bridge between various e-commerce standards. QOM [7] uses only simple similarity measures, aiming to reduce time complexity, without significant loss of accuracy. Ehrig & Sure [6] propose a rule-based approach, combined with neural networks. Other approaches are discussed in recent surveys for ontology mapping [11].

3 SCHEMA

This section discusses the SCHEMA framework, together with all the assumptions for our product taxonomy matching algorithm. Figure 1 illustrates the high-level overview of the framework. This sequence of steps is executed for every category in the source taxonomy. First, the name of the source category is

disambiguated, to acquire a set of synonyms of the correct sense. This set is used to find candidate categories from the target taxonomy, and is needed to account for the varying denominations throughout taxonomies. After the *Candidate Target Category Selection*, every candidate category path is compared with the path of the source category, by means of the *Candidate Target Path Key Comparison*. The best-fitting candidate target category is selected as the winner. The objective of SCHEMA is to map source categories to a selected target category, if and only if, all products in the source categories fit in the selected target category. This reflects our definition of a successful and meaningful category mapping. First, the general assumptions — the basis for the development of SCHEMA — are explained. Next, each step of the framework, as shown in Fig. 1, will be discussed in more detail.

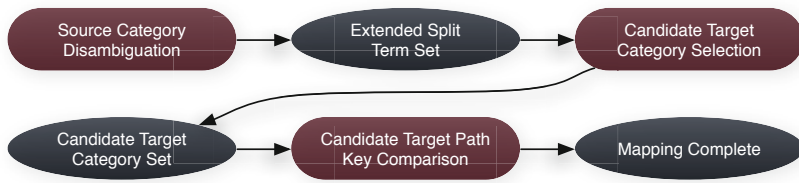


Fig. 1. Framework overview for SCHEMA

3.1 General Assumptions

In product taxonomies, a frequently seen phenomenon is that of composite categories. These are nodes, that combine multiple — usually related — classes into one, like category ‘Movies, Music & Games’ from Amazon. Each of the three parts could have been a separate class as well, as different product concepts are represented. An assumption in the development of SCHEMA was that composite categories need to be treated adequately, as the target taxonomy might not use the same conjunction of classes. To handle the phenomenon, SCHEMA splits categories on ampersands, commas, and the string ‘and’. The resulting set of classes, making up the composite category, is called the *Split Term Set*.

Product taxonomies are tree-structured data schemes, and thus have a root node. However, in product taxonomies, root categories (e.g. ‘Products’ or ‘Shopping’) are meaningless, as they do not provide information about the products falling under. The assumption used for SCHEMA is that, as root nodes are meaningless, they should get automatically mapped in taxonomy matching. Furthermore, roots should be disregarded in all computations, such as in path comparisons. Fig. 2 shows that the root categories in dark blue (the left-hand side categories in black & white printing) are matched by SCHEMA, despite being lexically dissimilar.

Between different product taxonomies, it is apparent that varying degrees of specialisation exist with respect to the product classification. This could mean

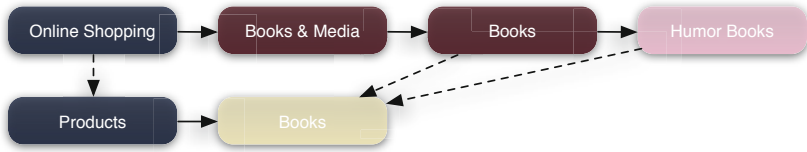


Fig. 2. Mapping example for Overstock (top) to Amazon (bottom) categories. Normal lines indicate a parent-child relationship; dashed lines indicate SCHEMA’s mapping.

that there possibly is no direct match for a very specific source category in the target taxonomy. In such a case, it makes sense to match the source category to a more general target category, as from a hierarchical definition, products from a specific category should also fit into a more general class. Figure 2 shows that category ‘Books’ (Overstock) is mapped to ‘Books’ (Amazon), as one would expect. Unfortunately, there is no direct match for ‘Humor Books’ (Overstock) in Amazon. However, humor books are also a kind of books, so SCHEMA will map this category to the more general ‘Books’ category from Amazon. The more general category is found by following the defined mapping for the parent of the current source category. Note that root mappings are precluded.

SCHEMA’s last assumption is, that as usage of capitals in category names does not affect the meaning, all lexical matching is performed case-insensitive.

3.2 Source Category Disambiguation

The first step in creating a mapping for a category from the source taxonomy, is to disambiguate the meaning of its name. As different taxonomies use varying denominations to identify the same classes, it is required that synonyms of the source category label are taken into account for finding candidate target categories. However, using all synonyms could result in inclusion of synonyms of a faulty sense, which could for example cause a ‘laptop’ to be matched with a book to write notes (i.e., a notebook). To account for this threat, SCHEMA uses a disambiguation procedure in combination with WordNet [17], to find only synonyms of the correct sense for the current source category. This procedure is based on context information in the taxonomy, of which can be expected that it gives some insight into the meaning of the source category name. Concerning the general assumption on composite categories in Sect. 3.1, SCHEMA disambiguates every part of the source category (Split Term Set) separately. The result after disambiguation is called the *Extended Split Term Set*. Note that the target taxonomy does not play a role in the source category disambiguation.

Algorithm 1 explains the procedure that is used to create the Extended Split Term Set for the current source category. First, Algorithm 1 splits the (composed) source category into separate classes: the Split Term Set. The same split is performed for all children, and for the parent of the source category, which will act as ‘context’ for the disambiguation process. Next, the disambiguation procedure itself, which will be discussed shortly, is called for every split part

of the source category. The result, the Extended Split Term Set, contains a set of synonyms of the correct sense for each individual split term. The Extended Split Term Set is used in SCHEMA to find candidate target categories, and to evaluate co-occurrence of nodes for path-comparison.

Algorithm 1. Finding Source Category's Extended Split Term Set

Require: source category to disambiguate: w_{category}
Require: source category's parent: w_{parent} , and set of its children: W_{children}
Require: function $\text{splitComposite}(w)$, which splits composite category name w into a set of individual classes: a split term set W
Require: function $\text{disambiguate}(w_{\text{target}}, W_{\text{context}})$, disambiguates a word using a set of context words, resulting in a set of correct synonyms (described by Algorithm 2)

- 1: {First, all used categories get split on composite classes}
- 2: $W_{\text{category}} \leftarrow \text{splitComposite}(w_{\text{category}})$
- 3: $W_{\text{parent}} \leftarrow \text{splitComposite}(w_{\text{parent}})$
- 4: $W_{\text{child}} \leftarrow \emptyset$
- 5: **for all** $w_{\text{currentChild}} \in W_{\text{children}}$ **do**
- 6: $W_{\text{child}} \leftarrow W_{\text{child}} \cup \text{splitComposite}(w_{\text{currentChild}})$
- 7: **end for**
- 8: $W_{\text{context}} \leftarrow W_{\text{child}} \cup W_{\text{parent}}$
- 9: $\text{extendedSplitTermSet} \leftarrow \emptyset$
- 10: {For every split part of the source category, find the extended term set}
- 11: **for all** $w_{\text{srcSplit}} \in W_{\text{category}}$ **do**
- 12: $\text{extendedTermSet} \leftarrow \text{disambiguate}(w_{\text{srcSplit}}, W_{\text{context}})$
 {Always include original split term, also when WSD is unsuccessful}
- 13: $\text{extendedTermSet} \leftarrow \text{extendedTermSet} \cup \{w_{\text{srcSplit}}\}$
- 14: $\text{extendedSplitTermSet} \leftarrow \text{extendedSplitTermSet} \cup \{\text{extendedTermSet}\}$
- 15: **end for**
- 16: **return** $\text{extendedSplitTermSet}$

As explained before, disambiguation of the source category name is based on a set of words from its context. The idea to use this context is based on a well-known algorithm for word sense disambiguation from Lesk [13]. However, traditional dictionary glosses, used by Lesk, may not provide sufficient vocabulary for successful matching. Therefore Banerjee & Pedersen [2] propose to use the rich semantic relations of WordNet, considering also related glosses of both target and context words to reduce this effect. Unfortunately, this introduces another problem: the computation time increases exponentially with the number of context words. To prevent computation time from exploding, Kilgarriff & Rosenzweig [12] propose to use Lesk's traditional algorithm with heuristics to simplify the search. Instead of using a dictionary gloss for every context word, they propose to use only the context words. This method reduces time complexity, but has similar vocabulary-related restrictions as the original Lesk algorithm. SCHEMA uses the best of these procedures, utilising the rich semantic relations of WordNet for the target word, while comparing only to the plain terms from

the context, as described in Algorithm 2. For every possible sense of the target word, the overlap between its related glosses and the plain context words is assessed. The length of the longest common substring is used as similarity measure, and the sense with the highest accumulated score is picked as winner.

Algorithm 2. Context-Based Target Word Disambiguation

Require: word to disambiguate: w_{target} , and set of context words: W_{context}
Require: function $\text{getSynsets}(w)$, gives all synonym sets (representing one sense in WordNet), of which word w is a member
Require: function $\text{getRelated}(S)$, gives synonym sets directly related to synset S in WordNet, based on hypernymy, hyponymy, meronymy and holonymy. Result includes synset S as well.
Require: function $\text{longestCommonSubstring}(w_a, w_b)$, which computes the length of the longest common sequence of consecutive characters between two strings, corrected for length of the longest string, resulting in an index in the range $[0, 1]$
Require: function $\text{getGloss}(S)$, returns the gloss associated to a synset S in WordNet

```

1:  $Z \leftarrow \text{getSynsets}(w_{\text{target}})$  { $Z$  holds all possible senses}
2:  $\text{bestScore} \leftarrow 0$ 
3:  $\text{bestSynset} \leftarrow \emptyset$ 
   {Evaluate every possible sense (synset)  $S \in Z$  of target word  $w_{\text{target}}$ }
4: for all  $S \in Z$  do
5:    $\text{senseScore} \leftarrow 0$ 
6:    $R \leftarrow \text{getRelated}(S)$ 
   {For every combination of context words & (related) glosses, check similarity}
7:   for all  $(S_{\text{related}}, w_{\text{context}}) \in R \times W_{\text{context}}$  do
8:      $\text{gloss} \leftarrow \text{getGloss}(S_{\text{related}})$ 
9:      $\text{senseScore} \leftarrow \text{senseScore} + \text{longestCommonSubstring}(\text{gloss}, w_{\text{context}})$ 
10:  end for
11:  if  $\text{senseScore} > \text{bestScore}$  then
12:     $\text{bestScore} \leftarrow \text{senseScore}$ 
13:     $\text{bestSynset} \leftarrow S$  {Update best known synset so far}
14:  end if
15: end for
16: return  $\text{bestSynset}$ 

```

3.3 Candidate Target Category Selection

The result of the Source Category Disambiguation, the Extended Split Term Set, is used to find matching categories in the target taxonomy. This set of candidate categories is basically a pre-selection for the decision to which target category the current category can be mapped to. The selection relies on SCHEMA's definition of a category node match, *Semantic Match*, described by Algorithm 3, which is used consistently throughout SCHEMA. It is used to classify a source category and a target category as equivalent or dissimilar, utilising the enriched information provided by the Extended Split Term Set for the source category,

in combination with lexical matching to evaluate similarity between the category names. For the composite categories, SCHEMA assumes that with respect to the split terms, the source category is a subset of the target category. This ensures that all products in a mapped source category fit in the target category.

For every split part of the source category, Semantic Match checks whether there is a matching part in the target category. A match can mean either that the source split part is contained as separate component in a target part, or that they share a lexical similarity based on the normalised Levenshtein index [14], exceeding a chosen threshold. When all split parts of the source category have a match in the target category, the match is considered semantically correct.

Algorithm 3. Semantic Match

Require: extended split term set E , with sets of synonyms S of the correct sense for every split term of the source category

Require: target taxonomy category name: w_{target}

Require: Node Match Threshold t_{node} , defines the minimum degree of lexical similarity in order to classify two class names as equal

Require: function $\text{splitComposite}(w)$, splits composite category name w into a set of individual classes: a split term set W

Require: function $\text{levenshtein}(w_a, w_b)$, computes the edit distance between two strings

Require: function $\text{containsAsSeparateComponent}(w_a, w_b)$, indicates whether string w_a contains string w_b as separate part (middle of another word is not sufficient)

1: $W_{\text{target}} \leftarrow \text{splitComposite}(w_{\text{target}})$

2: $\text{subSetOf} \leftarrow \text{true}$ {Starting assumption: source split term set is subset of target}

3: **for all** $S_{\text{srcSplit}} \in E$ **do**

4: $\text{matchFound} \leftarrow \text{false}$

5: **for all** $(w_{\text{srcSplitSyn}}, w_{\text{targetSplit}}) \in S_{\text{srcSplit}} \times W_{\text{target}}$ **do**

6: $\text{edit_dist} \leftarrow \text{levenshtein}(w_{\text{srcSplitSyn}}, w_{\text{targetSplit}})$
 {Normalise distance based on length and convert to similarity measure}

7: $\text{similarity} \leftarrow 1 - \text{edit_dist} / \max(w_{\text{srcSplitSyn}}, w_{\text{targetSplit}})$

8: **if** $\text{containsAsSeparateComponent}(w_{\text{targetSplit}}, w_{\text{srcSplitSyn}})$ **then**

9: $\text{matchFound} \leftarrow \text{true}$

10: **else if** $\text{similarity} \geq t_{\text{node}}$ **then**

11: $\text{matchFound} \leftarrow \text{true}$

12: **end if**

13: **end for**

14: **if** $\text{matchFound} = \text{false}$ **then**

15: $\text{subSetOf} \leftarrow \text{false}$

16: **end if**

17: **end for**

18: **return** subSetOf

Figure 3 shows some candidates that have been found for category ‘Tubs’ from Overstock. The Source Category Disambiguation procedure discussed in Sect. 3.2 results in the following Extended Split Term Set: $\{\{\text{Tubs}, \text{bathtub}, \text{bathing tub}, \text{bath}, \text{tub}\}\}$. Synonym ‘bath’ is sufficient for candidate category ‘Kitchen

& Bath Fixtures’ (at the top of Fig. 3), to be selected. As ‘bath’ is included in split target part ‘Bath Fixtures’ (as separate word), it matches, according to Algorithm 3, making target category ‘Kitchen & Bath Fixtures’ a superset of source category ‘Tubs’. Hence it is classified as a semantic match, and thus selected as proper candidate target category.

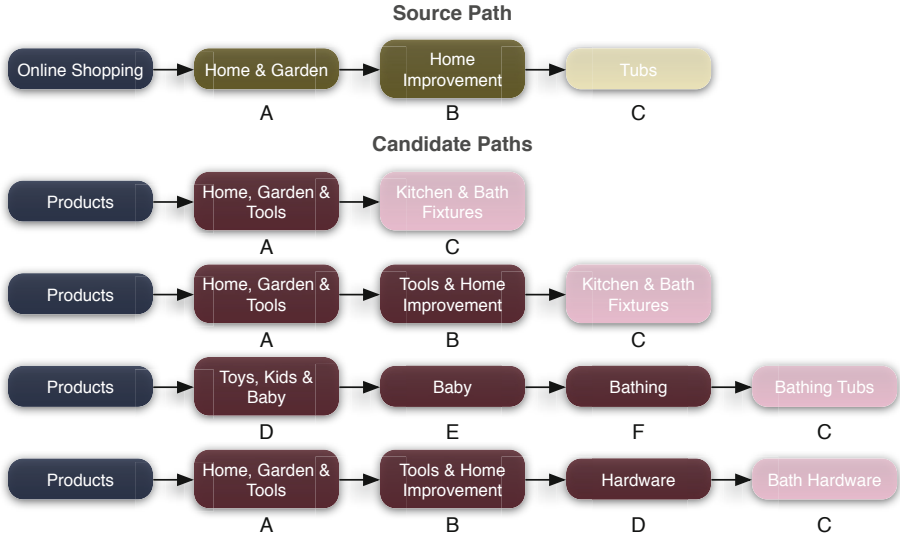


Fig. 3. Source category path for ‘Tubs’ in Overstock, with associated candidate target categories from Amazon

3.4 Candidate Target Path Key Comparison

SCHEMA’s last step is to select the best alternative from the set of found candidate target categories, using a method that scores the similarity of the source category path against a candidate target path. This *Candidate Target Path Key Comparison* is used for every element from the set of candidate target paths. The candidate with the highest score is selected as winner. The idea of the Candidate Path Key Comparison is simple in nature, though powerful in the sense that it assesses similarity based on both structural and lexical relatedness.

For both source and candidate target path, a key is generated for every node (category) in the path. This is done in such a way, that every unique node gets a unique key. Similarly, when two nodes — independent from the path they come from — are seen as identical, they are labelled with the same key. An important question is: when are two nodes seen as identical? A straightforward way would be to base this purely on lexical similarity of the category names. However, SCHEMA uses a richer source of information for nodes from the source path: the Extended Split Term Set. Two nodes from the source path are seen as

identical if and only if their Extended Split Term Sets are the same. A node from the source path and a node from the candidate target path are seen as identical when Algorithm 3, the Semantic Match procedure, decides so. The result is a key list for both the source path and the current candidate target path.

Figure 3 shows the key list for the source and candidate targets paths for category ‘Tubs’. The candidate path at the bottom, is a good example of how Semantic Match classifies nodes as being similar. Candidate node ‘Tools & Home Improvement’ is assigned the same key (‘B’) as source node ‘Home Improvement’, as the first one is a superset of the last one, thus all products under the second should fit into the first. Considering candidate ‘Bath Hardware’ itself, one of the synonyms of source category ‘Tubs’ (‘bath’), is included in the name of the candidate category. Hence, ‘Bath Hardware’ gets the same key (‘C’) as ‘Tubs’.

For the key lists found for source and candidate path, the similarity is assessed using the Damerau-Levenshtein distance [4]. This measure captures the (dis)similarity and transposition of the nodes, hence both the number of co-occurring nodes and the consistency of the node order are taken into account. As the Damerau-Levenshtein distance is used in normalised form, a dissimilar node in a long candidate path is weighted as less bad than the same dissimilar node in a shorter path, which can unfortunately lead to biased results. Therefore, a penalty is added for every unique key assigned solely to the candidate path, or more precise: for every node for which no match exists in the source path. The formula used as similarity measure for the key lists is as follows:

$$candidateScore = 1 - \frac{damLev(K_{src}, K_{candidate}) + p}{\max(K_{src}, K_{candidate}) + p} \quad (1)$$

where K is a key list, p the penalty (# dissimilar nodes in candidate path), $damLev()$ computes the Damerau-Levenshtein distance between two key lists, and $\max()$ computes the maximum length of two key lists.

In Fig. 3, the uppermost and lowermost candidate paths give an example of the penalty’s usefulness. One is too short, the other too long. The shortest (‘Kitchen & Bath Fixtures’) does not contain new nodes in comparison to the source path. With just one edit operation (insertion of key ‘B’), it gets a candidate score of $1 - \frac{1+0}{3+0} = \frac{2}{3}$. The longest contains a new node: ‘Hardware’. This gives the long path a penalty of 1, while the edit distance is also 1 (deletion of key ‘D’), resulting in a score of $1 - \frac{1+1}{4+1} = \frac{3}{5}$. Without penalty it would score $\frac{3}{4}$, causing it to win from the short path, which does not contain unrelated nodes. Clearly, we prefer the first candidate path, because the second candidate path possibly changes the meaning of node ‘C’ as it has as parent a new node ‘D’.

Once the candidate target category with the highest score has been found, it is mapped if the score exceeds the *Final Similarity Threshold* (t_{final}). This threshold prevents the algorithm of performing incorrect mappings, and should not be confused with the Node Match Threshold used in Algorithm 3. When a path does not pass the Final Similarity Threshold, or when no candidate paths have been found, the source category is mapped to the mapping of its parent

(but excluding the root), according to the assumption in Sect. 3.1. The complete framework procedure then repeats for the next source taxonomy category.

4 Evaluation

In order to assess SCHEMA’s performance, it is compared to similar algorithms. We have chosen to compare it with PROMPT [19], being a general-purpose algorithm that is well-known in the field of ontology mapping. Additionally, the algorithm of Park & Kim [20] is included in the comparison, due to their focus on product taxonomy mapping in particular. First, we briefly discuss how the evaluation has been set up. Then, we present the results for each algorithm and discuss their relative performance.

4.1 Evaluation Design

Three product taxonomies from real-life datasets were used for the evaluation. The first dataset contains more than 2,500 categories and is from Amazon (www.amazon.com). The second dataset contains more than 1,000 categories and is from Overstock (www.o.co). Overstock is an online retailer with RDFa-tagged product pages for the GoodRelations [9] ontology. The last dataset contains over 44,000 categories and is from the shopping division in the Open Directory Project (ODP, www.dmoz.org). Using these three datasets, six different combinations of source and target taxonomies can be made. In order to evaluate the algorithms’ performance on the mappings, it is required that each of the mappings is done manually as well. However, as the datasets are too large to manually map every category, we have taken a random sample of five hundred category nodes from each dataset. For every node it is assured that its ancestors are included in the sample as well. The mappings are made from a sampled source taxonomy to a full target taxonomy. Occasionally there are multiple nodes in the reference taxonomy to which a source category node could be correctly mapped. To account for this fact, the manual mapping may define multiple correct mappings for each source category node. The manual mappings were collectively made by three independent individuals, in order to prevent bias.

Each algorithm performed a mapping for every combination of datasets. SCHEMA and the algorithm of Park & Kim carried out multiple mappings, with different parameter values for each combination. Both algorithms use a final score threshold, referred to as t_{final} , ranging from 0 to 1, with increments of 0.05. Furthermore, SCHEMA uses a threshold for node matching, denoted by t_{node} , with range 0.50 to 1 and increments of 0.025. The completed mappings, generated by the algorithms, are compared with the manual mappings, in order to obtain their performance measures. Though ordinary classification and confusion matrix measures apply, the situation is slightly different as there are n ‘positive’ classes (all target categories), and only one negative (null mapping). We therefore define the ‘false positives’ as number of mappings to an incorrect path (either wrong or null), and the ‘false negative’ as incorrect mappings to null. The ‘true’ classes are similar to those in binary classification.

4.2 Results

Table 1 presents a comparison of average precision, recall and F_1 -score for every algorithm. Tables 2, 3, and 4 give a more detailed overview of the results achieved by SCHEMA, the algorithm of Park & Kim, and PROMPT, respectively.

Table 1. Comparison of the best average results for each algorithm

Algorithm	Precision	Recall	F_1 -score	Senses found	WSD accuracy
PROMPT	28.93%	16.69%	20.75%	n/a	n/a
Park & Kim	47.77%	25.19%	32.52%	5.70%	83.72%
SCHEMA	42.21%	80.73%	55.10%	82.03%	84.01%

Table 2. Best results for SCHEMA

Mapping	Precision	Accuracy	Specificity	Recall	F_1 -score	t_{node}	t_{final}
A → ODP	27.27%	40.00%	34.12%	52.50%	35.90%	0.800	0.25
A → O.co	36.34%	49.40%	34.30%	82.69%	50.49%	0.850	0.15
ODP → A	57.49%	68.94%	51.70%	93.66%	71.24%	0.875	0.30
ODP → O.co	39.13%	50.70%	29.59%	95.03%	55.43%	0.850	0.25
O.co → A	53.72%	56.60%	29.13%	84.96%	65.83%	0.850	0.15
O.co → ODP	39.30%	45.80%	27.27%	75.52%	51.69%	0.925	0.30
Average	42.21%	51.91%	38.26%	80.73%	55.10%		

As shown in Table 1, SCHEMA performs better than PROMPT and the algorithm of Park & Kim, on both average recall and F_1 -score. The recall has improved considerably with 221% in comparison to the algorithm from Park & Kim, and 384% against PROMPT. This can be partly attributed to the ability of SCHEMA to cope with lexical variations in category names, using the Levenshtein distance metric, as well as the ability to properly deal with composite categories. Furthermore, SCHEMA maps a category node to its parent’s mapping when no suitable candidate path was found, improving the recall when the reference taxonomy only includes a more general product concept. Achieving a high recall is important in e-commerce applications, as the main objective is to automatically combine the products of heterogeneous product taxonomies in one overview, in order to reduce search failures. A low recall means that many categories would not be aligned, which would mean that many products will be missing from search results. For this reason, it is generally better to map to a more general category rather than not mapping at all. Worthy to mention is the slight decrease in average precision for SCHEMA compared with the algorithm of Park & Kim: 42.21% against 47.77%. This is due to the fact that there is a trade-off between precision and recall: achieving a higher recall means that an

Table 3. Best results for Park & Kim algorithm

Mapping	Precision	Accuracy	Specificity	Recall	F ₁ -score	t_{final}
A → ODP	35.77%	34.00%	57.89%	16.84%	22.90%	0.05
A → O.co	60.16%	47.20%	76.78%	25.61%	35.92%	0.00
ODP → A	37.06%	41.48%	51.94%	30.29%	33.33%	0.00
ODP → O.co	36.76%	35.87%	48.68%	25.09%	29.82%	0.10
O.co → A	61.14%	36.20%	52.11%	29.89%	40.15%	0.00
O.co → ODP	55.71%	36.60%	62.87%	23.42%	32.98%	0.50
Average	47.77%	38.56%	58.38%	25.19%	32.52%	

Table 4. Best results for PROMPT

Mapping	Precision	Accuracy	Specificity	Recall	F ₁ -score
A → ODP	13.55%	25.40%	44.17%	8.08%	10.12%
A → O.co	51.69%	45.40%	74.44%	22.02%	30.89%
ODP → A	20.20%	35.47%	46.44%	19.61%	19.90%
ODP → O.co	20.86%	29.86%	42.64%	16.18%	18.22%
O.co → A	50.00%	32.20%	45.96%	25.66%	33.92%
O.co → ODP	17.27%	25.80%	47.73%	8.57%	11.46%
Average	28.93%	32.36%	50.23%	16.69%	20.75%

algorithm has to map more categories, resulting in possible imprecision when the similarity between categories is low. Both SCHEMA and the algorithm of Park & Kim use configurable final thresholds to filter out weaker matches, but it cannot fully prevent mistakes from occurring. Despite the slightly worse performance on precision, SCHEMA manages to find a more suitable trade-off between precision and recall for product taxonomy mapping than PROMPT and the algorithm of Park & Kim. This is illustrated by the good performance on recall and the higher F₁-score of 55.10%. PROMPT uses a conservative mapping approach, well-suited for general ontology mapping, but unsuitable for e-commerce due to the small portion of mappings. The algorithm of Park & Kim performs better in this regard, especially on precision, but the recall is hampered by the fact that it neglects the existence of composite categories. Furthermore, it uses a rather strict lexical matching procedure between category names, in which a category name has to be a full substring of the other, creating issues when slight lexical variations occur. In addition, the disambiguation procedure from Park & Kim only manages to find a sense in WordNet in 5.70% of the total categories on average. Unfortunately, the rather good accuracy of disambiguation (83.72%) is therefore based on a very small amount of cases, making the number rather untrustworthy. The Lesk-based disambiguation algorithm employed by SCHEMA performs well on both the percentage of senses found and the accuracy, scoring 82.03% and 84.01%, respectively.

5 Conclusions and Future Work

This paper proposes SCHEMA, an algorithm capable of performing automated mapping between heterogeneous product taxonomies in e-commerce. The main objective for developing SCHEMA is facilitating the aggregation of product information from different sources, thus reducing search failures when shopping online. To achieve this objective, SCHEMA utilises word sense disambiguation techniques on category labels, based on the ideas from the algorithm proposed by Lesk [13], in combination with the WordNet semantic lexicon. Furthermore, it deals with domain-specific characteristics, such as composite categories, and lexical variations in category labels. It employs a node matching function, based on inclusiveness of the categories in conjunction with the Levenshtein distance for the class labels, for finding candidate map categories and for assessing the path-similarity. The final mapping quality score is calculated using the Damerau-Levenshtein distance, with an added penalty for dissimilar nodes in the target category's path.

The performance of our algorithm was tested on three real-life datasets and compared with the performance of PROMPT and the algorithm of Park & Kim. This evaluation demonstrates that SCHEMA achieves a considerably higher average recall than the other algorithms, with a relatively small loss of precision. The average F_1 -score resulted in 55.10% for SCHEMA, against 20.75% for PROMPT, and 32.52% for the algorithm of Park & Kim.

As future work, we would like to improve SCHEMA by making use of part-of-speech tagging. As a noun is often more important for concept similarity than an adjective, it makes sense to distinguish between them and treat them accordingly. Another possibility is to combine the hierarchical category structure with product information, as the data fields in product instances could yield extra information for the taxonomy mapping. Additionally, this work could support the creation of an automatic product comparison Web site, capable of autonomously matching products and product taxonomies from different sources.

References

1. Aumueller, D., Do, H.H., Massmann, S., Rahm, E.: Schema and Ontology Matching with COMA++. In: ACM SIGMOD International Conference on Management of Data 2005 (SIGMOD 2005), pp. 906–908. ACM (2005)
2. Banerjee, S., Pedersen, T.: An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In: Gelbukh, A. (ed.) CILCing 2002. LNCS, vol. 2276, pp. 136–145. Springer, Heidelberg (2002)
3. Castano, S., Ferrara, A., Montanelli, S.: H-MATCH: An Algorithm for Dynamically Matching Ontologies in Peer-Based Systems. In: 1st VLDB Int. Workshop on Semantic Web and Databases (SWDB 2003), pp. 231–250 (2003)
4. Damerau, F.J.: A Technique for Computer Detection and Correction of Spelling Errors. *Communications of the ACM* 7(3), 171–176 (1964)
5. Do, H.-H., Melnik, S., Rahm, E.: Comparison of Schema Matching Evaluations. In: Chaudhri, A.B., Jeckle, M., Rahm, E., Unland, R. (eds.) *Web Databases and Web Services 2002*. LNCS, vol. 2593, pp. 221–237. Springer, Heidelberg (2003)

6. Ehrig, M., Sure, Y.: *Ontology Mapping - An Integrated Approach*. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) *ESWS 2004*. LNCS, vol. 3053, pp. 76–91. Springer, Heidelberg (2004)
7. Ehrig, M., Staab, S.: *QOM – Quick Ontology Mapping*. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) *ISWC 2004*. LNCS, vol. 3298, pp. 683–697. Springer, Heidelberg (2004)
8. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: *S-Match: An Algorithm And An Implementation of Semantic Matching*. In: *Dagstuhl Seminar Proceedings of Semantic Interoperability and Integration 2005* (2005)
9. Hepp, M.: *GoodRelations: An Ontology for Describing Products and Services Offers on the Web*. In: Gangemi, A., Euzenat, J. (eds.) *EKAW 2008*. LNCS (LNAI), vol. 5268, pp. 329–346. Springer, Heidelberg (2008)
10. Horrigan, J.B.: *Online Shopping*. Pew Internet & American Life Project Report 36 (2008)
11. Kalfoglou, Y., Schorlemmer, M.: *Ontology Mapping: The State of the Art*. *The Knowledge Engineering Review* 18(1), 1–31 (2003)
12. Kilgarriff, A., Rosenzweig, J.: *Framework and Results for English SENSEVAL*. *Computers and the Humanities* 34(1-2), 15–48 (2000)
13. Lesk, M.: *Automatic Sense Disambiguation using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone*. In: *5th Annual International Conference on Systems Documentation (SIGDOC 1986)*, pp. 24–26. ACM (1986)
14. Levenshtein, V.: *Binary Codes Capable of Correcting Deletions, Insertions, and Reversals* 10(8), 707–710 (1966)
15. Madhavan, J., Bernstein, P.A., Rahm, E.: *Generic Schema Matching with Cupid*. In: *27th International Conference on Very Large Data Bases (VLDB 2001)*. pp. 49–58. Morgan Kaufmann Publishers Inc. (2001)
16. Melnik, S., Garcia-Molina, H., Rahm, E.: *Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching*. In: *18th International Conference on Data Engineering (ICDE 2002)*. pp. 117–128. IEEE (2002)
17. Miller, G.A.: *WordNet: A Lexical Database for English*. *Communications of the ACM* 38(11), 39–41 (1995)
18. Niles, I., Pease, A.: *Towards a Standard Upper Ontology*. In: *International Conference on Formal Ontology in Information Systems 2001 (FOIS 2001)*. ACM (2001)
19. Noy, N.F., Musen, M.A.: *The PROMPT Suite: Interactive Tools for Ontology Merging and Mapping*. *International Journal of Human-Computer Studies* 59(6), 983–1024 (2003)
20. Park, S., Kim, W.: *Ontology Mapping between Heterogeneous Product Taxonomies in an Electronic Commerce Environment*. *International Journal of Electronic Commerce* 12(2), 69–87 (2007)
21. Rahm, E., Bernstein, P.A.: *A Survey of Approaches to Automatic Schema Matching*. *The VLDB Journal* 10(4), 334–350 (2001)
22. 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)
23. VijayaLakshmi, B., GauthamiLatha, A., Srinivas, D.Y., Rajesh, K.: *Perspectives of Semantic Web in E- Commerce*. *International Journal of Computer Applications* 25(10), 52–56 (2011)
24. Yu, Y., Hillman, D., Setio, B., Heflin, J.: *A Case Study in Integrating Multiple E-commerce Standards via Semantic Web Technology*. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) *ISWC 2009*. LNCS, vol. 5823, pp. 909–924. Springer, Heidelberg (2009)
25. Zhang, G.Q., Zhang, G.Q., Yang, Q.F., Cheng, S.Q., Zhou, T.: *Evolution of the Internet and its Cores*. *New Journal of Physics* 10(12), 123027 (2008)

Voting Theory for Concept Detection

Amal Zouaq^{1,2}, Dragan Gasevic^{2,3}, and Marek Hatala³

¹Department of Mathematics and Computer Science, Royal Military College of Canada, CP 17000, Succursale Forces, Kingston ON Canada K7K 7B4

²School of Computing and Information System, Athabasca University, 1 University Drive, Athabasca, AB, Canada T9S 3A3

³School of Interactive Arts and Technology, Simon Fraser University, 250-102nd Avenue, Surrey, BC Canada V3T 0A3

amal.zouaq@rmc.ca, {dgasevic,mhatala}@sfu.ca

Abstract. This paper explores the issue of detecting concepts for ontology learning from text. Using our tool OntoCmaps, we investigate various metrics from graph theory and propose voting schemes based on these metrics. The idea draws its root in social choice theory, and our objective is to mimic consensus in automatic learning methods and increase the confidence in concept extraction through the identification of the best performing metrics, the comparison of these metrics with standard information retrieval metrics (such as TF-IDF) and the evaluation of various voting schemes. Our results show that three graph-based metrics Degree, Reachability and HITS-hub were the most successful in identifying relevant concepts contained in two gold standard ontologies.

Keywords: Concept extraction, voting theory, social choice theory, ontology learning, graph-based metrics.

1 Introduction

Building domain ontologies is one of the pillars of the Semantic Web. However, it is now widely acknowledged within the research community that domain ontologies do not scale well when created manually due to the constantly increasing amount of data and the evolving nature of knowledge. (Semi) Automating the ontology building process (ontology learning) is thus unavoidable for the full-realization of the Semantic Web.

Ontology learning (from texts, xml, etc.) is generally decomposed in a number of steps or layers, which target the different components of an ontology: concepts, taxonomy, conceptual relationships, axioms and axioms schemata [3]. This paper is concerned with the first building block of ontologies which are concepts (classes). In fact, concept extraction is a very active research field, which is of interest to all knowledge engineering disciplines. Generally, research in ontology learning from texts considers that a lexical item (a term) becomes a concept once it reaches a certain value on a given metric (e.g. TFIDF). Numerous metrics such as TF-IDF, C/NC value or entropy [3, 4, 8, 15] have been proposed to identify the most relevant terms from corpora in

information retrieval and ontology learning. For example, some approaches such as Text2Onto [4] and OntoGen [7] rely on metrics such as TFIDF to evaluate term relevance. However, generally the presented solutions either adopt one metric or require that the user identifies the most suitable metric for the task at hand [3]. Following our previous work on graph theory based metrics for concept and relation extraction in ontology learning [19], we propose to enrich this perspective by:

- Testing various metrics from graph theory and
- Taking into account a number of metrics in suggesting suitable concepts based on the Social choice theory [5, 14].

1.1 Motivation

This work aims at exploring the following research questions:

- Do we obtain better results with graph-based metrics rather than with traditional information retrieval measures?

In our previous work [19], we showed that some graph-based metrics are a promising option to identify concepts in an ontology learning system. This paper continues exploring this aspect by enriching the set of studied measures and extending the experiment to another gold standard.

- Do we obtain better results with voting schemes rather than with base metrics?

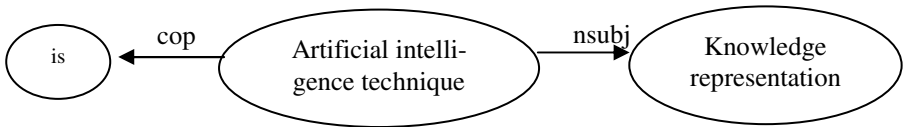
Social Choice Theory studies methods for the aggregation of various opinions in order to reach a consensus [5]. This theory is appealing in our case for two main reasons: firstly, at the practical level, it provides a mean to aggregate the results of various metrics in order to recommend concepts. Secondly, at the theoretical level, it gracefully integrates the idea of consensus, which is one of the main goals of ontologies. In fact, ontologies are meant to improve the communication between computers, between humans and computers and between humans [12]. At this level, another research question is: How can we mimic consensus with automatic ontology learning methods? Although consensus is generally concerned with human users, our hypothesis is that mimicking this characteristic at the level of automatic methods will provide more reliable results.

1.2 Contributions

This paper explores various metrics and voting schemes for the extraction of concepts from texts. Besides bringing a different perspective to this research avenue, the significance of our proposal is that it is applicable to a number of issues related to the Semantic Web, including (but not limited to) learning relationships, helping experts collaboratively build an ontology and reducing the noise that results from the automatic extraction methods.

2 Background

This paper is based on our ontology learning tool, OntoCmaps [19], which in turn is derived from our previous work [20, 21]. OntoCmaps is a “complete” ontology learning system in the sense that it extracts primitive and defined classes (concepts), conceptual relationships (i.e. relations with domain and range), taxonomical relationships (is-a links) and equivalence classes’ axioms (e.g. AI = Artificial Intelligence). OntoCmaps relies on dependency-based patterns to create a forest of multi-digraphs constituted of nodes (terms) and edges (hierarchical and conceptual relations). An example of pattern is:



Semantic Analysis

Is_a (knowledge representation, Artificial Intelligence technique)

By multi-digraphs, we mean that there can be multiple directed relationships from a given term X to a given term Y. For each term X, there can be various relationships to a set of terms S, which constitutes a term map. Some term maps might be isolated, others might be linked to other term maps through relationships, hence creating a forest. Figure 1 shows a term map around the term “intelligent agent”, which can in turn be related to the term of agent, which has itself a term map (and so on). Once the extraction of term maps is performed, the tool filters the results based on various graph-based metrics by assigning several scores to the potential candidates. These scores serve to promote candidate terms as concepts in the ontology. In our previous work [19], we identified a number of graph-based metrics as potential useful measures for extracting terms and relationships. We found promising results by comparing these graph-based metrics (Degree, Betweenness, PageRank and HITS-Authority) to information retrieval metrics such as TF-IDF and TF. We showed that graph-based metrics outperformed these commonly used metrics to identify relevant candidate concepts. We also tested some voting schemes (intersection voting scheme and majority voting scheme) and discovered that they contributed in increasing the precision in our results.

This paper investigates further this previous study, by expanding the set of considered graph-based metrics and by using voting theory methods to consider the vote of each metric for the selection of domain concepts. In fact, voting theory can be used to consider the contribution of each metric and to decrease the noise that results from a NLP pipeline. Voting theory has been experimented in a number of works in artificial intelligence such as agent group-decision-making [6], information mashups [2], ontology merging [14] but to our knowledge, there is no ontology learning tool which proposed to identify concepts through graph-based measures and to increase the confidence of the extractions by aggregating the results of the various metrics through voting theory. This type of aggregation, resulting from the Social Choice Theory [14], seems similar in spirit to ensemble learning methods frequently used in machine learning [13]. However, as previously stated, experimenting voting theories has the

potential to mimic real-world vote aggregation and seems a suitable approach to establishing consensus in learning domain concepts.

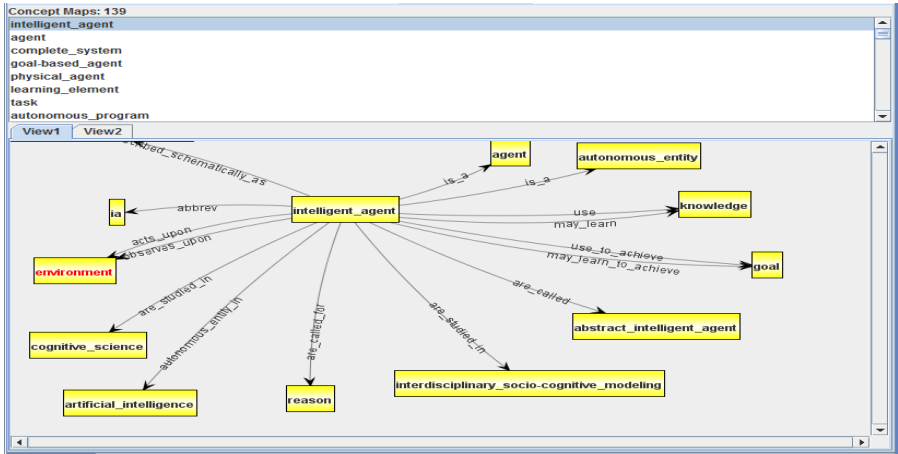


Fig. 1. An Example of OntoCmaps output

3 Voting Theory for Concept Detection

Concept detection through vote aggregation can closely be related to the problem of rank aggregation, which is a well-known problem in the context of Web search where there is a need of finding a consensus between the results of several search engines [5]. Vote aggregation can be defined as the process of reaching a consensus between various rankings of alternatives, given the individual ranking preferences of several voters [2]. In the context of a vote, each metric is considered as a voter.

3.1 Metrics

After the extraction of term maps, OntoCmaps assigns rankings to the extracted terms based on scores from various measures from graph theory (see below). In fact, since OntoCmaps generates a network of terms and relationships, computational network analysis methods are thus applicable and in this case. As outlined by [11], text mining in general and concept extraction in particular can be considered as a process of network traversal and weighting. In this paper, in addition to Degree, PageRank, HITS-Authority and Betweenness presented in [19], we computed three additional metrics HITS-Hubs, Clustering coefficient and Reachability centrality. As explained below, these metrics are generally good indicators of the connectedness and the accessibility of a node, which are two properties that might indicate the importance of a node (here a term) [11, 19].

The following metrics were calculated using the JUNG API [10]:

Degree (Deg) assigns a score to each term based on the number of its outgoing and incoming relationships;

PageRank (Prank) calculates the eigenvector probability of a term with a constant probability of the random walk restarting at a uniform-randomly chosen term [10];

HITS assigns hubs-and-authorities scores to terms based on complementary random walk processes. In our case, we considered that hubs scores and authority scores were two different metrics (Hits-hubs and Hits-Authority);

Betweenness (Bet) calculates a centrality measure where vertices that occur on many shortest paths between other vertices have higher betweenness than those that do not.

Clustering Coefficient (CC) is a measure of the connectedness between the neighbors of a node. It is given by the proportion of links between the terms within a term neighborhood divided by the number of relations that could possibly exist between them.

Reachability centrality (Reach) calculates the distance between each pair of terms using the Dijkstra algorithm.

Each of these metrics produces a ranked list of terms, i.e. a full-ordering of the extracted terms. Full-ordering is considered as the ideal scenario for rank aggregation [5].

3.2 Voting Theory Score-Based Methods

Here, we introduce voting theory methods, which can generally be divided in two main classes: score-based methods and rank-based methods.

In the score-based methods, each metric assigns a score to the elements of a list (here the extracted terms) and the resulting list must take into account the score assigned by each metric. Given the universe of Domain Terms DT , which is composed of all the nominal expressions extracted through our dependency patterns [19], the objective of the vote is to select the most popular terms $t \in DT$, given multiple metrics $m \in M$. Each metric m computes a score Stm for a term t . This score is used to create a fully ordered list TM for each metric.

Sum and maximum values are generally two functions that are used to assign an aggregated score to the terms [18]. We implemented two voting schemes based on scores: the intersection voting scheme and the majority voting scheme.

In the **Intersection Voting Scheme**, we select the terms for which there is a consensus among all the metrics and the score assigned is the sum of the scores of each individual metric normalized by their number.

In the **Majority Voting Scheme**, we select the terms for which there exists a vote from at least 50% of the metrics. The score is again the normalized sum of the score of each individual metric participating in the vote.

Each graph-based metric produced a full list of terms (DT) ordered in the decreasing order of scores. Top-k lists (partial ordering) may be created from full-ordered lists through setting up a threshold over the value of the metrics. In fact, such a threshold might be set to increase metrics' precision: in this case, only the portion of the list whose score is greater than or equal to a threshold is kept for each metric and the voting schemes operate on these partial lists.

3.3 Voting Theory Rank-Based Methods

In rank-based methods, also called positional methods, the elements are sorted based not on their score but on their positions in the lists. Besides the score, we consider a rank r_m , which is the position of a term t within the ordered list produced by a metric m . The total number of ranks in metric m is R_m , which is defined as $\max_t(r_{tm})$, the lowest assigned rank (1 being the best rank). There might be more terms than ranks, because multiple terms might share a rank position.

Following [2], we implemented three positional voting schemes: Borda Count, Nauru and RunOff as these schemes (especially Nauru and Borda) are generally widely accepted in voting theory.

Borda Count Voting Scheme: This method assigns a “rank” r_m to each candidate, with the lowest possible rank assigned to missing entries (usually 0). A candidate who is ranked first receive n points (n =size of the domain terms to be ranked), second $n-1$, third $n-2$ and so on. The “score” of a term for all metrics is equal to the sum of the points obtained by the term in each metric.

Nauru Voting Scheme: The Nauru voting scheme is based on the sum of the inverted rank of each term in each metric ($\sum(1/r_{tm})$). It is used to put more emphasis on higher ranks and to lessen the impact of one bad rank [2].

RunOff Voting Scheme: This voting scheme selects terms one at a time from each metric in a fixed order starting from the highest ranked terms. Once the same term has been selected by at least 50% of the metrics it is added to the voting list and further mentions of it are ignored. This operation is repeated until no remaining terms exist (full-ordering).

4 Methodology

4.1 Dataset

We used a corpus of 30,000 words on the SCORM standard which was extracted from the SCORM manuals [16] and which was used in our previous experiments [19]. This corpus was exploited to generate a gold standard ontology that was validated by a domain expert. To counterbalance the bias that may be introduced by relying on a unique domain expert, we performed user tests to evaluate the correctness of the gold standard. We randomly extracted concepts and their corresponding conceptual and taxonomical relationships from the gold standard and exported them in Excel worksheets. The worksheets were then sent together with the domain corpus and the obtained gold standard ontology to 11 users from Athabasca University, Simon Fraser University, the University of Belgrade, and the University of Lugano. The users were university professors (3), postdoctoral researchers (2), and PhD (5) and master’s (1) students. The users were instructed to evaluate their ontology subset by reading the domain corpus and/or having a look to the global ontology. Each user had a distinct set of items (no duplicated items) composed of 20 concepts and all their conceptual and taxonomical relationships. Almost

29% of the entire gold standard was evaluated by users and overall more than 93% of the concepts were accepted as valid and understandable by these users. This size of the sample and the fact that the sample evaluated by the users was selected randomly can provide us with solid evidence that the results of the user evaluation of the sample can be generalized to the entire gold standard.

To improve its quality, there have been slight modifications to the previous gold standard: class labels were changed by using lemmatization techniques instead of stemming, which introduced some changes in the GS classes. Additionally, some defined classes were also created, and new relationships were discovered due to new patterns added to OntoCmaps. The following table shows the statistics associated to the classes in our current GS¹.

Table 1. GS1 statistics (SCORM)

Primitive classes	Defined Classes	Conceptual Relationships	Taxonomical Relationships
1384	81	895	1121

Once the GS ontology was created, we ran the OntoCmaps tool on the same corpus. The aim was to compare the expert GS concepts with the concepts learned by the tool. We ran our ontology learning tool on the SCORM corpus and generated a ranking of the extractions based on all the above-mentioned metrics: Degree, Betweenness, PageRank, Hits, Clustering Coefficient and Reachability. The tool extracted 2423 terms among which the metrics had to choose the concepts of the ontology.

We also tested our metrics and voting schemes on another smaller corpus (10574 words) on Artificial Intelligence (AI) extracted from Wikipedia pages about the topic. The tool extracted 1508 terms among which the metrics had to choose the concepts of the ontology. Table 2 shows the statistics of the extracted AI gold standard.

Table 2. GS2 statistics (Artificial Intelligence)

Primitive classes	Defined Classes	Conceptual Relationships	Taxonomical Relationships
773	65	287	644

As previously explained, OntoCmaps produced a ranking of terms based on the various metrics introduced in this study, and we divided our results in Top-N lists, gradually increasing the number of considered terms. Recall that metrics order terms from the highest rank to the lowest one. The point was to determine how quickly the accuracy of the results would degrade as we expand the set of considered terms.

Since the SCORM GS contained 1384 primitive classes (concepts), we limited the evaluation to the first 1500 terms in our experiments on SCORM. In the AI GS, we stopped at Top-600 with 773 primitive classes in the GS. We then divided each dataset in small Top-k lists versus large Top-k lists. Small lists are expected to have the higher precision as they include the best rated terms. In the SCORM GS, small lists

¹ <http://azouaq.athabascau.ca/Corpus/SCORM/Corpus.zip>

included Top-k, k=50, 100, 200 (up to ~14.5% of the expected terms) and large lists had k>200. In the AI GS, small lists were Top-50 and Top-100 (up to ~13% of the expected terms).

4.2 Evaluation Criteria

Our experimental evaluation of the different ranking methods tests each of the individual metrics and each of the aforementioned voting systems. There are a number of methods that are used to evaluate similar types of research: in information retrieval and ontology learning, the results are generally evaluated using precision/recall and F-measure [3]. In our case, we chose to concentrate on the precision measure as ontology learning methods obtain difficultly good precision results (see for example [Brewster et al., 2009] and the results of Text2Onto in [4] and in our experiments [19]). Moreover, it is better to offer correct results to the user rather than a more complete but rather a noisier list of concepts [9]. In voting theory and rank aggregation studies [2], the results are often evaluated through Social Welfare Function (SWF). A SWF is a mathematical function that measures the increased social welfare of the voting system. SWF employed in similar research include Precision Optimal Aggregation and the Spearman Footrule distance [2, 17]. Given that Precision Optimal Aggregation is similar in spirit to the precision metric employed in information retrieval, we employed standard precision (Precision Function) against our GS:

Precision = items the metric identified correctly / total number of items generated by the metric

This precision metric was computed for a number of Top-N lists.

For the voting methods, we also calculated a social welfare function (SWF) by computing the proportion of the contribution of each metric to the overall ranking. In our case, the SWF is defined by the number of terms from the gold standard which were included in the promoted concepts of the overall ranking proposed by each voting method.

4.3 Experiments

Quality of Individual Metrics. In [1], the authors indicate that the performance of each individual ranker might have a strong influence over the overall impact of the aggregation. Therefore, we decided first to assess the performance of each metric in various partial lists: Top-50, Top-100, Top-600, Top-1000, Top-1500 and Top-2000. Table 3 show the performance of each metric in each of these lists.

For smaller N-Lists (N=50,100, 200), we can notice that Betweenness, PageRank and Degree are the best performing metrics, while the metrics Reachability, Degree and Hits-Hub become the best ones with larger lists (N=400..2000). Only the degree metrics seems to be constantly present in the best three results of each Top-N list.

Table 3. Precision results for each metric on the SCORM GS

	Bet	Prank	Deg	HITS (Auth)	HITs (Hubs)	Reach	CC
Top-50	96.00	96.00	94.00	86.00	88.00	92.00	70.00
Top-100	96.00	82.00	95.00	77.00	87.00	89.00	75.00
Top-200	88.00	81.00	87.00	79.50	84.50	85.50	78.50
Top-400	77.00	76.00	79.25	73.50	80.25	81.75	73.50
Top-600	75.00	69.67	75.00	71.67	80.50	82.33	69.00
Top-1000	66.30	63.80	71.90	66.10	77.30	77.60	63.40
Top-1500	63.47	61.07	66.67	61.07	71.20	70.07	62.27
Top-2000	61.35	60.90	64.00	60.90	63.95	63.95	61.45

In order to compare our results and make another experiment, we tested our metrics on the second gold standard (AI). The following table shows the results of this experiment. We notice that HITS-Hub and Reachability give the best performance overall.

Table 4. Precision results for each metric on the AI Gold Standard

	Bet	Prank	Deg	HITS (Auth)	HITs (Hub)	Reach	CC
Top-50	78.00	74.00	88.00	74.00	88.00	84.00	84.00
Top-100	71.00	69.00	86.00	62.00	88.00	81.00	67.00
Top-200	70.00	61.50	75.00	57.50	79.50	73.00	56.50
Top-400	60.75	50.75	64.00	57.00	74.50	73.00	54.50
Top-600	56.50	48.83	62.50	53.50	69.67	68.17	54.67
Top-1000	52.80	50.60	57.10	50.60	58.20	58.20	52.30

Choice of Metrics Combinations. Next, we computed the SWF Precision Optimal Aggregation (which in our case is equal to the Precision measure) for each voting method. In order to test various combinations and identify if some metrics were performing better than others, we ran the Weka tool on the SCORM GS, on the AI GS and on the merged set of AI and SCORM GSs (ALL). For each row in the GS, data contained a given term, the scores attributed by each metric and whether or not the term has been included in the GS (Yes/No). Using subset selection in Weka, we tried to identify a subset of features (metrics) which had a significant effect for predicting if a term should belong to the GS or not. Thus, we ran a wrapper-based subset selection algorithm which used a 10 fold cross-validation based on the CfsSubsetEval Attribute Evaluator and a BestFirst search in order to determine the most important attributes.

Table 5. Metrics selection using CfsSubsetEval Attribute Evaluator² and a BestFirst search

Attributes	Number of folds (%) SCORM	Number of folds (%) AI	Number of folds (%) ALL
1 Betw	10(100 %)	1(10 %)	0(0 %)
2 Prank	4(40 %)	10(100 %)	8(80 %)
3 Deg	10(100 %)	10(100 %)	2(20 %)
4 HITS(Auth)	0(0 %)	7(70 %)	10(100 %)
5 HITS(Hubs)	10(100 %)	0(0 %)	1(10 %)
6 Reach	10(100 %)	0(0 %)	10(100 %)
7 CC	8(80 %)	0(0 %)	0(0 %)

Table 5 shows us how many times each metric was selected during a 10-fold cross validation. We can see that some metrics are used more times than others during each cross validation. According to these results, only two metrics Degree and Reachability are present in all 10 folds of our cross-validation (10(100%)) over two datasets: Degree appears over the SCORM and AI datasets while reachability appears over the SCORM and combined (All) datasets. However, we can notice that each individual GS has other significant metrics.

Based on these results, we decided to compute the following voting schemes:

- *Intersection Voting Schemes* (IVS_1, IVS_2 and IVS_3), where IVS_1 is based on all the metrics except the clustering coefficient (which appears to be significant only for SCORM): Hits_Hub, Hits_Authority, PageRank, Degree, Reachability and Betweenness. IVS_2 uses Reachability and Betweenness while IVS_3 is based on Betweenness, Reachability, Hits_Hub and Degree.
- *Majority Voting Schemes* (MVS_1 and MVS_2), where MVS_1 and MVS_2 uses the same metrics respectively as IVS_1 and IVS_3.
- *Borda, Nauru and Runoff* were all based on the metrics Betweenness, Reachability, Degree and HITS-Hubs which are the best metrics for the SCORM GS.

Precision Optimal Aggregation Results on the SCORM and AI GS. In the Top-50 list of the SCORM GS, we noticed that all the voting schemes, except Runoff (96% precision), were successful (100% precision) in identifying relevant concepts among the highest ranked 50 terms. However, as the number of considered terms increases (Table 6), we can notice that the Intersection voting schemes and the majority voting schemes (~82%) beat slightly the other voting scheme systems (Runoff: 77.5%, Nauru: 79.8%, and Borda: 80.5%). In our experiments on the AI GS (Table 6), the best performing voting schemes were:

- Nauru first (90%) and then Runoff, IVS_1 and MVS_2 with 88% in the Top-50 list
- Runoff first (81.5%), Nauru (80%) and then IVS_1 and MVS_2 with 79% in the Top-200 list;
- IVS_2 first (67.5%), Nauru and Runoff with 67%, Borda with 66% and then IVS_1 and MVS_2 with 65.5% in the Top-600 list.

² In Weka, CfsSubsetEval evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them.

Table 6. Performance of voting methods for Top-600 terms on the SCORM GS

	SCORM Top-600	AI Top-600
IVS_1	82.66	65.5
IVS_2	82.33	67.5
IVS_3	82.66	61.83
MVS_1	82.66	61.83
MVS_2	82.66	65.5
Borda	80.5	66
Nauru	79.83	67
RunOff	77.5	67

Comparison with other Metrics on the SCORM Gold Standard. In order to compare our results with some baselines, we computed standard measures used in information retrieval: Term frequency (TF) and TF-IDF as well as random term selection (see table 7). TF and TF-IDF were computed on two sets of terms: TF and TFIDF are computed on all the extracted terms from the corpus while TF(DT) and TFIDF(DT) are computed on domain terms only, i.e. terms that were selected by OntoCmaps as already potential domain terms through patterns and stop words filtering.

Table 7. Traditional Metrics results on the SCORM GS

	TFIDF	TF	TFIDF (DT)	TF (DT)	Random
Top-50	72.00	74.00	92.00	90.00	34.00
Top-100	70.00	66.00	89.00	88.00	33.00
Top-200	66.50	66.00	85.50	79.50	36.00
Top-400	56.75	52.50	72.25	69.50	39.00
Top-600	51.00	47.33	66.83	65.83	40.33
Top-1000	43.70	44.80	62.40	62.70	43.10
Top-1500	43.80	43.07	59.93	59.93	43.74
Top-2000	43.60	43.60	59.80	59.800	43.74

As we can see in table 7, the metrics TFIDF and TF are more successful when they are applied on the pre-filtered domain terms (TFIDF (DT) and TF (DT)). We can also notice that the graph-based metrics and their combination through voting schemes beat the traditional metrics (compare Table 3 and Table 7). Up to the Top-200 list, Betweenness is the best performing metrics, then Reachability (in Top-400, Top-600 and Top-1000), then HITS-Hub (Top-1500), and finally Degree (Top-2000).

Comparison with other Metrics on the AI Gold Standard. We repeated the same experiment on the AI GS. As shown in Table 8, among the traditional metrics, we can also notice that the best performing ones are TFIDF (DT) and TF (DT). If we compare these metrics from Table 8 with the graph-based ones (Table 4), we also see that again graph-based metrics have much better performance in all the Top-k lists (k=50, 100, 200, 400, 600 and 1000). For example, the best in the Top-50 list is the Degree and HITS-Hub with 88% versus (72% for TFIDF (DT)) and in the Top-600, the best is HITS-hub (69.67%) versus 50.83% for TF (DT) and TFIDF (DT).

Table 8. Traditional Metrics results on the AI GS

	TFIDF	TF	TFIDF (DT)	TF (DT)	Random
Top-50	38.00	50.00	72.00	70.00	28.00
Top-100	41.00	47.00	69.00	71.00	30.00
Top-200	40.00	39.00	62.50	61.50	25.50
Top-400	35.00	34.00	56.25	53.25	27.00
Top-600	32.83	31.50	50.83	50.83	28.00
Top-1000	28.00	28.10	56.30	56.40	27.56

Based on the results presented in Tables 3, 4, 7 and 8, we ran a paired sample t-test on each of these metrics combinations and the differences were statistically significant and in favor of graph-based metrics in general, and in favor of Degree, reachability and Hits-hubs in particular.

5 Discussion

In this section, we summarize our findings and the limitations of our work.

5.1 Findings

Our findings are related to our initial research questions:

Do we obtain better results with graph-based metrics rather than with traditional ones?

Obviously, it is possible to confirm this research hypothesis through our experiments with the best performing metrics being:

- SCORM – small lists : Betweenness, PageRank, and Degree
- SCORM- large lists: Hits-Hub, Degree, Reachability
- AI- small lists: Degree, Hits-Hubs, Reachability
- AI- large lists: Hits-Hub, Degree, Reachability

We can observe that Degree is constantly present and that Degree, Hits-Hub and Reachability seem to be the best performing graph-based metrics. This result is confirmed by our machine learning experiments (Table 5) for at least two metrics Degree and Reachability.

Do we obtain better results with voting schemes rather than with base metrics?

As far as voting schemes are concerned, the first question is whether we were able to increase the precision of the results by using these voting schemes (see Table 9). In previous experiments [19], we noticed that some voting schemes were enabling us to get better performance but our ranked lists contained only those terms whose weight was greater than the mean value of the considered metric, which had already a strong impact on the precision of each metric.

Table 9. Comparison between voting schemes and base metrics

	SCORM	AI
Top-50	100% : All voting schemes except Runoff 96%: Bet and PageRank	90%: Nauru 88%: Deg and HITS-hub
Top-100	97%: IVS_3, MVS_1, MVS_2 96%: Bet	86%: Runoff 88%: HITS-hub
Top-200	87%: IVS_1 and MVS_2 88%: Bet	81.5%: Runoff 79.5%: HITS-hub
Top-400	83.75%: IVS_3 and MVS_1 81.75% : Reach	72.75%: Runoff 74.5%: HITS-hub
Top-600	82.67%: IVS_1, IVS_3, MVS_1, MVS_2 82.33%: Reach	67.5: IVS_2 69.67%: HITS-hub
Top-1000	77.7%: IVS_1 and MVS_2 77.6%: Reach	60.7%: IVS_1 and MVS_2 58.2%: HITS-hub and Reach
Top-1500	71.26%: IVS_1 and MVS_2 71.20%: HITS_hub	NA
Top-2000	65.15%: IVS_1 and MVS_2 64%: Degree	NA

Despite a small increase in almost all the cases in favor of voting schemes, the difference between voting schemes and base metrics such as Degree, Hits-Hub and Reachability was not really noteworthy. This asks the question whether such voting schemes are really necessary and whether the identified best graph-based metrics would not be enough, especially if we don't take the mean value as a threshold for the metrics. Having identified that the best base metrics were Degree, Reachability and HITS-hub, we tried some combinations of metrics on the SCORM GS. Despite an improvement of voting theory schemes (e.g. Borda) in some Top-n lists, we did not notice a major difference. Our future work will continue testing combinations of voting schemes and voting theory measures, based on these metrics, on various gold standards. We also plan to compare this voting-based approach with ensemble machine learning algorithms.

5.2 Limitations

One of the most difficult aspects in evaluating this type of work is the necessity to build a gold standard, which in general requires a lot of time and resources. Building a GS that represents a universal ground truth is not possible. Ideally, the experiments presented in this paper should be repeated over various domains to evaluate the generalizability of the approach. However, this is often impossible due to the cost of such a large scale evaluation. In this paper, we extended our previous evaluation on another corpus, and we also extended the set of tested metrics and voting schemes. Future work will have to continue the validation of our approach and to expand the set of “traditional” metrics (such as C/NC value) to be compared with graph-based metrics.

Another limitation is that the metrics that we propose for discovering concepts are graph-based metrics, which involves processing the corpus to obtain a graph while metrics commonly used in information retrieval such as TF-IDF only require the corpus. In our experiments, we always relied on OntoCmaps to generate this graph. However, we do not believe that this could represent a threat to the external validity of our findings, as these metrics are already applied successfully in other areas such social network analysis and information retrieval and are not dependent on anything else than a set of nodes (terms) and edges (relationships).

Finally, despite our focus on concepts in this paper, such a graph-based approach is worth the effort only if the aim is to extract a whole ontology and not only concepts, as it involves discovering terms and relationships between terms. This requirement is also closely linked to another limitation: since we rely on deep NLP to produce such a graph, it requires time to process the corpus and calculate the graph-based metrics. However, we believe that this is not a major limitation, as ontologies are not supposed to be generated on the fly.

6 Conclusion

In this paper, we presented various experiments involving a) the comparison between graph-based metrics and traditional information retrieval metrics and b) the comparison between various voting schemes, including schemes relying on voting theory. Our finding indicates that graph-based metrics always outperform traditional metrics in our experiments. In particular, Degree, Reachability and HITS-Hub seem to be the best performing ones. Although voting schemes increased precision in our experiments, there was only a slight improvement on the precision as compared to the three best performing metrics.

Acknowledgments. This research was funded partially by the NSERC Discovery Grant Program and the Burrough Wellcome Fund.

References

1. Adali, S., Hill, B., Magdon-Ismael, M.: The Impact of Ranker Quality on Rank Aggregation Algorithms: Information vs. Robustness. In: Proc. of 22nd Int. Conf. on Data Engineering Workshops. IEEE (2006)

2. Alba, A., Bhagwan, V., Grace, J., Gruhl, D., Haas, K., Nagarajan, M., Pieper, J., Robson, C., Sahoo, N.: Applications of Voting Theory to Information Mashups. In: IEEE International Conference on Semantic Computing, pp. 10–17 (2008)
3. Cimiano, P.: *Ontology Learning and Population from Text. Algorithms, Evaluation and Applications*. Springer (2006)
4. Cimiano, P., Völker, J.: Text2Onto. In: Montoyo, A., Muñoz, R., Métais, E. (eds.) NLDB 2005. LNCS, vol. 3513, pp. 227–238. Springer, Heidelberg (2005)
5. Dwork, C., Kumar, R., Naor, M., Sivakumar, D.: Rank aggregation methods for the Web. In: Proc. of the 10th International Conference on WWW, pp. 613–622. ACM (2001)
6. Endriss, U.: Computational Social Choice: Prospects and Challenges. *Procedia Computer Science* 7, 68–72 (2011)
7. Fortuna, B., Grobelnik, M., Mladenic, D.: Semi-automatic Data-driven Ontology Construction System. In: Proc. of the 9th Int. Multi-Conference Information Society, pp. 309–318. Springer (2006)
8. Frantzi, K.T., Ananiadou, S.: The C/NC value domain independent method for multi-word term extraction. *Journal of Natural Language Processing* 3(6), 145–180 (1999)
9. Hatala, M., Gašević, D., Siadaty, M., Jovanović, J., Torniai, C.: Can Educators Develop Ontologies Using Ontology Extraction Tools: an End User Study. In: Proc. 4th Euro. Conf. Technology-Enhanced Learning, pp. 140–153 (2009)
10. JUNG, <http://jung.sourceforge.net/> (last retrieved on December 6, 2011)
11. Kozareva, Z., Hovy, E.: Insights from Network Structure for Text Mining. In: Proc. of the 49th Annual Meeting of the ACL Human Language Technologies, Portland (2011)
12. Maedche, A., Staab, S.: Ontology Learning for the Semantic Web. *IEEE Intelligent Systems* 16(2), 72–79 (2001)
13. Polikar, R.: Bootstrap inspired techniques in computational intelligence: ensemble of classifiers, incremental learning, data fusion and missing features. *IEEE Signal Processing Magazine* 24, 59–72 (2007)
14. Porello, D., Endriss, U.: Ontology Merging as Social Choice. In: Proceedings of the 12th International Workshop on Computational Logic in Multi-agent Systems (2011)
15. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. *Information Processing & Management* 24(5), 515–523 (1988)
16. SCORM (2011), <http://www.adlnet.gov> (last retrieved on December 10, 2011)
17. Sculley, D.: Rank Aggregation for Similar Items. In: Proc. of the 7th SIAM International on Data Mining (2007)
18. Shili, L.: Rank aggregation methods. In: *WIREs Comp. Stat.* 2010, vol. 2, pp. 555–570 (2010)
19. Zouaq, A., Gasevic, D., Hatala, M.: Towards Open Ontology Learning and Filtering. *Information Systems* 36(7), 1064–1081 (2011)
20. Zouaq, A., Nkambou, R.: Evaluating the Generation of Domain Ontologies in the Knowledge Puzzle Project. *IEEE Trans. on Kdge and Data Eng.* 21(11), 1559–1572 (2009)
21. Zouaq, A.: *An Ontological Engineering Approach for the Acquisition and Exploitation of Knowledge in Texts*. PhD Thesis, University of Montreal (2008) (in French)

Modelling Structured Domains Using Description Graphs and Logic Programming*

Despoina Magka, Boris Motik, and Ian Horrocks

Department of Computer Science, University of Oxford
Wolfson Building, Parks Road, OX1 3QD, UK

Abstract. Although OWL 2 is widely used to describe complex objects such as chemical molecules, it cannot represent ‘structural’ features of chemical entities (e.g., having a ring). A combination of rules and *description graphs* (DGs) has been proposed as a possible solution, but it still exhibits several drawbacks. In this paper we present a radically different approach that we call Description Graph Logic Programs. Syntactically, our approach combines DGs, rules, and OWL 2 RL axioms, but its semantics is defined via a translation into logic programs under stable model semantics. The result is an expressive OWL 2 RL-compatible formalism that is well suited for modelling objects with complex structure.

1 Introduction

OWL 2 [7] is commonly used to represent objects with complex structure, such as complex assemblies in engineering applications [8], human anatomy [22], or the structure of chemical molecules [10]. In order to ground our discussion, we next present a concrete application of the latter kind; however, the problems and the solution that we identify apply to numerous similar scenarios.

The European Bioinformatics Institute (EBI) has developed the ChEBI ontology—a public dictionary of *molecular entities* used to ensure interoperability of applications supporting tasks such as drug discovery [17]. In order to automate the classification of molecular entities, ChEBI descriptions have been translated into OWL and then classified using state of the art Semantic Web reasoners. While this has uncovered numerous implicit subsumptions between ChEBI classes, the usefulness of the approach was somewhat limited by a fundamental inability of OWL 2 to precisely represent the structure of complex molecular entities. As we discuss in more detail in Section 3, OWL 2 exhibits a so-called *tree-model* property [23], which prevents one from describing non-tree-like relationships using OWL 2 schema axioms. For example, OWL 2 axioms can state that butane molecules have four carbon atoms, but they cannot state that the four atoms in a cyclobutane molecule are arranged in a ring. Please note that this applies to *schema* descriptions only: the structure of a particular cyclobutane molecule can be represented using class and property assertions, but the general definition of *all* cyclobutane molecules—a problem that terminologies such as ChEBI aim to solve—cannot be

* This work was supported by the EU FP7 project SEALS and the EPSRC projects ConDOR, ExODA, and LogMap.

fully described in OWL 2. As we show in Section 3, an ontology may therefore fail to entail certain desired consequences.

A common solution to this problem is to extend OWL 2 with a rule-based formalism such as SWRL [11]. However, adding rules to even a very small fragment of OWL 2 makes the basic reasoning problems undecidable [14], which hinders practical usability. Decidability can be ensured by applying the rules only to the explicitly named individuals [21], or by restricting the shape of the rules [12]. Such restrictions, however, typically prevent the rules from axiomatising the required structures.

In our previous work, we considered a combination of OWL 2, rules, and *description graphs* (DGs) [20]—a graphical notation for describing non-tree-like structures. We ensured decidability of reasoning via a *property separation* condition and by requiring DGs to be *acyclic*. Intuitively, the latter means that DGs can describe structures of arbitrary shape, but bounded in size, while the former limits the interaction between the OWL and DG parts, thus preventing multiple DG structures from merging into one structure of (potentially) unbounded size. As reported in [10], DGs solved only some of the problems related to the representation of structured objects, and our subsequent discussions with EBI researchers have revealed the following drawbacks.

First, the DG approach [20] does not allow one to define structures based on the *absence* of certain characteristics. For example, an inorganic molecule is commonly described as ‘a molecule *not* containing a carbon atom’, which can then be used to classify water as an inorganic molecule. Designing an axiomatisation that produces the desired entailment is very cumbersome with the DG approach: apart from stating that ‘each water molecule consists of one oxygen and two hydrogen atoms’, one must additionally state that ‘these three atoms are the only atoms in a water molecule’ and that ‘neither hydrogen nor oxygen atoms are carbon atoms’. Second, the separation conditions governing the interaction of the OWL 2 and DG components makes the combined language rather difficult to use, as no role can be used in both components. Third, the acyclicity condition from [20] is rather cumbersome: a modeller must add a number of negative class assertions to DGs so as to make any ontology with cyclic implications between DGs unsatisfiable. This solution fails to cleanly separate the semantic consequences of an ontology from the acyclicity check.

In response to this critique, in this paper we present a radically different approach to modelling complex objects via a novel formalism that we call Description Graph Logic Programs (DGLP). At the syntactic level, our approach combines DGs, rules, and OWL 2 RL axioms [19]. In order to overcome the first problem, we give semantics to our formalism via a translation into logic programs (which can be easily done for OWL 2 RL axioms [9]) interpreted under stable model semantics. As we show in Section 4, the resulting formalism can capture conditions based on the absence of information. Moreover, we address the second problem by ensuring decidability without the need for complex property separation conditions.

To address the third problem, in Section 5 we discuss existing syntactic acyclicity conditions, such as weak acyclicity [6] and super-weak acyclicity [16], and argue that they unnecessarily rule out some very simple and intuitively reasonable ontologies. As a remedy, we present a novel *semantic acyclicity* condition. Roughly speaking, a modeller is required to specify a precedence relation describing which DGs are allowed to imply

other DGs; a cyclic ontology that is not compatible with this precedence relation entails a special propositional symbol. A cyclic ontology can still entail useful consequences, but termination of reasoning can no longer be guaranteed.

In Section 6 we consider the problem of reasoning with ontologies including only negation-free rules. We show that the standard bottom-up evaluation of logic programs can decide the relevant reasoning problems for semantically acyclic ontologies, and that it can also decide whether an ontology is semantically acyclic. Furthermore, in Section 7 we show that this result can be extended to ontologies with *stratified* negation.

In Section 8 we present the results of a preliminary evaluation of our formalism. We show that molecule descriptions from the ChEBI ontology can be translated into a DGLP ontology that entails the desired subsumption consequences. Furthermore, despite the very preliminary nature of our implementation, we show that reasoning with DGLP ontologies is practically feasible. Thus, in this paper we lay the theoretical foundations of a novel, expressive, and OWL 2 RL-compatible ontology language that is well suited to modelling objects with complex structure.

The proofs of all technical results presented in this paper are given in a technical report that is available online¹

2 Preliminaries

We assume the reader to be familiar with OWL and description logics. For brevity, we write OWL axioms using the DL notation; please refer to [11] for an overview of the DL syntax and semantics. Let $\Sigma = (\Sigma_C, \Sigma_F, \Sigma_P)$ be a *first-order logic signature*, where Σ_C , Σ_F , and Σ_P are countably infinite sets of constant, function, and predicate symbols, respectively, and where Σ_P contains the 0-ary predicate \perp . The arity of a predicate A is given by $ar(A)$. A vector t_1, \dots, t_n of first-order terms is often abbreviated as \mathbf{t} . An *atom* is a first-order formula of the form $A(\mathbf{t})$, where $A \in \Sigma_P$ and \mathbf{t} is a vector of the terms $t_1, \dots, t_{ar(A)}$. A *rule* r is an implication of the form

$$B_1 \wedge \dots \wedge B_n \wedge \mathbf{not} B_{n+1} \wedge \dots \wedge \mathbf{not} B_m \rightarrow H_1 \wedge \dots \wedge H_\ell \quad (1)$$

where H_1, \dots, H_ℓ are atoms, B_1, \dots, B_m are atoms different from \perp , $m \geq 0$, and $\ell > 0$. Let $head(r) = \{H_i\}_{1 \leq i \leq \ell}$, $body^+(r) = \{B_i\}_{1 \leq i \leq n}$, $body^-(r) = \{B_i\}_{n < i \leq m}$, and $body(r) = body^+(r) \cup body^-(r)$. A rule r is *safe* if every variable that occurs in $head(r)$ also occurs in $body^+(r)$. If $body(r) = \emptyset$ and r is safe, then r is a *fact*. We denote with $head_P(r)$, $body_P^+(r)$, $body_P^-(r)$, and $body_P(r)$ the set of predicates that occur in $head(r)$, $body^+(r)$, $body^-(r)$, and $body(r)$, respectively. A rule r is *function-free* if no function symbols occur in r . A *logic program* P is a set of rules. A logic program P is *negation-free* if, for each rule $r \in P$, we have $body^-(r) = \emptyset$.

Given a logic program P , $HU(P)$ is the set of all terms that can be formed using the constants and the function symbols from P (w.l.o.g. we assume that P contains at least one constant). If no variables occur in an atom (rule), then the atom (rule) is *ground*. Given a logic program P , the set $HB(P)$ is the set of all ground atoms constructed using the terms in $HU(P)$ and the predicates occurring in P . The grounding of a rule r

¹ <http://www.cs.ox.ac.uk/isg/people/despoina.magka/>

w.r.t. a set of terms T is the set of rules obtained by substituting the variables of r by the terms of T in all possible ways. Given a logic program P , the program $ground(P)$ is obtained from P by replacing each rule $r \in P$ with its grounding w.r.t. $HU(P)$.

Let $I \subseteq HB(P)$ be a set of ground atoms. Then, I satisfies a ground rule r if $body^+(r) \subseteq I$ and $body^-(r) \cap I = \emptyset$ imply $head(r) \subseteq I$. Furthermore, I is a model of a (not necessarily ground) program P , written $I \models P$, if $\perp \notin I$ and I satisfies each rule $r \in ground(P)$. Given a negation-free program P , set I is a *minimal model* of P if $I \models P$ and no $I' \subsetneq I$ exists such that $I' \models P$. The Gelfond-Lifschitz reduct P^I of a logic program P w.r.t I is obtained from $ground(P)$ by removing each rule r such that $body^-(r) \cap I \neq \emptyset$, and removing all atoms **not** B_i in all the remaining rules. A set I is a *stable model* of P if I is a minimal model of P^I . Given a fact A , we write $P \models A$ if $A \in I$ for each stable model I of P ; otherwise, we write $P \not\models A$.

A substitution is a partial mapping of variables to ground terms. The result of applying a substitution θ to a term, atom, or a set of atoms M is written as $M\theta$ and is defined as usual. Let P be a logic program in which no predicate occurring in the head of a rule in P also occurs negated in the body of a (possibly different) rule in P . Operator T_P applicable to a set of facts X is defined as follows:

$$T_P(X) = X \cup \{h\theta \mid h \in head(r), r \in P, \theta \text{ maps the variables of } r \text{ to } HU(P \cup X) \text{ such that } body^+(r)\theta \subseteq X \text{ and } body^-(r)\theta \cap X = \emptyset\}$$

Let $T_P^0 = \emptyset$, let $T_P^i = T_P(T_P^{i-1})$ for $i \geq 1$, and let $T_P^\infty = \bigcup_{i=1}^\infty T_P^i$. Clearly, $T_P^i \subseteq T_P^{i+1}$ for each $i \geq 0$. Furthermore, such P has at most one stable model [3], and T_P^∞ is the stable model of P if and only if $\perp \notin T_P^\infty$.

3 Motivating Application

We next motivate our work using examples from the chemical Semantic Web application mentioned in the introduction. The goal of this application is to automatically classify chemical entities based on descriptions of their properties and structure. Unfortunately, as discussed in [20], OWL cannot describe cyclic structures with sufficient precision. This causes problems when modelling chemical compounds since molecules often have cyclic parts. For example, the cyclobutane molecule contains four carbon atoms connected in a ring² as shown in Figure 1(a). One might try to represent this structure using the following OWL axiom:

$$\text{Cyclobutane} \sqsubseteq \text{Molecule} \quad \sqcap = 4 \text{ hasAtom.}[\text{Carbon} \sqcap (= 2 \text{ bond. Carbon})]$$

This axiom is satisfied in first-order interpretations I and I' shown in Figures 1(b) and 1(c), respectively; however, only interpretation I correctly reflects the structure of cyclobutane. Furthermore, interpretation I' cannot be ruled out by writing additional axioms: OWL has a variant of the *tree-model property*, so each satisfiable TBox has at least one tree-shaped interpretation. This can prevent the entailment of certain desired

² In fact, cyclobutane also contains two hydrogens attached to each carbon; however, hydrogen atoms are commonly left implicit to simplify the presentation.

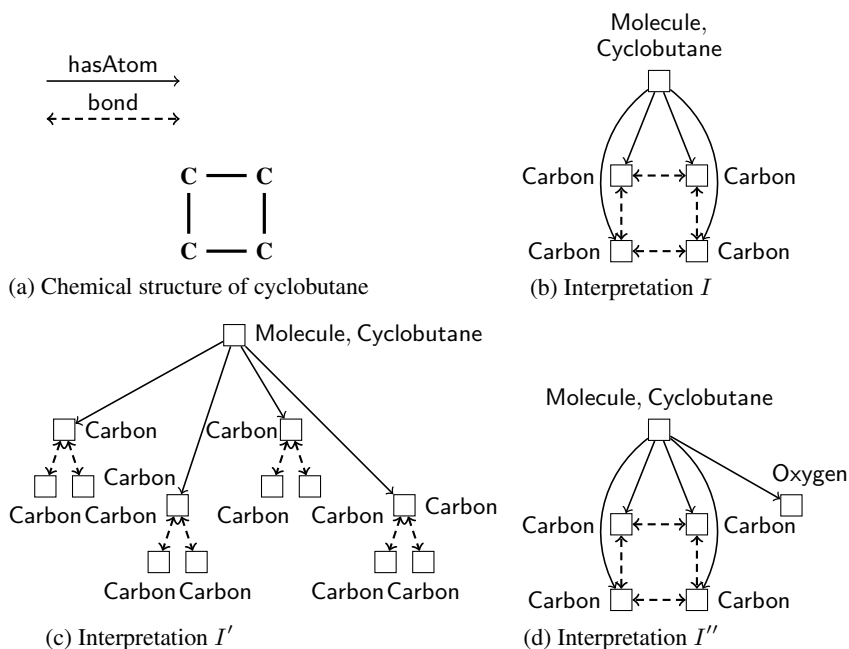


Fig. 1. The chemical structure and the models of cyclobutane

consequences. For example, one cannot define the class of molecules containing four-membered rings that will be correctly identified as a superclass of cyclobutane.

The formalism from [20] addresses this problem by augmenting an OWL ontology with a set of rules and a set of *description graphs* (DGs), where each DG describes a complex object by means of a directed labeled graph. To avoid misunderstandings, we refer to the formalism from [20] as DGDL (Description Graph Description Logics), and to the formalism presented in this paper as DGLP (Description Graph Logic Programs). Thus, cyclobutane can be described using the DG shown in Figure 2(a). The first-order semantics of DGDL ontologies ensures that all models of an ontology correctly represent the DG structure; for example, interpretation I' from Figure 1(c) does not satisfy the DG in Figure 2(a). Nevertheless, the first-order models of DGDL ontologies can still be insufficiently precise. For example, the interpretation I'' shown in Figure 1(d) satisfies the definition of cyclobutane under the semantics of DGDL ontologies. We next show how the presence of models with excess information can restrict entailments³

One might describe the class of hydrocarbon molecules (i.e., molecules consisting exclusively of hydrogens and carbons) using axiom (2). One would expect the definition of cyclobutane (as given in a DGDL ontology) and (2) to imply subsumption (3).

³ Krötzsch et al. [13] also suggest an extension of OWL 2 for the representation of graph-like structures. As we show next, the first-order semantics of this formalism exhibits the same problems as that of DGDL ontologies.

$$\text{Molecule} \sqcap \forall \text{hasAtom.}(\text{Carbon} \sqcup \text{Hydrogen}) \sqsubseteq \text{Hydrocarbon} \quad (2)$$

$$\text{Cyclobutane} \sqsubseteq \text{Hydrocarbon} \quad (3)$$

This, however, is not the case, since interpretation I'' does not satisfy axiom (3). One might preclude the existence of extra atoms by adding cardinality restrictions requiring each cyclobutane to have exactly four atoms. Even so, axiom (3) would not be entailed because of a model similar to I , but where one carbon atom is also an oxygen atom. One could eliminate such models by introducing disjointness axioms for all chemical elements. Such gradual circumscription of models, however, is not an adequate solution, as one can always think of additional information that needs to be ruled out [18].

In order to address such problems, we present a novel expressive formalism that we call Description Graph Logic Programs (DGLP). DGLP ontologies are similar to DGD L ontologies in that they extend OWL ontologies with DGs and rules. In our case, however, the ontology is restricted to OWL 2 RL so that the ontology can be translated into rules [9]. We give semantics to our formalism by translating DGLP ontologies into logic programs with function symbols. As is common in logic programming, the translation is interpreted under *stable* models. Consequently, interpretations such as I'' are not stable models of the DG in Figure 2(a), and hence subsumption (3) is entailed.

Logic programs with function symbols can axiomatise infinite non-tree-like structures, so reasoning with DGLP ontologies is trivially undecidable [4]. Our goal, however, is not to model arbitrarily large structures, but to describe complex objects up to a certain level of granularity. For example, acetic acid has a carboxyl part, and carboxyl has a hydroxyl part, but hydroxyl does not have an acetic acid part (see Fig. 3(a)).

In Section 5 we exploit this intuition and present a condition that ensures decidability. In particular, we require the modeller to specify an ordering on DGs that, intuitively, describes which DGs are allowed to imply existence of other DGs. Using a suitable test, one can then check whether implications between DGs are acyclic and hence whether DGs describe structures of bounded size only. The resulting *semantic* acyclicity condition allows for the modelling of naturally-arising molecular structures, such as acetic acid, that would be ruled out by existing syntax-based acyclicity conditions [6,16].

4 Description Graph Logic Programs

We now present the DGLP formalism in detail. We start by defining a slightly modified notion of description graphs; compared to the definition from [20], the new definition allows for only a single *start predicate* instead of a set of *main concepts*, and it includes a *graph mode* that allows for the automatic generation of ‘recognition’ rules—something that had to be introduced ‘by hand’ in [20].

Definition 1 (Description Graph). A description graph $G = (V, E, \lambda, A, m)$ is a directed labeled graph where

- $V = \{1, \dots, n\}$ is a nonempty set of vertices,
- $E \subseteq V \times V$ is a set of edges,

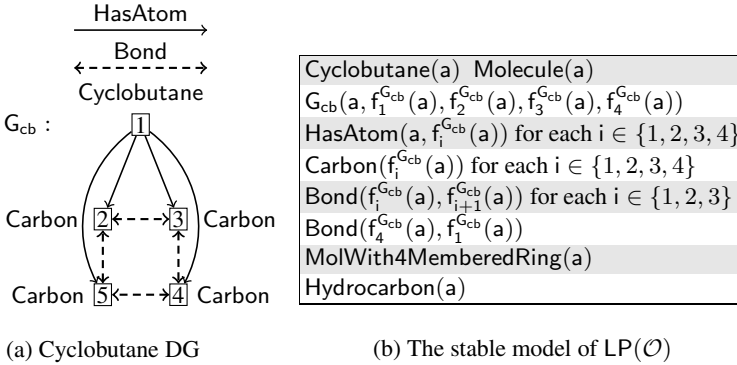


Fig. 2. Representing cyclobutane with DGLP

- λ assigns a set of unary predicates $\lambda(v) \subseteq \Sigma_P$ to each vertex $v \in V$ and a set of binary predicates $\lambda(v_1, v_2) \subseteq \Sigma_P$ to each edge $(v_1, v_2) \in E$,
- $A \in \Sigma_P$ is a start predicate for G such that $A \in \lambda(1)$, and
- $m \in \{\Rightarrow, \Leftarrow, \Leftrightarrow\}$ is a mode for G .

A description graph (DG) abstracts the structure of a complex object by means of a directed labeled graph. For example, Figure 2 illustrates a DG that represents the structure of a cyclobutane molecule. The start predicate of the graph (Cyclobutane in this case) corresponds to the name of the object that the graph describes. The mode determines whether a graph should be interpreted as an ‘only if’, ‘if’, or ‘if and only if’ statement. More precisely, \Rightarrow means that each instance of the DG’s start predicate implies the existence of a corresponding instantiation of the entire graph structure; \Leftarrow means that an instantiation of a suitable graph structure is ‘recognised’ as an instance of the corresponding DG; and \Leftrightarrow means both of the above.

Next we define *graph orderings*, which will play an important role in ensuring the decidability of DGLP.

Definition 2 (Graph Ordering). A graph ordering on a set of description graphs DG is a transitive and irreflexive relation $\prec \subseteq DG \times DG$.

Intuitively, a graph ordering specifies which DGs can imply the existence of instances of other DGs. For example, let G_{AA} be a graph that represents the structure of the acetic acid, and let G_{Cxl} be a graph that represents the structure of the carboxyl group that is a substructure of acetic acid (see Figure 3); then, one might define \prec such that $G_{AA} \prec G_{Cxl}$, so that an acetic acid instance may imply the existence of a carboxyl group instance, but not vice versa. We are now ready to define DGLP ontologies.

Definition 3 (DGLP Ontology). A DGLP ontology $\mathcal{O} = \langle DG, \prec, R, F \rangle$ is a quadruple where DG is a finite set of description graphs, \prec is a graph ordering on DG , R is a finite set of function-free and safe rules, and F is a finite set of function-free facts.

For the sake of simplicity, we do not explicitly include an OWL 2 RL TBox into the definition of DGLP ontologies: OWL 2 RL axioms can be translated into rules as shown

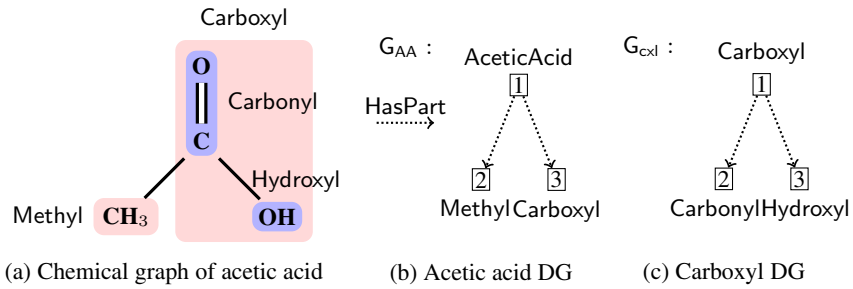


Fig. 3. The chemical graph of acetic acid and the G_{AA} and the G_{cx1} DGs

in [9] and included in R , and datatypes can be handled as in [15]. Similarly, we could think of F as an OWL 2 ABox, as ABox assertions correspond directly to facts [9]. An example of a DGLP ontology is $\{\{G_{AA}, G_{cx1}\}, \{(G_{AA}, G_{cx1})\}, \emptyset, \{\text{AceticAcid}(a)\}\}$.

We next define the semantics of DGLP via a translation into logic programs. Since R and F are already sets of rules and \prec serves only to check acyclicity, we only need to specify how to translate DGs into rules.

Definition 4 (Start, Layout, and Recognition Rule). Let $G = (V, E, A, \lambda, m)$ be a description graph and let $f_1^G, \dots, f_{|V|-1}^G$ be fresh distinct function symbols uniquely associated with G . The start rule s_G , the layout rule ℓ_G , and the recognition rule r_G of G are defined as follows:

$$A(x) \rightarrow G(x, f_1^G(x), \dots, f_{|V|-1}^G(x)) \quad (s_G)$$

$$G(x_1, \dots, x_{|V|}) \rightarrow \bigwedge_{i \in V, B \in \lambda(i)} B(x_i) \wedge \bigwedge_{\langle i, j \rangle \in E, R \in \lambda(i, j)} R(x_i, x_j) \quad (\ell_G)$$

$$\bigwedge_{i \in V, B \in \lambda(i), B \neq A} B(x_i) \wedge \bigwedge_{\langle i, j \rangle \in E, R \in \lambda(i, j)} R(x_i, x_j) \rightarrow G(x_1, \dots, x_{|V|}) \quad (r_G)$$

The start and layout rules of a description graph serve to unfold the graph's structure. The function terms $f_1^G(x), \dots, f_{|V|-1}^G(x)$ correspond to existential restrictions whose existentially quantified variables have been skolemised.

Example 1. The DG of cyclobutane from Figure 2 can be naturally represented by the existential restriction (4). The skolemised version of (4) is the start rule ($s_{G_{cb}}$).

$$\text{Cyclobutane}(x) \rightarrow \exists y_1, y_2, y_3, y_4 : G_{cb}(x, y_1, y_2, y_3, y_4) \quad (4)$$

$$\text{Cyclobutane}(x) \rightarrow G_{cb}(x, f_1^{G_{cb}}(x), f_2^{G_{cb}}(x), f_3^{G_{cb}}(x), f_4^{G_{cb}}(x)) \quad (s_{G_{cb}})$$

The layout rule ($\ell_{G_{cb}}$) encodes the edges and the labelling of the description graph. Finally, the rule $r_{G_{cb}}$ is responsible for identifying the cyclobutane structure:

⁴ In the rest of the paper for simplicity we assume that bonds are unidirectional.

$$\begin{aligned}
G_{\text{cb}}(x_1, x_2, x_3, x_4, x_5) \rightarrow & \text{Cyclobutane}(x_1) \wedge \bigwedge_{2 \leq i \leq 4} \text{Bond}(x_i, x_{i+1}) \wedge \text{Bond}(x_5, x_2) \wedge \\
& \bigwedge_{2 \leq i \leq 5} \text{HasAtom}(x_1, x_i) \wedge \bigwedge_{2 \leq i \leq 5} \text{Carbon}(x_i) \quad (\ell_{G_{\text{cb}}}) \\
\bigwedge_{2 \leq i \leq 5} \text{HasAtom}(x_1, x_i) \wedge & \bigwedge_{2 \leq i \leq 5} \text{Carbon}(x_i) \wedge \bigwedge_{2 \leq i \leq 4} \text{Bond}(x_i, x_{i+1}) \wedge \text{Bond}(x_5, x_2) \rightarrow \\
G_{\text{cb}}(x_1, x_2, x_3, x_4, x_5) & \quad (r_{G_{\text{cb}}})
\end{aligned}$$

Next, we define $\text{Axioms}(DG)$, which is a logic program that encodes a set of DGs.

Definition 5 ($\text{Axioms}(DG)$). For a description graph $G = (V, E, \lambda, A, m)$, the program $\text{Axioms}(G)$ is the set of rules that contains the start rule s_G and the layout rule ℓ_G if $m \in \{\Rightarrow, \Leftrightarrow\}$, and the recognition rule r_G if $m \in \{\Leftarrow, \Leftrightarrow\}$. For a set of description graphs $DG = \{G_i\}_{1 \leq i \leq n}$, let $\text{Axioms}(DG) = \bigcup_{G_i \in DG} \text{Axioms}(G_i)$.

For each DGLP ontology $\mathcal{O} = \langle DG, \prec, R, F \rangle$, we denote with $\text{LP}(\mathcal{O})$ the program $\text{Axioms}(DG) \cup R \cup F$. To check whether a class C is subsumed by a class D , we can proceed as in standard OWL reasoning: we assert $C(a)$ for a a fresh individual, and we check whether $D(a)$ is entailed.

Definition 6 (**Subsumption**). Let \mathcal{O} be a DGLP ontology, let C and D be unary predicates occurring in \mathcal{O} , and let a be a fresh individual not occurring in \mathcal{O} . Then, D subsumes C w.r.t. \mathcal{O} , written $\mathcal{O} \models C \sqsubseteq D$, if $\text{LP}(\mathcal{O}) \cup \{C(a)\} \models D(a)$ holds.

Example 2. We now show how a DGLP ontology can be used to obtain the inferences described in Section 3. Rule (r1) encodes the class of four-membered ring molecules:

$$\begin{aligned}
\text{Molecule}(x) \wedge \bigwedge_{1 \leq i \leq 4} \text{HasAtom}(x, y_i) \wedge \bigwedge_{1 \leq i \leq 3} \text{Bond}(y_i, y_{i+1}) \wedge \text{Bond}(y_4, y_1) \\
\bigwedge_{1 \leq i < j \leq 4} \text{not } y_i = y_j \rightarrow \text{MolWith4MemberedRing}(x) \quad (r_1)
\end{aligned}$$

The use of the equality predicate $=$ in the body of r_1 does not require an extension to our syntax: if $=$ occurs only in the body and not in the head of the rules, then negation of equality can be implemented using a built-in predicate. In addition, we represent the class of hydrocarbons with rules (r2) and (r3).

$$\text{Molecule}(x) \wedge \text{HasAtom}(x, y) \wedge \text{not Carbon}(y) \wedge \text{not Hydrogen}(y) \rightarrow \text{NHC}(x) \quad (r_2)$$

$$\text{Molecule}(x) \wedge \text{not NHC}(x) \rightarrow \text{HydroCarbon}(x) \quad (r_3)$$

$$\text{Cyclobutane}(x) \rightarrow \text{Molecule}(x) \quad (r_4)$$

Finally, we state that cyclobutane is a molecule using (r4) that corresponds to the OWL 2 RL axiom $\text{Cyclobutane} \sqsubseteq \text{Molecule}$. Let $DG = \{G_{\text{cb}}\}$, let $\prec = \emptyset$, let $R = \{r_i\}_{i=1}^4$, let $F = \{\text{Cyclobutane}(a)\}$, and let $\mathcal{O} = \langle DG, \prec, R, F \rangle$. Figure 2(b) shows the only stable model of $\text{LP}(\mathcal{O})$ by inspection of which we see that $\text{LP}(\mathcal{O}) \models \text{HydroCarbon}(a)$ and $\text{LP}(\mathcal{O}) \models \text{MolWith4MemberedRing}(a)$, as expected.

5 Semantic Acyclicity

Deciding whether a logic program with function symbols entails a given fact is known to be undecidable in general [4]. This problem is closely related to the problem of reasoning with datalog programs with existentially quantified rule heads (known as tuple-generating dependencies or tgds) [5]. For such programs, conditions such as weak acyclicity [6] or super-weak acyclicity [16] ensure the termination of bottom-up reasoning algorithms. Roughly speaking, these conditions examine the syntactic structure of the program's rules and check whether values created by a rule's head can be propagated so as to eventually satisfy the premise of the same rule. Due to the similarity between tgds and our formalism, such conditions can also be applied to DGLP ontologies. These conditions, however, may overestimate the propagation of values introduced by existential quantification and thus rule out unproblematical programs that generate only finite structures. As we show in Example 4, this turns out to be the case for programs that naturally arise from DGLP representations of molecular structures.

To mitigate this problem, we propose a new *semantic* acyclicity condition. The idea is to detect repetitive construction of DG instances by checking the entailment of a special propositional symbol *Cycle*. To avoid introducing an algorithm-specific procedural definition, our notion is declarative. The graph ordering \prec of a DGLP ontology \mathcal{O} is used to extend $\text{LP}(\mathcal{O})$ with rules that derive *Cycle* whenever an instance of a DG G_1 implies existence of an instance of a DG G_2 but $G_1 \not\prec G_2$.

Definition 7 (*Check*(\mathcal{O})). Let $G_i = (V_i, E_i, \lambda_i, A_i, m_i)$, $i \in \{1, 2\}$ be two description graphs. We define $\text{ChkPair}(G_1, G_2)$ and $\text{ChkSelf}(G_i)$ as follows:

$$\text{ChkPair}(G_1, G_2) = \{G_1(x_1, \dots, x_{|V_1|}) \wedge A_2(x_k) \rightarrow \text{Cycle} \mid 1 \leq k \leq |V_1|\} \quad (5)$$

$$\text{ChkSelf}(G_i) = \{G_i(x_1, \dots, x_{|V_i|}) \wedge A_i(x_k) \rightarrow \text{Cycle} \mid 1 < k \leq |V_i|\} \quad (6)$$

Let $DG = \{G_i\}_{1 \leq i \leq n}$ be a set of description graphs and let \prec be a graph ordering on DG . We define $\text{Check}(DG, \prec)$ as follows:

$$\text{Check}(DG, \prec) = \bigcup_{i, j \in \{1, \dots, n\}, i \neq j, G_i \not\prec G_j} \text{ChkPair}(G_i, G_j) \cup \bigcup_{1 \leq i \leq n} \text{ChkSelf}(G_i)$$

For a DGLP ontology $\mathcal{O} = \langle DG, \prec, R, F \rangle$, we define $\text{Check}(\mathcal{O}) = \text{Check}(DG, \prec)$.

Example 3. Figure 3(a) shows the structure of acetic acid molecules and the parts they consist of. In this example, however, we focus on the description graphs for acetic acid (G_{AA}) and carboxyl (G_{cxl}), which are shown in Figures 3(b) and 3(c), respectively. Since an instance of acetic acid implies the existence of an instance of a carboxyl, but not vice versa, we define our ordering as $G_{AA} \prec G_{cxl}$. Thus, for $DG = \{G_{AA}, G_{cxl}\}$ and $\prec = \{(G_{AA}, G_{cxl})\}$, set $\text{Check}(DG, \prec)$ contains the following rules:

$$\begin{array}{ll} G_{cxl}(x_1, x_2, x_3) \wedge \text{AceticAcid}(x_i) \rightarrow \text{Cycle} & \text{for } 1 \leq i \leq 3 \\ G_{AA}(x_1, x_2, x_3) \wedge \text{AceticAcid}(x_i) \rightarrow \text{Cycle} & \text{for } 2 \leq i \leq 3 \\ G_{cxl}(x_1, x_2, x_3) \wedge \text{Carboxyl}(x_i) \rightarrow \text{Cycle} & \text{for } 2 \leq i \leq 3 \end{array}$$

We next define when a DGLP ontology is semantically acyclic. Intuitively, this condition will ensure that the evaluation of $\text{LP}(\mathcal{O})$ does not generate a chain of description graph instances violating the DG ordering.

Definition 8. A DGLP ontology \mathcal{O} is said to be semantically acyclic if and only if $\text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O}) \not\models \text{Cycle}$.

Example 4. Let $DG = \{G_{AA}, G_{cxl}\}$ with $m_{AA} = m_{cxl} = \Leftrightarrow$, let $\prec = \{(G_{AA}, G_{cxl})\}$, let $F = \{\text{AceticAcid}(a)\}$, and let $\mathcal{O} = \langle DG, \prec, \emptyset, F \rangle$. By Definition 5, logic program $\text{LP}(\mathcal{O})$ contains F and the following rules (HP abbreviates HasPart):

$$\begin{aligned} \text{AceticAcid}(x) &\rightarrow G_{AA}(x, f_1(x), f_2(x)) \\ G_{AA}(x, y, z) &\rightarrow \text{AceticAcid}(x) \wedge \text{Methyl}(y) \wedge \text{Carboxyl}(z) \wedge \text{HP}(x, y) \wedge \text{HP}(x, z) \\ \text{Methyl}(y) \wedge \text{Carboxyl}(z) \wedge \text{HP}(x, y) \wedge \text{HP}(x, z) &\rightarrow G_{AA}(x, y, z) \\ \text{Carboxyl}(x) &\rightarrow G_{cxl}(x, g_1(x), g_2(x)) \\ G_{cxl}(x, y, z) &\rightarrow \text{Carboxyl}(x) \wedge \text{Carbonyl}(y) \wedge \text{Hydroxyl}(z) \wedge \text{HP}(x, y) \wedge \text{HP}(x, z) \\ \text{Carbonyl}(y) \wedge \text{Hydroxyl}(z) \wedge \text{HP}(x, y) \wedge \text{HP}(x, z) &\rightarrow G_{cxl}(x, y, z) \end{aligned}$$

Let also $\text{Check}(\mathcal{O}) = \text{Check}(DG, \prec)$ as defined in Example 3. The stable model of $P = \text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$ can be computed using the T_P operator:

$$\begin{aligned} T_P^\infty = \{ &\text{AceticAcid}(a), G_{AA}(a, f_1(a), f_2(a)), \text{HP}(a, f_1(a)), \text{HP}(a, f_2(a)), \\ &\text{Methyl}(f_1(a)), \text{Carboxyl}(f_2(a)), G_{cxl}(f_2(a), g_1(f_2(a)), g_2(f_2(a))), \text{Carbonyl}(g_1(f_2(a))), \\ &\text{Hydroxyl}(g_2(f_2(a))), \text{HP}(f_2(a), g_1(f_2(a))), \text{HP}(f_2(a), g_2(f_2(a)))\} \end{aligned}$$

Since Cycle is not in the (only) stable model of P , we have $P \not\models \text{Cycle}$ and \mathcal{O} is semantically acyclic. However, P is neither weakly [6] nor super-weakly acyclic [16]. This, we believe, justifies the importance of semantic acyclicity for our applications.

Example 5 shows how functions may trigger infinite generation of DG instances.

Example 5. Let $\mathcal{O} = \langle \{G\}, \emptyset, \{B(x) \rightarrow A(x)\}, \{A(a)\} \rangle$ be a DGLP ontology where G is such that $\text{Axioms}(G)$ is as follows:

$$\text{Axioms}(G) = \{A(x) \rightarrow G(x, f(x)), G(x_1, x_2) \rightarrow A(x_1) \wedge B(x_2) \wedge R(x_1, x_2)\}$$

For $\text{Check}(\mathcal{O}) = \{G(x_1, x_2) \wedge A(x_2) \rightarrow \text{Cycle}\}$ and $P = \text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$, we have $T_P^\infty = \{A(a), G(a, f(a)), R(a, f(a)), B(f(a)), A(f(a)), \text{Cycle}, \dots\}$. Now \mathcal{O} is not semantically acyclic because $\text{Cycle} \in T_P^\infty$, which indicates that T_P can be applied to F in a repetitive way without terminating.

Semantic acyclicity is a sufficient, but not a necessary termination condition: bottom-up evaluation of $\text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$ can terminate even if \mathcal{O} is not semantically acyclic.

Example 6. Let $\mathcal{O} = \langle \{G\}, \emptyset, \{R(x_1, x_2) \wedge C(x_1) \rightarrow A(x_2)\}, \{A(a), C(a)\} \rangle$ be a DGLP ontology where G , $\text{Check}(\mathcal{O})$, and P are defined as in Example 5. One can see that $\{A(a), C(a), G(a, f(a)), R(a, f(a)), B(f(a)), A(f(a)), \text{Cycle}, G(f(a), f(f(a))), B(f(f(a))), R(f(a), f(f(a)))\}$ is the stable model of P computable by finitely many applications of the T_P operator; however, \mathcal{O} is not semantically acyclic since $\text{Cycle} \in T_P^\infty$.

6 Reasoning with Negation-Free DGLP ontologies

In the present section, we consider the problem of reasoning with a DGLP ontology $\mathcal{O} = \langle DG, \prec, R, F \rangle$ where R is negation-free. Intuitively, one can simply apply the T_P operator to $P = \text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$ and compute $T_P^1, T_P^2, \dots, T_P^i$ and so on. By Theorem 7 for some i we will either reach a fixpoint or derive Cycle. In the former case, we have the stable model of \mathcal{O} (if $\perp \notin T_P^i$), which we can use to decide the relevant reasoning problems; in the latter case, we know that \mathcal{O} is not acyclic.

Theorem 7. *Let $\mathcal{O} = \langle DG, \prec, R, F \rangle$ be a DGLP ontology with R negation-free, and let $P = \text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$. Then, $\text{Cycle} \in T_P^i$ or $T_P^{i+1} = T_P^i$ for some $i \geq 1$.*

By Theorem 7 checking the semantic acyclicity of \mathcal{O} is thus decidable. If the stable model of $\text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$ is infinite, then Cycle is derived; however, the inverse does not hold as shown in Example 6. Furthermore, a stable model of $\text{LP}(\mathcal{O})$, if it exists, is clearly contained in the stable model of $\text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$, and the only possible difference between the two stable models is for the latter to contain Cycle.

7 Reasoning with Stratified Negation-as-Failure

We now extend the reasoning algorithm from Section 6 to the case of DGLP ontologies $\mathcal{O} = \langle DG, \prec, R, F \rangle$ where R contains stratified negation-as-failure. We start by recapitulating several definitions. Let P be a logic program. A *stratification* of P is a mapping $\sigma : P \rightarrow \mathbb{N}$ such that for each rule $r \in P$ the following conditions hold:

- if $B \in \text{body}_P^+(r)$, then $\sigma(r') \leq \sigma(r)$ for each $r' \in P$ with $B \in \text{head}_P(r')$; and
- if $B \in \text{body}_P^-(r)$, then $\sigma(r') < \sigma(r)$ for each $r' \in P$ with $B \in \text{head}_P(r')$.

A logic program P is *stratifiable* if there exists a stratification of P . Moreover, a partition P_1, \dots, P_n of P is a *stratification partition* of P w.r.t. σ if, for each $r \in P$, we have $r \in P_{\sigma(r)}$. The sets P_1, \dots, P_n are called the *strata* of P . Let $U_{P_0}^\infty = T_{P_1}^1$, $U_{P_j}^i = T_{P_j}^i(U_{P_{j-1}}^\infty)$ for $1 \leq j \leq n$ and $i \geq 1$ and $U_{P_j}^\infty = T_{P_j}^\infty(U_{P_{j-1}}^\infty)$. The stable model of P is given by $U_{P_n}^\infty$.

Example 8. Take \mathcal{O} from Example 2 and let $P = \text{LP}(\mathcal{O})$. The mapping $\sigma : P \rightarrow \mathbb{N}$ such that we have $\sigma(\text{Cyclobutane(a)}) = \sigma(s_{\text{Gcb}}) = \sigma(\ell_{\text{Gcb}}) = \sigma(r_{\text{Gcb}}) = \sigma(r_4) = 1$, $\sigma(r_1) = \sigma(r_2) = 2$, and $\sigma(r_3) = 3$, is a stratification of P .

Next we introduce the notion of a DG-stratification, which ensures that the cycle detection rules are assigned to the strata containing the relevant start rules of DGs.

Definition 9 (DG-stratification). *Let $\mathcal{O} = \langle DG, \prec, R, F \rangle$ be a DGLP ontology, let $P = \text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$ and let P_1, \dots, P_n be a stratification partition of P w.r.t. some stratification σ of P . Then, σ is a DG-stratification if*

- for each $G_1, G_2 \in DG$ such that $G_1 \neq G_2$, $G_1 \not\prec G_2$, and $\{s_{G_1}, s_{G_2}\} \subseteq P_i$, we have $\text{ChkPair}(G_1, G_2) \subseteq P_i$, and
- for each $G \in DG$ such that $s_G \in P_i$, we have $\text{ChkSelf}(G) \subseteq P_i$.

The following result shows that, as long as $\text{LP}(\mathcal{O})$ is stratified, one can always assign the cycle checking rules in $\text{Check}(\mathcal{O})$ to the appropriate strata and thus obtain a DG-stratification of $\text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$.

Lemma 1. *Let $\mathcal{O} = \langle DG, \prec, R, F \rangle$ be a DGLP ontology. If σ is a stratification of $\text{LP}(\mathcal{O})$, then σ can be extended to a DG-stratification σ' of $\text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$.*

The following theorem implies that, given a stratifiable DGLP ontology, we can decide whether the ontology is semantically acyclic, and if so, we can compute its stable model and thus solve all relevant reasoning problems.

Theorem 9. *Let \mathcal{O} be a DGLP ontology and $P = \text{LP}(\mathcal{O}) \cup \text{Check}(\mathcal{O})$. If P_1, \dots, P_n is a stratification partition of P w.r.t. a DG-stratification of P , then, for each j with $1 \leq j \leq n$, there exists $i \geq 1$ such that $\text{Cycle} \in U_{P_j}^i$, or $U_{P_j}^{i+1} = U_{P_j}^i$ and $U_{P_j}^i$ is finite.*

8 Implementation Results and Discussion

In order to test the applicability of our approach in practice, we have developed a prototypical implementation based on the XSB system⁵. We used the XSB engine because it supports function symbols, which are needed in the transformation of DGLP ontologies to logic programs. XSB does not compute the stable model(s) of a logic program, but implements query answering as the main reasoning task; by Definitions 8 and 6, this is sufficient for checking acyclicity and subsumption.

To obtain test data, we extracted from the ChEBI ontology seven DGLP ontologies $\mathcal{O}_i = \langle DG_i, \prec, R, F_i \rangle$, $1 \leq i \leq 7$. Each set of description graphs DG_i contained $i \cdot 10$ description graphs, each describing the structure of a particular molecule; the start predicate of each DG is the molecule's name; and the mode of the DG is \Rightarrow . ChEBI describes about 24,000 molecules in total⁶ however, due to the prototypical nature of our implementation we can currently handle only a small subset of ChEBI. Also, since DG_i does not model DGs that imply existence of other DGs, for each \mathcal{O}_i we set $\prec = \emptyset$. Each \mathcal{O}_i contains the same set of rules R that models general chemical knowledge, such as 'cyclobutane is a molecule' and 'a single bond is a bond'; furthermore, R contains rules that classify molecules into five classes T_1 – T_5 shown in Figure 4. Finally, each F_i contains a fact of the form $M_j(m_j)$ for each molecule predicate M_j and fresh individual m_j . Thus, F_i 'instantiates' all description graphs in DG_i , so we can check $\mathcal{O}_i \models M_j \sqsubseteq T_k$ by equivalently checking $\text{LP}(\mathcal{O}_i) \models T_k(m_j)$. All test ontologies are available online⁷.

We conducted the following tests for each \mathcal{O}_i . First, we loaded $\text{LP}(\mathcal{O}_i)$ into XSB. Second, we checked whether \mathcal{O}_i is acyclic. Third, for each class of molecules T_k , $1 \leq k \leq 5$ shown in Table 4, we measured the time needed to test $\mathcal{O}_i \models M_j \sqsubseteq T_k$ for each molecule M_j in \mathcal{O}_i ; each test of the latter form was performed by checking $\text{LP}(\mathcal{O}_i) \models T_k(m_j)$. Figure 4 summarises the loading and classification times; the times needed to check acyclicity are not shown since they were under a second in all cases. The experiments were performed on a desktop computer with 3.7 GB of RAM and Intel Core™ 2 Quad Processors running at 2.5 GHz and 64 bit Linux.

⁵ <http://xsb.sourceforge.net/>

⁶ <http://www.ebi.ac.uk/chebi/statisticsForward.do>

⁷ <http://www.cs.ox.ac.uk/isg/people/despoina.magka/tools/ChEBIClassifier.tar.gz>

No mol.	No rules	Loading time	T ₁	T ₂	T ₃	T ₄	T ₅	Total time
10	1417	2.08	< 0.01	< 0.01	< 0.01	0.36	0.02	2.47
20	5584	8.35	< 0.01	< 0.01	0.02	2.07	0.21	10.66
30	8994	11.35	0.01	< 0.01	0.03	2.23	0.23	13.85
40	14146	16.14	0.01	< 0.01	0.04	2.58	0.29	19.06
50	21842	23.11	0.01	0.01	0.06	3.55	0.41	27.15
60	55602	168.71	0.04	0.02	0.51	109.88	21.68	300.84
70	77932	239.06	0.06	0.03	0.75	172.14	35.08	447.12

T₁: hydrocarbons, T₂: inorganic molecules, T₃: molecules with exactly two carbons,
T₄: molecules with a four-membered ring, T₅: molecules with a benzene ring

Fig. 4. Evaluation results

Our tests have shown all of the ontologies to be acyclic. Furthermore, all tests have correctly classified the relevant molecules into appropriate molecule classes; for example, we were able to conclude that acetylene has exactly two carbons, that cyclobutane has a four-membered ring, and that dinitrogen is inorganic. Please note that none of these inferences can be derived using the approach from [20] due to the lack of negation-as-failure, or using OWL only due to its tree-model property.

All tests were accomplished in a reasonable amount of time: no test required more than a few minutes. Given the prototypical character of our application, we consider these results to be encouraging and we take them as evidence of the practical feasibility of our approach. The most time-intensive test was T₄, which identified molecules containing a four-membered ring. We do not consider this surprising, given that the rule for recognising T₄ contains many atoms in the body and thus requires evaluating a complex join. We noticed, however, that reordering the atoms in the body of the rule significantly reduces reasoning time. Thus, trying to determine an appropriate ordering of rule atoms via join-ordering optimisations, such as the one used for query optimisation in relational databases, might be a useful technique in an optimised DGLP implementation.

9 Conclusions and Future Work

In this paper, we presented an expressive and decidable formalism for the representation of objects with complex structure; additionally, our preliminary evaluation provides evidence that our formalism is practically feasible. In our future work, we shall modify our approach in order to avoid the explicit definition of graph ordering on behalf of the user; furthermore, we plan to investigate whether semantic acyclicity can be combined with other conditions (such as [2]) in order to obtain a more general acyclicity check. Finally, we shall optimise our prototypical implementation in order to obtain a fully-scalable chemical classification system.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook: Theory, Implementation, and Applications. CUP (2007)

2. Baget, J.F., Leclère, M., Mugnier, M.-L., Salvat, E.: On rules with existential variables: Walking the decidability line. *Artif. Intell.* 175(9-10), 1620–1654 (2011)
3. Baral, C., Gelfond, M.: *Logic Programming and Knowledge Representation*. *Journal of Logic Programming* 19, 73–148 (1994)
4. Beeri, C., Vardi, M.Y.: The Implication Problem for Data Dependencies. In: Even, S., Kariv, O. (eds.) *ICALP 1981*. LNCS, vol. 115, pp. 73–85. Springer, Heidelberg (1981)
5. Cali, A., Gottlob, G., Lukasiewicz, T., Marnette, B., Pieris, A.: Datalog+/-: A family of logical knowledge representation and query languages for new applications. In: *LICS (2010)*
6. Fagin, R., Kolaitis, P.G., Miller, R.J., Popa, L.: Data Exchange: Semantics and Query Answering. In: Calvanese, D., Lenzerini, M., Motwani, R. (eds.) *ICDT 2003*. LNCS, vol. 2572, pp. 207–224. Springer, Heidelberg (2002)
7. Grau, B.C., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P.F., Sattler, U.: OWL 2: The next step for OWL. *J. Web Sem.* 6(4), 309–322 (2008)
8. Graves, H.: Representing Product Designs Using a Description Graph Extension to OWL 2. In: *Proc. of the 5th OWLED Workshop (2009)*
9. Grosz, B.N., Horrocks, I., Volz, R., Decker, S.: Description Logic Programs: Combining Logic Programs with Description Logic. In: *WWW (2003)*
10. Hastings, J., Dumontier, M., Hull, D., Horridge, M., Steinbeck, C., Sattler, U., Stevens, R., Hörne, T., Britz, K.: Representing Chemicals using OWL, Description Graphs and Rules. In: *OWLED (2010)*
11. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosz, B., Dean, M.: SWRL: A semantic web rule language combining OWL and RuleML. W3C Member Submission (May 21, 2004), <http://www.w3.org/Submission/SWRL/>
12. Krötzsch, M., Rudolph, S., Hitzler, P.: ELP: Tractable Rules for OWL 2. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) *ISWC 2008*. LNCS, vol. 5318, pp. 649–664. Springer, Heidelberg (2008)
13. Krötzsch, M., Maier, F., Krisnadhi, A., Hitzler, P.: A better uncle for owl: nominal schemas for integrating rules and ontologies. In: Srinivasan, S., Ramamritham, K., Kumar, A., Ravindra, M.P., Bertino, E., Kumar, R. (eds.) *WWW*, pp. 645–654. ACM (2011)
14. Levy, A.Y., Rousset, M.C.: Combining Horn Rules and Description Logics in CARIN. *Artificial Intelligence* 104(1-2), 165–209 (1998)
15. Lutz, C., Areces, C., Horrocks, I., Sattler, U.: Keys, nominals, and concrete domains. *J. of Artificial Intelligence Research* 23, 667–726 (2004)
16. Marnette, B.: Generalized Schema-Mappings: from Termination to Tractability. In: *PODS (2009)*
17. de Matos, P., Alcántara, R., Dekker, A., Ennis, M., Hastings, J., Haug, K., Spiteri, I., Turner, S., Steinbeck, C.: Chemical Entities of Biological Interest: an update. *Nucleic Acids Research* 38(Database-Issue), 249–254 (2010)
18. McCarthy, J.: Circumscription - a form of non-monotonic reasoning. *Artif. Intell.* (1980)
19. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 Web Ontology Language: Profiles, W3C Recommendation (October 27, 2009)
20. Motik, B., Grau, B.C., Horrocks, I., Sattler, U.: Representing Ontologies Using Description Logics, Description Graphs, and Rules. *Artif. Int.* 173, 1275–1309 (2009)
21. Motik, B., Sattler, U., Studer, R.: Query Answering for OWL-DL with Rules. *J. Web Sem.* 3(1), 41–60 (2005)
22. Rector, A.L., Nowlan, W.A., Glowinski, A.: Goals for concept representation in the GALEN project. In: *SCAMC 1993*, pp. 414–418 (1993)
23. Vardi, M.Y.: Why is modal logic so robustly decidable? *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pp. 149–184 (1996)

Extending Description Logic Rules^{*}

David Carral Martínez and Pascal Hitzler

Kno.e.sis Center, Wright State University, Dayton, OH, U.S.A.

Abstract. Description Logics – the logics underpinning the Web Ontology Language OWL – and rules are currently the most prominent paradigms used for modeling knowledge for the Semantic Web. While both of these approaches are based on classical logic, the paradigms also differ significantly, so that naive combinations result in undesirable properties such as undecidability. Recent work has shown that many rules can in fact be expressed in OWL. In this paper we extend this work to include some types of rules previously excluded. We formally define a set of first order logic rules, C-Rules, which can be expressed within OWL extended with role conjunction. We also show that the use of nominal schemas results in even broader coverage.

1 Introduction

Several different paradigms have been devised to model ontologies for the Semantic Web [6]. Currently, the most prominent approaches for modeling this knowledge are description logics (DLs) [1] and rules based on the logic programming paradigm. Although both are based on classical logic, they differ significantly and the search for a satisfactory integration is still ongoing [4, 11].

Even if the DL-based Web Ontology Language OWL [5], a W3C standard, is the main language for modeling ontologies in the Semantic Web, rule-based approaches have also proven very successful. Included in many commercial applications, rules continue to be pursued in parallel to OWL using the Rule Interchange Format RIF [2], also a W3C standard, as a rule exchange layer. Understanding the differences between both paradigms in order to come up with workable combinations has become a major effort in current research.

This paper extends on the work presented in [13] where it has been shown that, in fact, many rules can be expressed in OWL. We extend this work to include some types of rules previously excluded. We formally define C-Rules, a set of rules that can be embedded directly into OWL extended with role conjunction. We also discuss how our approach can be used in conjunction with previous weaker methods for embedding rules based on nominal schemas.

To express C-Rules in DL notation we employ the DL $\mathcal{SROIQ}(\sqcap\exists)$, an extension of \mathcal{SROIQ} [8], which underlies OWL 2 DL. $\mathcal{SROIQ}(\sqcap\exists)$ encompasses \mathcal{SROIQ} adding a restricted form of role conjunction.

^{*} This work was supported by the National Science Foundation under award 1017225 III: Small: TROn – Tractable Reasoning with Ontologies.

To introduce our approach, consider the following rule R which cannot be readily expressed in \mathcal{SROIQ} using known techniques. Although R may not have a single directly equivalent axiom in DL, we can transform it into a set of equisatisfiable statements in first-order predicate logic (FOL).

$$\text{hasFather}(x, y) \wedge \text{hasBrother}(y, z) \wedge \text{hasTeacher}(x, z) \rightarrow \text{TaughtByUncle}(x)$$

The example rule R can be represented as an equisatisfiable set of statements:

- $\text{hasFather}(x, y) \wedge \text{hasBrother}(y, z) \rightarrow \text{hasUncle}(x, z)$
- $\text{hasUncle}(x, z) \wedge \text{hasTeacher}(x, z) \rightarrow \text{hasTeacherAndUncle}(x, z)$
- $\text{hasTeacherAndUncle}(x, z) \rightarrow \text{TaughtByUncle}(x)$

This set of FOL statements can then be translated into

- $\text{hasFather} \circ \text{hasBrother} \sqsubseteq \text{hasUncle}$
- $\text{hasUncle} \sqcap \text{hasTeacher} \sqsubseteq \text{hasTeacherAndUncle}$
- $\exists \text{hasTeacherAndUncle} . \top \sqsubseteq \text{TaughtByUncle}$

and therefore the rule R can be expressed in DL notation.

Although some rules fall under our definition and are expressible using these combinations of role constructors, there are even more complex rules that cannot be simplified using the approach presented in this paper.

A prominently discussed idea for retaining decidability, and still be able to express complex rules, is to restrict the applicability of rules to named individuals. Rules with this kind of semantics are called DL-safe, and the combination of OWL and DL-safe rules is indeed decidable [7,16]. In this paper we also discuss the use of nominal schemas to express complex rules. Nominal schemas are a DL constructor presented in [15] described as “variable nominal classes.” Restricted to stand only for named individuals as DL-safe variables, nominal schemas have the advantage of allowing us to express complex rules in native OWL notation.

The plan for the paper is as follows. After providing some preliminaries in Section 2, the paper continues in Section 3 with the formal definition of C-Rules with unary predicates in the head. Section 4 extends the approach presented in the previous section to rules with binary predicates in the head. Section 5 contains some examples. Section 6 contains the discussion about rules and nominal schemas. Section 7 includes the conclusions of the paper and future work.

2 Preliminaries

We introduce $\mathcal{SROIQ}(\sqcap, \exists)$, a DL fragment that adds role conjunction, in a restricted way, to \mathcal{SROIQ} [8]. Axioms of the form $R_1 \sqcap R_2 \sqsubseteq V$ are allowed in $\mathcal{SROIQ}(\sqcap, \exists)$, where R_1 and R_2 are two (possibly complex) roles.¹

Roles which appear on the right hand sides of axioms of the form $R_1 \sqcap R_2 \sqsubseteq V$ are restricted to only appear in concepts of the form $\exists V.C$. Although this

¹ In a sense, role conjunction was already implicit in [14], and is also used in [13], for a similar purpose.

precondition might look very restrictive, it suffices for the use of role conjunction for expressing rules, as discussed in this paper. As a technical note, in terms of regularity of RBoxes (required for decidability), we assume that for a role V appearing in an axiom $R_1 \sqcap R_2 \sqsubseteq V$ we have that both $R_1 \prec V$ and $R_2 \prec V$ (\prec indicates the order in a regular role hierarchy).

$\mathcal{SROIQ}(\sqcap\exists)$ bears the same semantics as \mathcal{SROIQ} , with the exception of the role conjunction constructor. The formal semantics is as usual (see, e.g., [17]), and for lack of space we do not repeat it here. Note that it follows easily from the arguments laid out in [17] that $\mathcal{SROIQ}(\sqcap\exists)$ is decidable.

2.1 Description Logic Rules and Graph Notation

We define *U-Rules* (respectively, *B-Rules*) as rules of the form $\bigwedge B_i \rightarrow H$, where all B_i are unary or binary predicates and H is a unary (binary) predicate.² We refer to $\bigwedge B_i$ and H as the *body* and the *head* of the rule respectively. We assume without loss of generality that at least one of the variables in the head of the rule also appears in the body of the rule at least once.³

Let R be any given U-Rule or B-Rule. We can define an *undirected graph* G_R derived from R as a set of vertices and edges s.t. every variable in the body of R is a vertex of G_R and G_R contains an edge (t, u) if $S(t, u)$ is a binary predicate in the body of R , and t and u are different variables. Note, that rules containing a predicate of the form $R(x, x)$ where R is a complex role cannot be expressed in \mathcal{SROIQ} , as it will be shown in Section 3.1. Consequently, if this is the case the reduction process cannot be performed. Graphs will be used across the paper to represent and reduce rules in an easy and intuitive way.

Note that constants, and even binary predicates containing only one variable, are not included in the graph. Having a prefixed meaning, constants can be simplified independently and therefore, there is no need to relate them to other elements in the graph. On the other hand, FOL variables, having shared non-fixed meanings, need to keep the links that determine their relation to the predicates in the rule where they appear.

We have defined undirected graphs G_R as sets and consequently there cannot be repetitions amongst its elements. Even if two different binary predicates relate the same pair of terms, in the graph these predicates map to the same single edge. Note also that we work with undirected graphs and therefore, elements (u, t) and (t, u) stand for different representations of the same edge.

For an undirected graph G_R , derived from a U-Rule or a B-rule R , we define a vertex as a *root vertex* r_R if it has been derived from a variable appearing both in the head and the body of the rule. Note that by our standing assumption, there will always be at least one root vertex for any given graph G_R .

We define, for two given vertices t and u , to be *directly connected* if $(t, u) \in G_R$. We define, for two given vertices t and u , to be *connected* as the symmetric

² In our context unary predicates will refer to any kind of valid unary $\mathcal{SROIQ}(\sqcap\exists)$ concept in negation normal form, either in the head or in the body of the rule.

³ Note that, if none of the variables in the head of the rule appears in the body, we can always include one of them in the body using a unary top predicate.

transitive closure of being directly connected. We assume without loss of generality that any two vertices in the graph are connected⁴ We define the *degree of a vertex* $d(t)$ as the number of different edges (t, u) in the graph G_R where t appears.

3 Description Logic Rules in $\mathcal{SROIQ}(\sqcap\exists)$

In this section, we formally describe rules that can be expressed in the DL fragment $\mathcal{SROIQ}(\sqcap\exists)$. While close in general spirit to the brief exhibit in [12, Section 8.3], our treatment is quite different.

Definition 1. *We propose in this definition a formal method to verify if a U-Rule R is expressible in $\mathcal{SROIQ}(\sqcap\exists)$. Given a graph G_R derived from a given rule R , repeat non-deterministically and exhaustively:*

- *Substitute a pair of edges (t, u) and (u, v) by a single edge (t, v) and eliminate vertex u if $d(u) = 2$ and u is a non-root vertex.*

$$G_R := \{G_R \setminus \{(t, u), (u, v), u\} \mid d(u) = 2, u \neq r_R\} \cup \{(t, v)\}$$

- *Eliminate an edge (t, u) and a vertex u where $d(u) = 1$ and u is a non-root vertex.*

$$G_R := \{G_R \setminus \{(t, u), u\} \mid (t, u) \in G_R, d(u) = 1, u \neq r_R\}$$

A rule R is expressible in $\mathcal{SROIQ}(\sqcap\exists)$ if the graph G_R gets reduced to the root vertex after the reduction process.

The reduction process defined in Definition 1 only halts under two different situations. Either the graph G_R gets reduced to a single vertex or, for every non-root vertex u in the graph we have that $d(u) \geq 3$ (possibly after several reduction steps). Steps 1 and 2 presented in the previous reduction process can remove a vertex u if $d(u) = 2$ or $d(u) = 1$ respectively if u is a non-root vertex. It is obvious that a graph containing a non-root vertex u s.t. $d(u) < 3$ can be reduced at least one step further, and a graph where $d(u) \geq 3$ for every non-root vertex $u \in G_R$ cannot be simplified.

The process presented in Definition 1 presents a formal approach to determine if a given rule R is expressible in $\mathcal{SROIQ}(\sqcap\exists)$. In the sequel we explain how, from this graph reduction approach, we can derive a set of $\mathcal{SROIQ}(\sqcap\exists)$ axioms equivalent to the original rule R . Formally stated, the following two lemmas and theorem hold. Their correctness is shown in Section 3.1.

Note that, even if we do not provide an explicit definition for rules that are not expressible in $\mathcal{SROIQ}(\sqcap\exists)$ this can be easily inferred. Rules with four nodes that are fully connected through paths not containing any of those nodes are not expressible. Even if all the other nodes in the paths get simplified these four nodes will form a clique where each one of them has degree three or higher. Therefore, they cannot be reduced using the approach presented in this paper.

⁴ Again, we can connect any two variables in a rule using the universal role—although regularity issues need to be taken into consideration here.

Lemma 1. *Let R be a U -Rule with a derived graph G_R . Every transformation performed on G_R as explained in Definition 1 leads to a new reduced graph $G_{R'}$. Then a new rule R_1 can be constructed from R s.t. we can derive a graph G_{R_1} from R_1 and $G_{R_1} = G_{R'}$. Furthermore, there exist a set of $SR\mathcal{OIQ}(\sqcap\exists)$ statements α_1 s.t. $\{R_1\} \cup \alpha_1 \equiv \{R\}$.*

Definition 2. *The process defined in Lemma 1 can be repeated iteratively producing a new rule from every rule derived from R s.t. $R \equiv R_1 \cup \alpha_1$, $R_1 \equiv R_2 \cup \alpha_2$, ..., $R_{t-1} \equiv R_t \cup \alpha_t$. R_t is a rule with a derived graph G_{R_t} s.t. G_{R_t} cannot be reduced anymore. If G_{R_t} has been reduced to a single vertex we call R_t a terminal rule.*

Lemma 2. *A terminal rule R_t is directly expressible as a $SR\mathcal{OIQ}(\sqcap\exists)$ axiom.*

Intuitively, the simplifications done on the graph will be mirrored in the rule. Through this iterative process we can start reducing the rule, splitting it into several $SR\mathcal{OIQ}$ axioms that preserve equisatisfiability. When the rule gets reduced to a terminal rule it can be directly translated into $SR\mathcal{OIQ}(\sqcap\exists)$. Adding this translation to the set of previous axioms derived in the reduction process we obtain an equivalent set of atoms in $SR\mathcal{OIQ}(\sqcap\exists)$. The following Theorem follows directly from Lemmas 1 and 2.

Theorem 1. *Assume that a rule R is reduced to terminal rule R_t . Then the original knowledge base KB containing R is equisatisfiable w.r.t. a new knowledge base KB' . KB' is obtained from KB by substituting R by $\alpha_1 \cup \dots \cup \alpha_{t-1} \cup \{R_t\}$ where R_t is the direct translation of the terminal rule R_t and $\alpha_1, \dots, \alpha_{t-1}$ are the sets of axioms produced in every step during the reduction process of rule R .*

3.1 Satisfiability Preserving Transformations

We show how to carry out the mentioned graph transformations for a given rule R , preserving equisatisfiability in every step. For every transformation we present a table with three columns. The first column shows the initial subset of R that will be replaced in the next iteration of the reduction process. The second column includes the new simplified subset. By substituting the initial subset by the simplified one in the original rule R we obtain the new simplified rule. Column three shows all the $SR\mathcal{OIQ}(\sqcap\exists)$ statements that we need to add to the knowledge base in order to preserve equivalence.

Rolling Up Unary Predicates. We start by proving that unary predicates can be automatically embedded into binary predicates using role constructors. Therefore, these predicates do not need to be considered when we build the graph to know if a description logic rule R is expressible in $SR\mathcal{OIQ}(\sqcap\exists)$. This technique, called rolification, is presented in [11].

Initial rule subset	Eq Subset	Set α
$V(x, y) \wedge D(y)$	$S(x, y)$	$D \sqsubseteq \exists W.\text{Self}$ $V \circ W \sqsubseteq S$

Hereby, W and S are fresh names not already appearing in the knowledge base.

Proposition 1. *Let KB be a knowledge base containing a U-Rule R s.t. $V(x, y) \wedge D(y)$ appears in R . From KB we can construct an equivalent knowledge base KB' s.t. $KB' := \{KB \setminus R\} \cup \{R'\} \cup \alpha$, where R' is a new U-Rule obtained from R by substituting $V(x, y) \wedge D(y)$ with $S(x, y)$ and the set α contains the axioms $D \equiv \exists W.Self$ and $V \circ W \sqsubseteq S$, with W and S fresh predicate names. We prove that KB and KB' are equisatisfiable knowledge bases.*

Proof. Assume M is a model for the KB . Obviously M models R . From M we can build a model M' for KB' s.t. M' is identical to M except for the mappings $W^{M'} = \{(b, b) \mid b \in D^M\}$ and $S^{M'} = \{(a, b) \mid (a, b) \in V^M \text{ and } (b, b) \in W^{M'}\}$. These mappings are enforced by the new α axioms added to the knowledge base. Obviously, if M models R , M' models axioms $\{R'\} \cup \alpha$ and hence, M' is a model for KB' .

Now assume that M' is a model of $\{R'\} \cup \alpha$. Then we have that $M = M'$ is a model for R . Consider ground instances of the rule and assume that $M \models \bigwedge B_i$. Now we need to prove that $M \models H$. Since we assume that $M \models \bigwedge B_i$ we have that M models every B_i in the body of the rule, in particular $M \models V(a, b) \wedge D(b)$. Hence, $(b, b) \in W^{M'}$ and $(a, b) \in S^{M'}$. Then we have that $M \models \bigwedge B'_i$ where B'_i is the R' body. Therefore $M \models H'$, where H' is the head of the new rule R' . We have that $H' = H$, hence, we have shown that that $M \models H$ being H . Consequently M models R and is a model for KB . □

All the other transformations presented below can be proved in a similar way. We thus refrain from including any more detailed formal arguments for them.

Note that $V(x, y) \wedge D(x)$ can be simplified in a similar way. In the following, W and S are fresh role names in the ontology.

Initial rule subset	Eq Subset	Set α
$V(x, y) \wedge D(x)$	$S(x, y)$	$D \sqsubseteq \exists W.Self$ $W \circ V \sqsubseteq S$

Undirected Graph. The direction of the edges in the graph can be changed using the inverse role constructor. Therefore, there is no need for our algorithm in Definition 11 to check the direction of binary predicates when we apply different simplifications. Below, S is fresh role name in the knowledge base.

Rule Subset	Eq Subset	Set α to the KB
$R(x, y)$	$S(y, x)$	$R^- \sqsubseteq S$

Reducing Vertices of Degree Two

Rule Subset	Eq Subset	Set α to the KB
$R(x, y) \wedge W(y, z)$	$S(x, z)$	$R \circ W \sqsubseteq S$

Hereby, $d(y) = 2$ and S is fresh role name in the knowledge base.

Note that when we simplify a vertex by using a role chain we loose the reference to the term u in the middle of the role chain for a given rule R . Therefore this can only be executed for terms u with a degree of two, without further references in other predicates of the rule R .

Reducing Vertices of Degree One

Rule Subset	Eq Subset	Set α to the KB
$R(x, y) \wedge V(y, z)$	$R(x, y) \wedge D(y)$	$\exists V. \top \sqsubseteq D$

Hereby, $d(z) = 1$ and D is a fresh concept name in the knowledge base.

Note that even if we part from a very similar subset rule as in the previous subsection we follow a different method here. The fresh unary predicate produced can be simplified as shown in earlier sections.

Simplifying Binary Predicates with One Constant

Initial rule subset	Eq Subset	Set α
$V(x, a)$	$S(x)$	$\exists V. \{a\} \sqsubseteq S$

Hereby, S is a fresh unary predicate and a is a constant.

The fresh unary predicate can be simplified as shown in earlier sections. Note that we can always swap the order of the terms in a predicate of the form $V(a, x)$.

Simplifying Binary Predicates of the Form (x, x)

Initial rule subset	Eq Subset	Set α
$V(x, x)$	$S(x)$	$\exists V. \text{Self} \sqsubseteq S$

Hereby, S is a fresh unary predicate.

Using this simplification whenever possible, there is no need to consider these kind of predicates in the graph. Note that this can only be done if V is a simple role w.r.t. role box in the knowledge base. To retain decidability *SRDIOQ* does not allow to use complex in a concept of the form $\exists C. \text{Self}$.

Unifying Binary Predicates

Rule Subset	Eq Subset	Set α to the KB
$R(x, y) \wedge V(x, y)$	$S(x, y)$	$R \sqcap V \sqsubseteq S$

Hereby, S is a fresh role name in the knowledge base.

Any pair of binary predicates can be unified if both contain the same pair of variables. Note that, even if the variables do not appear in the same order, we can use the inverse role construct to align them. We assume without loss of generality that every pair of binary predicates containing the same pair of variables in a rule is automatically unified and therefore, we can define graphs as sets without repetitions of the same edge. Note that the unification of binary predicates needs to be done with a higher priority than other transformations, such as the reduction of vertices of degree two. If not cycles may be reduced to predicates of the form $R(x, x)$ where R is a complex role that cannot be simplified using the $\exists R. \text{Self}$.

The transformations just shown correspond to graph transformations as mentioned in Definition 1. The arguments just given thus constitute a proof of Lemma 1. Note that the order of the transformations is non-deterministic. This is a kind of “don’t care” determinism, where the order in which we apply the rules does not matter. We further elaborate about this in Section 5.

Translating Terminal Rules. A terminal rule R is a rule of the form $\bigwedge B_i \rightarrow H$. We have that the body of the rule $\bigwedge B_i$ contains one, and at most one, free FOL variable x appearing only once in a unary predicate of the form $B(x)$ (the graph has been reduced to the root vertex, and therefore, there is only one variable left appearing only once). The body $\bigwedge B_i$ might also contain other predicates of the form $C(a)$ or $R(b, c)$ s.t. a , b , and c are constants. The head H is composed of a single unary predicate $H(x)$ s.t. x is the same free variable that appears in the body.

A terminal rule R is translated into a DL inclusion axiom of the form $\prod B_i \sqsubseteq H$. This axiom contains a fresh concept H on the right hand side of the role inclusion axiom and a concept intersection on the left hand side featuring the next elements:

- A fresh concept B standing for the unary predicate $B(x)$ s.t. x is the only free variable appearing in the rule.
- A concept $\exists U.(C \sqcap \{a\})$ for every unary predicate of the form $C(a)$ appearing in the body where a is a constant.
- A concept $\exists U.(\{b\} \sqcap \exists R.\{c\})$ for every binary predicate of the form $R(b, c)$ appearing in the body of the rule where b and c are constants.

The argument just given also constitutes a proof of Lemma 2.

4 Rules with Binary Predicates in the Head

We can extend our approach to cover rules with binary predicates in the head. As already stated, at least one variable in the head of the rule needs to appear in the body. Attending to this fact we need to consider two different situations.

If only one of the terms in the head of the rule appears in the body the simplification is straightforward. We just need to substitute the binary predicate $R(x, y)$ in the head by a fresh unary predicate $C(x)$ and add the axiom $\exists R.\top \sqsubseteq C$ to the $SR\text{OIQ}(\sqcap\exists)$ knowledge base. After this modification the rule can be reduced using the same approach presented in the previous section.

If both variables appearing in the head of the rule appear in the body we need to slightly modify our method presented in Section 3. In this case, we consider both variables in the head as root vertices. Now, if the rule is expressible in $SR\text{OIQ}(\sqcap\exists)$ the graph gets reduced to a single edge containing both variables in the head. This new kind of terminal rule can be expressed in $SR\text{OIQ}(\sqcap\exists)$ using a role inclusion axiom $S \sqsubseteq R$.

Theorem 2. *Let R be a B-Rule s.t. $S(x, y)$ is the predicate in the head H of R and both x and y appear in the body $\bigwedge B_i$ of the rule.*

Given a graph G_R derived from rule R , where we consider both x and y to be root vertices. Then exhaustively apply steps 1 and 2 from Definition 7.

If after the reduction process, the graph G_R gets reduced to a single edge, then rule R is expressible as a $SR\text{OIQ}(\sqcap\exists)$ axiom.

⁵ Assume x is the root vertex. Otherwise use the inverse role constructor to change the order of the terms in the predicate.

Again, we see that any G_R where every vertex u has $d(u) \geq 3$ (possibly obtained after several reduction steps) cannot be simplified. Otherwise the graph can be reduced to a single edge (u, t) s.t. $u \in H$ and $t \in H$ with H the head of the rule. Note that the procedure is almost the same as in Section 3, except for the accepting condition.

The process to translate the rule into a set of equivalent $\mathcal{SROIQ}(\sqcap, \sqsupseteq)$ statements and proofs remain the same as the one presented in Section 3 except for the trivial translation of the terminal rule.

It is important to remark that in some cases a B-Rule may not be expressible while a U-Rule with the same body is. The second vertex might block a possible role reduction forbidding further simplifications. As an example we have that $R_1(x, y) \wedge R_2(x, w) \wedge R_3(w, y) \wedge R_4(y, z) \wedge R_5(w, z) \rightarrow C(x)$ is expressible in $\mathcal{SROIQ}(\sqcap, \sqsupseteq)$ using our approach, while $R_1(x, y) \wedge R_2(x, w) \wedge R_3(w, y) \wedge R_4(y, z) \wedge R_5(w, z) \rightarrow C(x, z)$ is not.

Definition 3. *Formally, by C-Rules we mean the collection of all rules which are U-Rules or B-Rules and which are expressible in \mathcal{SROIQ} using the approach presented in this paper.*

5 Examples

We start with a worked example for our transformation. As initial rule, we use

$$A(x, y) \wedge B(y) \wedge C(z, y) \wedge D(y, z) \wedge E(x, a) \wedge F(x, z) \wedge Y(a, b) \rightarrow Z(x)$$

where a and b are constant and x, y and z are free variables. Transformations following the discussion from Section 3 are detailed in Table 1.

Note that the rule listed in step 6 of Table 1 can already be directly translated to $\mathcal{SROIQ}(\sqcap, \sqsupseteq)$ as $\exists M.\top \sqcap \exists E.\{a\} \sqcap \exists U.\{\{a\} \sqcap \exists Y.\{b\}\} \sqsubseteq Z$. But to improve readability of the paper, our rule reduction approach has been presented in a simpler form, avoiding such shortcuts. So, although the method shown is sound and correct, there are U-Rules and B-Rules, as the one presented in the example, where at some step of the reduction process no further simplifications are strictly required. An earlier translation of the rule reduces the number of statements that need to be added to the knowledge base. Recall, in particular, that rules with tree shaped graphs are directly expressible in DL [11, 13].

Also, we have that reductions according to our transformations are applied non-deterministically. Although any rule reduction leading to a terminal rule is essentially correct, there might be differences in the set of axioms added to the knowledge base. For example, let R be a U-Rule containing the binary predicates $A(x, y)$ and $B(y, z)$ s.t. both y and z are variables not appearing anywhere else in the rule (hence, we have that $d(y) = 2$ and $d(z) = 1$). In the next reduction step, we can decide which variable, y or z , we want to erase.

Assuming we want to reduce $A(x, y)$ and $B(y, z)$ to $Z(x)$, there are two different ways of doing so, namely (1) first reducing y , and (2) first reducing z . In the first case, we end up with two axioms $A \circ B \sqsubseteq C$ and $\exists C.\top \sqsubseteq Z$, while in the

Table 1. Reduction example. For every step substitute the rule in the previous row by the one in the current one and add the axioms on the second column to the knowledge base.

Step	Add to KB	Rule
1. Original Rule		$A(x, y) \wedge B(y) \wedge C(z, y) \wedge D(y, z) \wedge E(x, a) \wedge F(x, z) \wedge Y(a, b) \rightarrow Z(x)$
2. Invert C	$C^- \sqsubseteq H$	$A(x, y) \wedge B(y) \wedge H(y, z) \wedge D(y, z) \wedge E(x, a) \wedge F(x, z) \wedge Y(a, b) \rightarrow Z(x)$
3. Intersect D and H	$D \sqcap H \sqsubseteq I$	$A(x, y) \wedge B(y) \wedge I(y, z) \wedge E(x, a) \wedge F(x, z) \wedge Y(a, b) \rightarrow Z(x)$
4. Role Up B	$B \sqsubseteq \exists J.\text{Self}$ $A \circ J \sqsubseteq K$	$K(x, y) \wedge I(y, z) \wedge E(x, a) \wedge F(x, z) \wedge Y(a, b) \rightarrow Z(x)$
5. Simplify E	$\exists E.\{a\} \sqsubseteq L$	$K(x, y) \wedge I(y, z) \wedge L(x) \wedge F(x, z) \wedge Y(a, b) \rightarrow Z(x)$
6. Role Up L	$L \sqsubseteq \exists M.\text{Self}$ $M \circ F \sqsubseteq N$	$K(x, y) \wedge I(y, z) \wedge N(x, z) \wedge Y(a, b) \rightarrow Z(x)$
7. Reduce y	$K \circ I \sqsubseteq O$	$O(x, z) \wedge N(x, z) \wedge Y(a, b) \rightarrow Z(x)$
8. Intersect N and O	$N \sqcap O \sqsubseteq P$	$P(x, z) \wedge Y(a, b) \rightarrow Z(x)$
9. Reduce z	$\exists P.\top \sqsubseteq Q$	$Q(x) \wedge Y(a, b) \rightarrow Z(x)$
10. Translate Terminal Rule		$Q \sqcap \exists U.\{\{a\} \sqcap \exists Y.\{b\}\} \sqsubseteq Z$

second case four axioms are required $\exists B.\top \sqsubseteq D$, $D \sqsubseteq \exists E.\text{Self}$, $A \circ E \sqsubseteq F$, and $\exists F.\top \sqsubseteq Z$. Note that giving priority to the reduction of variables with degree 2 reduces the number of required axioms to preserve equisatisfiability.

The regularity issue. In order to preserve decidability, $\mathcal{SROIQ}(\sqcap\exists)$ enforces a strict partial order on complex roles (known as the *regularity* condition). When a C-Rule R is translated into \mathcal{SROIQ} , we add many new complex role inclusions axioms to the Rbox. These new roles may violate the partial order established by previous roles or even contradict role regularity by themselves. After the reduction of a C-Rule and the inclusion of new produced $\mathcal{SROIQ}(\sqcap\exists)$ axioms, role regularity needs to be carefully checked in order to preserve decidability.

It is important to note that only the translation of expressible B-Rules might cause these role regularity violations. Although the role regularity hierarchy is modified in many steps of our rule reduction approach note that for every statement $R \prec S$ added we have that S is a fresh role. Fresh roles do not appear in any other part of the role hierarchy and therefore they cannot produce violations of the irreflexive order.

The only step of the process where the RBox may loose its regularity is in the translation of a terminal B-Rule. Adding this last axiom of the form $R \sqsubseteq S$ also adds the statement $R \prec S$ to the role hierarchy where S is a role which may have appear in any other part of the knowledge base.

Let us look at an example of a knowledge base where the reduction of some of the rules leads to role regularity violations. Let KB be a knowledge base containing the following rule.

$\text{TeacherOf}(y, x) \wedge \text{ReviewerOf}(y, z) \wedge \text{AuthorOf}(x, z) \rightarrow \text{IllegalReviewerOf}(y, z)$

This rule places a pair of individuals under the binary predicate IllegalReviewerOf if the first is a teacher of the student who is the author of the reviewed paper. It can be transformed into the following set of $\mathcal{SROIQ}(\sqcap, \sqsupseteq)$ axioms.

$$\begin{aligned} \text{TeacherOf} \circ \text{AuthorOf} &\sqsubseteq R_1 \\ \text{ReviewerOf} \sqcap R_1 &\sqsubseteq R_2 \\ R_2 &\sqsubseteq \text{IllegalReviewer} \end{aligned}$$

From these axioms, we obtain the relations $\text{TeacherOf} \prec R_1$, $\text{AuthorOf} \prec R_1$, $\text{ReviewerOf} \prec R_2$, $R_1 \prec R_2$, and $R_2 \prec \text{IllegalReviewer}$, which entail the statement $\text{ReviewerOf} \prec \text{IllegalReviewerOf}$. It would be natural to also add the axiom $\text{IllegalReviewerOf} \sqsubseteq \text{ReviewerOf}$ to the same ontology. However, the inclusion of this axiom adds the statement $\text{IllegalReviewerOf} \prec \text{ReviewerOf}$ which then violates role regularity.

A workaround to this issue, however, is possible, namely by employing nominal schemas, and we will return to this issue at the end of the next section.

6 Using Nominal Schemas and $\mathcal{SROIQV}(\sqcap, \sqsupseteq)$

In earlier sections of this paper we have shown how to translate some FOL rules into DL notation. Although some rules can be translated to $\mathcal{SROIQ}(\sqcap, \sqsupseteq)$ using the presented approach there are still more complex rules that cannot be simplified in the same way. To express these rules we employ the DL $\mathcal{SROIQV}(\sqcap, \sqsupseteq)$.

$\mathcal{SROIQV}(\sqcap, \sqsupseteq)$ adds nominal schemas, a DL constructor that can be used as "variable nominal classes," to the previously described $\mathcal{SROIQ}(\sqcap, \sqsupseteq)$. We will refrain from introducing all formal details and refer the reader to [9,10,11,15] for this. While the semantic intuition behind nominal schemas is the same as that behind DL-safe variables, nominal schemas integrate seamlessly with DL syntax. As a consequence, the DL fragment $\mathcal{SROIQV}(\sqcap, \sqsupseteq)$ encompasses DL-safe variables while staying within the DL/OWL language paradigm avoiding the use of hybrid approaches.

Using these nominal schemas we are able to express FOL rules that are not part of the treatment in Sections 3 and 4. Consider, for example, the rule

$$R_1(x, y) \wedge R_2(x, z) \wedge R_3(x, w) \wedge R_4(y, z) \wedge R_5(y, w) \wedge R_6(w, z) \rightarrow C(x). \quad (1)$$

Using $\{z\}$ and $\{w\}$ as nominal schemas, we can express it as

$$\exists R_1. (\exists R_4. \{z\} \sqcap \exists R_5. \{w\}) \sqcap \exists R_2. \{z\} \sqcap \exists R_3. (\{w\} \sqcap \exists R_6. \{z\}) \sqsubseteq C$$

Note that, as already stated, nominal schemas do not share the same semantics defined for FOL variables. Nominal schemas, as DL-safe variables, are restricted to stand only for nominals which are explicitly present in the knowledge base,

while FOL variables can represent both named and unknown individuals. Therefore, the statements presented in the example just given are not strictly equivalent. Despite this fact, nominal schemas allow us to retain most of the entailments from the original FOL axiom without increasing the worst-case complexity of the DL fragment.

Although nominal schemas do not increase the worst-case complexity of the language [15], the number of different nominal schemas per axiom can affect the performance of the reasoning process [3,10]. It is therefore desirable to use as few nominal schemas as possible.

We now discuss two different ways of translating complex rules into DLs. First we prove the following.

Theorem 3. *Any U-Rule or B-Rule R containing m different free variables, where $m > 3$, can be directly expressed in DL using n nominal schemas s.t. $n \leq m - 2$.*

Proof. Given a rule R , firstly role up to simplify all binary predicates containing one constant as shown in Section 3. All binary predicates in the rule containing the same pair of variables are also replaced by a single binary predicate as described under *Unifying Binary Predicates* in Section 3.

Due to these transformations, we can now assume without loss of generality that the rule R contains only unary predicates with a constant, binary predicates with two constants, and binary predicates with two variables as arguments.

Now choose two variables x and y s.t. x is a root vertex and y is not. Using the inverse role construct we can now swap arguments in binary predicates s.t. x is always appearing in the first argument and y is in the first argument of every predicate where the other variable is not x . The variables selected will be the only ones not substituted by a nominal schema in the translated rule.

The rule body is now translated as shown in Table 2. The resulting DL expressions are joined by conjunction. $\bigwedge B_i(y)$, $R(x, y)$, and $\bigwedge R_i(y, v_i)$ are all the predicates where y appears.

Table 2. Translating Rules with Nominal Schemas

Predicate type	FOL	DL
Unary Predicates with 1 constant	$B(a)$	$\exists U.(\{a\} \sqcap B)$
Binary Predicates with 2 constants	$R(a, b)$	$\exists U.(\{a\} \sqcap \exists R.\{b\})$
Binary Predicates containing x and not y	$R(x, v)$	$\exists R.\{v\}$
Unary Predicates containing x	$B(x)$	B
Binary and Unary Predicates containing y	$R(x, y) \wedge \bigwedge B_i(y)$ $\wedge \bigwedge R_i(y, v_i)$	$\exists R.(\bigcap B_i \sqcap \bigcap R_i.\{v_i\})$
Unary Predicates not containing x, y , or constants	$B(v)$	$\exists U.(\{v\} \sqcap B)$
Binary Predicates not containing x, y , or constants	$R(v, u)$	$\exists U.(\{v\} \sqcap \exists R.\{u\})$

Finally, the head $H(x)$ can be rewritten into the concept H (or if it is a binary predicate $H(x, z)$, a concept of the form $\exists H.\{z\}$), and the implication arrow replaced by class inclusion \sqsubseteq .

It is straightforward to formally verify the correctness of this transformation, and parts of the proof are similar to the correctness proof from [15] for the embedding of binary Datalog into \mathcal{SROIQV} .

Clearly, the number of nominal schemas used to represent rule R is $n - 2$, the total number of free variables minus 2. \square

With the transformation just given, rule (II) can be rewritten as

$$\exists R_1.(\exists R_4.\{z\} \sqcap \exists R_5.\{w\}) \sqcap \exists R_2.\{z\} \sqcap \exists R_3.\{w\} \sqcap \exists U.(\{w\} \sqcap \exists R_6.\{z\}) \sqsubseteq C$$

As another example, the following rule transforms into the subsequent axiom.

$$\begin{aligned} R_1(x, y) \wedge R_2(y, z) \wedge R_3(w, z) \wedge R_4(x, z) \wedge R_5(y, w) \wedge R_6(w, u) \wedge R_7(y, u) \\ \rightarrow H(x, u) \\ \exists R_1.(\exists R_2.\{z\} \sqcap \exists R_5.\{w\} \sqcap \exists R_7.\{u\}) \sqcap \exists U.(\exists \{w\} \sqcap \exists R_4.\{z\}) \\ \sqcap \exists R_4.\{z\} \sqcap \exists U.(\{w\} \sqcap \exists R_6.\{u\}) \sqsubseteq \exists H.\{u\} \end{aligned}$$

Theorem 4. *Any U-Rule or B-Rule R containing m different free variables can be expressed in DL by fully grounding $m - 3$ free variables in R .*

Proof. By grounding every variable but three in the rule to named individuals we end up with a larger number of rules s.t. each one of them contains only three different free variables.⁶ All these new grounded rules are expressible in DL using the approach presented in Section 3 of this paper. \square

While the first of the approaches just mentioned allows us to represent all knowledge in $\mathcal{SROIQV}(\sqcap, \exists)$, the second one, although initially looking more efficient, requires preprocessing steps. Further research and algorithms are required to smartly deal with nominal schemas other than through such grounding, a cumbersome technique that requires too much space and time for current reasoners [3, 10].

Let us finally return to the regularity issue discussed at the very end of Section 5. In the example discussed there, if we desire to also add the statement $\text{IllegalReviewerOf} \sqsubseteq \text{ReviewerOf}$ to the knowledge base, we cannot do so directly without violating regularity. Using nominal schemas, however, we can weaken this axiom to the form

$$\exists \text{IllegalReviewerOf}.\{x\} \sqsubseteq \exists \text{ReviewerOf}.\{x\}$$

(or, e.g., to

$$\exists \text{IllegalReviewerOf}^-\{x\} \sqsubseteq \exists \text{ReviewerOf}^-\{x\}$$

⁶ Note that any rule with three variables can be reduced using our approach. Having only three nodes in the graph for all of them we have that $d(u) \leq 2$ and therefore all of them can be reduced.

or to both), where $\{x\}$ is a nominal schema. Essentially, this means that the role inclusion will apply in case the first argument or the second argument (the filler) is a known individual. I.e., the individuals connected by the `IllegalReviewerOf` property are not both, are unnamed. While this is weaker than the standard semantics, it should provide a viable workaround in many cases. Also note that, alternatively, the regularity violation could be avoided by using a similarly weakened form of any of the other statements involved in the violation.

7 Conclusions and Future Work

This paper presents an extension of previous work on Description Logic Rules. We specify a translation of rules into OWL extended by role conjunction (more precisely, the description logic $\mathcal{SROIQ}(\sqcap\exists)$), which strengthens previously obtained such translations. In essence, our work shows that the fragment of Datalog which can be expressed in description logics is larger than previously assumed.

We furthermore included a discussion proposing two approaches to express more complex rules within the DL notation. Two different options are considered, the use of nominal schemas and fully grounding of some variables to named individuals. While the former might present a higher complexity, it allows to express these rules in native DL/OWL notation and avoid some cumbersome preprocessing steps.

Future work includes the development of a goal directed algorithm that can solve inference problems in $\mathcal{SROIQV}(\sqcap\exists)$ possibly including some other role constructs (probably the extension of a current tableau algorithm). This algorithm could serve as basis for practical implementations of reasoners that include role constructs amongst their features.

Also, the development of APIs that can automatically validate FOL rules as DL expressible and translate them into sets of equisatisfiable OWL axioms might be a very useful tool. Although some aspects of modeling ontologies, such as building concept hierarchies, are very intuitive when we work with DL/OWL languages, the translation of DL rules, as shown in this paper, may not be so straightforward for users that do not have a strong background in more formal logics. Additional tool support will be required to provide convenient modeling interfaces.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications, 2nd edn. Cambridge University Press (2007)
2. de Bruijn, J.: RIF RDF and OWL Compatibility. W3C Recommendation (June 22, 2010), <http://www.w3.org/TR/rif-rdf-owl/>
3. Carral Martínez, D., Krisnadhi, A., Maier, F., Sengupta, K., Hitzler, P.: Reconciling OWL and rules. Tech. rep., Kno.e.sis Center, Wright State University, Dayton, Ohio, U.S.A. (2011), <http://www.pascal-hitzler.de/>

4. Hitzler, P., Parsia, B.: Ontologies and rules. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, 2nd edn., pp. 111–132. Springer (2009)
5. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S. (eds.): OWL 2 Web Ontology Language: Primer. W3C Recommendation (October 27, 2009), <http://www.w3.org/TR/owl2-primer/>
6. Hitzler, P., Krötzsch, M., Rudolph, S.: Foundations of Semantic Web Technologies. Chapman & Hall/CRC (2009)
7. Horrocks, I., Patel-Schneider, P.F., Bechhofer, S., Tsarkov, D.: OWL Rules: A proposal and prototype implementation. Journal of Web Semantics 3(1), 23–40 (2005)
8. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible *SR_OI_Q*. In: Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006), pp. 57–67. AAAI Press (2006)
9. Knorr, M., Hitzler, P., Maier, F.: Reconciling OWL and non-monotonic rules for the Semantic Web. Tech. rep., Kno.e.sis Center, Wright State University, Dayton, OH, U.S.A. (2011), <http://www.pascal-hitzler.de/>
10. Krisnadhi, A., Hitzler, P.: A tableau algorithm for description logics with nominal schemas. Tech. rep., Kno.e.sis Center, Wright State University, Dayton, OH, U.S.A. (2011), <http://www.pascal-hitzler.de/>
11. Krisnadhi, A., Maier, F., Hitzler, P.: OWL and Rules. In: Polleres, A., d’Amato, C., Arenas, M., Handschuh, S., Kroner, P., Ossowski, S., Patel-Schneider, P. (eds.) Reasoning Web 2011. LNCS, vol. 6848, pp. 382–415. Springer, Heidelberg (2011)
12. Krötzsch, M.: Description Logic Rules, Studies on the Semantic Web, vol. 008. IOS Press/AKA (2010)
13. Krötzsch, M., Rudolph, S., Hitzler, P.: Description logic rules. In: Ghallab, M., Spyropoulos, C.D., Fakotakis, N., Avouris, N.M. (eds.) Proceeding of the 18th European Conference on Artificial Intelligence, Patras, Greece, July 21–25, pp. 80–84. IOS Press, Amsterdam (2008)
14. Krötzsch, M., Rudolph, S., Hitzler, P.: ELP: Tractable rules for OWL 2. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 649–664. Springer, Heidelberg (2008)
15. Krötzsch, M., Maier, F., Krisnadhi, A.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), pp. 645–654. ACM (2011)
16. Motik, B., Sattler, U., Studer, R.: Query answering for OWL DL with rules. J. of Web Semantics 3(1), 41–60 (2005)
17. Rudolph, S., Krötzsch, M., Hitzler, P.: Cheap Boolean Role Constructors for Description Logics. In: Hölldobler, S., Lutz, C., Wansing, H. (eds.) JELIA 2008. LNCS (LNAI), vol. 5293, pp. 362–374. Springer, Heidelberg (2008)

Prexto: Query Rewriting under Extensional Constraints in *DL-Lite*

Riccardo Rosati

Dipartimento di Ingegneria Informatica, Automatica e Gestionale Antonio Ruberti
Sapienza Università di Roma, Italy
lastname@dis.uniroma1.it

Abstract. In this paper we present **Prexto**, an algorithm for computing a perfect rewriting of unions of conjunctive queries over ontologies expressed in the description logic *DL-Lite_A*. The main novelty of **Prexto** lies in the fact that it constitutes the first technique for query rewriting over ontologies which fully exploits *extensional constraints* to optimize query rewriting. In addition, **Prexto** makes use of functional role axioms and of concept and role disjointness axioms to optimize the size of the rewritten query. We show that these optimizations allow **Prexto** to outperform the existing query rewriting techniques for *DL-Lite* in practical cases.

1 Introduction

The *DL-Lite* family of description logics [42] is currently one of the most studied ontology specification languages. *DL-Lite* constitutes the basis of the OWL2 QL language [1], which is part of the standard W3C OWL2 ontology specification language. The distinguishing feature of *DL-Lite* is to identify ontology languages in which expressive queries, in particular, unions of conjunctive queries (UCQs), over the ontology can be efficiently answered. Therefore, query answering is the most studied reasoning task in *DL-Lite* (see, e.g., [13,9,7,15,6,5]).

The most common approach to query answering in *DL-Lite* is through query rewriting. This approach consists of computing a so-called *perfect rewriting* of the query with respect to a TBox: the perfect rewriting of a query q for a TBox \mathcal{T} is a query q' that can be evaluated on the ABox only and produces the same results as if q were evaluated on both the TBox and the ABox. This approach is particularly interesting in *DL-Lite*, because, for every UCQ q , query q' can be expressed in first-order logic (i.e., SQL), therefore query answering can be delegated to a relational DBMS, since it can be reduced to the evaluation of an SQL query on the database storing the ABox.

The shortcoming of the query rewriting approach is that the size of the rewritten query may be exponential with respect to the size of the original query. In particular, this is true when the rewritten query is in disjunctive normal form, i.e., is an UCQ. On the other hand, [5] shows the existence of polynomial perfect rewritings of the query in nonrecursive datalog.

However, it turns out that the disjunctive normal form is necessary for practical applications of the query rewriting technique, since queries of more complex forms, once translated in SQL, produce queries with nested subexpressions that, in general, cannot

be evaluated efficiently by current DBMSs. So, while in some cases resorting to more compact and structurally more complex perfect rewritings may be convenient, in general this strategy does not solve the problem of arriving at an SQL expression that can be effectively evaluated on the database.

In this scenario, a very interesting way to limit the size of the rewritten UCQ has been proposed in [11]. This approach proposes the use of the so-called *ABox dependencies* to optimize query rewriting in *DL-Lite_A*. ABox dependencies are inclusions between concepts and roles which are interpreted as integrity constraints over the ABox: in other words, the ABox is guaranteed to satisfy such constraints. In the presence of such constraints, the query answering process can be optimized, since this additional knowledge about the extensions of concepts and roles in the ABox can be exploited for optimizing query answering. Intuitively, the presence of ABox dependencies acts in a complementary way with respect to TBox assertions: while the latter complicate query rewriting, the former simplify it, since they state that some of the TBox assertions are already satisfied by the ABox.

As explained in [11], ABox dependencies have a real practical interest, since they naturally arise in many applications of ontologies, and in particular in ontology-based data access (OBDA) applications, in which a DL ontology acts as a virtual global schema for accessing data stored in external sources, and such sources are connected through declarative mappings to the global ontology. It turns out that, in practical cases, many ABox dependencies may be (automatically) derived from the mappings between the ontology and the data sources.

In this paper, we present an approach that follows the ideas of [11]. More specifically, we present **Prexto**, an algorithm for computing a perfect rewriting of a UCQ in the description logic *DL-Lite_A*. **Prexto** is based on the query rewriting algorithm **Presto** [13]: with respect to the previous technique, **Prexto** has been designed to fully exploit the presence of extensional constraints to optimize the size of the rewriting; moreover, differently from **Presto**, it also uses concept and role disjointness assertions, as well as role functionality assertions, to reduce the size of the rewritten query.

As already observed in [11], the way extensional constraints interact with reasoning, and in particular query answering, is not trivial at all: e.g., [11] defines a complex condition for the deletion of a concept (or role) inclusion from the TBox due to the presence of extensional constraints. In our approach, we use extensional constraints in a very different way from [11], which uses such constraints to “deactivate” corresponding TBox assertions in the TBox: conversely, we are able to define significant query minimizations even for extensional constraints for which there exists no corresponding TBox assertions. Based on these ideas, we define the **Prexto** algorithm: in particular, we restructure and extend the **Presto** query rewriting algorithm to fully exploit the presence of extensional constraints.

Finally, we show that the above optimizations allow **Prexto** to outperform the existing query rewriting techniques for *DL-Lite* in practical cases. In particular, we compare **Prexto** both with **Presto** and with the optimization presented in [11].

The paper is structured as follows. After some preliminaries, in Section 3 we introduce extensional constraints and the notion of extensional constraint Box (EBox). In Section 4 we discuss the interaction between intensional and extensional constraints in

query answering. Then, in Section 5 we present the **Prexto** query rewriting algorithm, and in Section 6 we compare **Prexto** with existing techniques for query rewriting in $DL-Lite_{\mathcal{A}}$. We conclude in Section 7

2 The Description Logic $DL-Lite_{\mathcal{A}}$

A Description Logic ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ consists of a TBox \mathcal{T} , representing intensional knowledge, and an ABox \mathcal{A} representing extensional knowledge. $\Gamma_{\mathcal{O}}$ will denote the alphabet of the ontology, that is, the union of the predicate symbols occurring in \mathcal{T} and \mathcal{A} , whereas Γ_C will denote the alphabet of constant symbols occurring in \mathcal{A} .

In this paper we consider ontologies specified in $DL-Lite_{\mathcal{A}}$ [10], a member of the $DL-Lite$ family of tractable Description Logics. $DL-Lite_{\mathcal{A}}$ 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:

$$\begin{array}{ll}
 B \longrightarrow A & | \exists Q & | \delta(U) & E \longrightarrow \rho(U) \\
 C \longrightarrow B & | \neg B & F \longrightarrow \top_D & | T_1 & | \dots & | T_n \\
 Q \longrightarrow P & | P^- & V \longrightarrow U & | \neg U \\
 R \longrightarrow Q & | \neg Q
 \end{array}$$

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, \dots, T_n are n pairwise disjoint unbounded value-domains. A $DL-Lite_{\mathcal{A}}$ TBox \mathcal{T} is a finite set of assertions of the form

$$B \sqsubseteq C \quad Q \sqsubseteq R \quad E \sqsubseteq F \quad U \sqsubseteq V \quad (\text{funct } Q) \quad (\text{funct } U)$$

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 $DL-Lite_{\mathcal{A}}$ 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). We call *concept disjointness assertions* the assertions of the form $B_1 \sqsubseteq \neg B_2$, and call *role disjointness assertions* the assertions of the form $Q_1 \sqsubseteq \neg Q_2$.

A $DL-Lite_{\mathcal{A}}$ ABox \mathcal{A} is a finite set of membership (or instance) assertions of the forms $A(a)$, $P(a, b)$, and $U(a, v)$, where A , P , and U are as above, a and b belong to $\Gamma_{\mathcal{O}}$, the subset of Γ_C containing object constants, and v belongs to Γ_V , the subset of Γ_C containing value constants, where $\{\Gamma_{\mathcal{O}}, \Gamma_V\}$ is a partition of Γ_C .

The semantics of a $DL-Lite_{\mathcal{A}}$ ontology is given in terms of first-order logic (FOL) interpretations \mathcal{I} over a non-empty domain $\Delta^{\mathcal{I}}$ such that $\Delta^{\mathcal{I}} = \Delta_V \cup \Delta_{\mathcal{O}}^{\mathcal{I}}$, where $\Delta_{\mathcal{O}}^{\mathcal{I}}$ is the domain used to interpret object constants in $\Gamma_{\mathcal{O}}$, and Δ_V is the fixed domain (disjoint from $\Delta_{\mathcal{O}}^{\mathcal{I}}$) used to interpret data values. Furthermore, in $DL-Lite_{\mathcal{A}}$ the Unique

Name Assumption (UNA) is adopted, i.e., in every interpretation \mathcal{I} , and for every pair $c_1, c_2 \in I_C$, if $c_1 \neq c_2$ then $c_1^{\mathcal{I}} \neq c_2^{\mathcal{I}}$. The notion of satisfaction of inclusion, disjointness, functionality, and instance assertions in an interpretation is the usual one in DL ontologies (we refer the reader to [10] for more details).

We denote with $Mod(\mathcal{O})$ the set of models of an ontology \mathcal{O} , i.e., the set of FOL interpretations that satisfy all the TBox and ABox assertions in \mathcal{O} . An ontology is *inconsistent* if $Mod(\mathcal{O}) = \emptyset$ (otherwise, \mathcal{O} is called *consistent*). As usual, an ontology \mathcal{O} entails an assertion ϕ , denoted $\mathcal{O} \models \phi$, if ϕ is satisfied in every $\mathcal{I} \in Mod(\mathcal{O})$.

Given an ABox \mathcal{A} , we denote by $\mathcal{I}_{\mathcal{A}}$ the *DL-Lite_A* interpretation such that, for every concept instance assertion $C(a)$, $a^{\mathcal{I}} \in C^{\mathcal{I}}$ iff $C(a) \in \mathcal{A}$, for every role instance assertion $R(a, b)$, $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle^{\mathcal{I}} \in R^{\mathcal{I}}$ iff $R(a, b) \in \mathcal{A}$, and for every attribute instance assertion $U(a, b)$, $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle^{\mathcal{I}} \in U^{\mathcal{I}}$ iff $U(a, b) \in \mathcal{A}$.

We now recall queries, in particular conjunctive queries and unions of conjunctive queries. A conjunctive query (CQ) q is an expression of the form

$$q(\mathbf{x}) \leftarrow \alpha_1, \dots, \alpha_n$$

where \mathbf{x} is a tuple of variables, and every α_i is an atom whose predicate is a concept name or a role name or an attribute name, and whose arguments are either variables or constants, such that every variable occurring in \mathbf{x} also occurs in at least one α_i . The variables occurring in \mathbf{x} are called the *distinguished variables* of q , while the variables occurring in some α_i but not in \mathbf{x} are called the *existential variables* of q . The predicate q is called the *predicate* of the query, and the number of elements of \mathbf{x} is called the *arity* of q . A CQ is a *Boolean CQ* if it has no distinguished variables.

A union of conjunctive queries (UCQ) Q is a set of conjunctive queries of the same arity and having the same query predicate. A UCQ Q is a *Boolean UCQ* if every CQ belonging to Q is Boolean.

Given a CQ q of arity n , we denote by $q(\mathbf{c})$ the Boolean CQ obtained from q by replacing the distinguished variables in \mathbf{x} with the constants in the n -tuple of constants \mathbf{c} . As usual, given a *DL-Lite_A* interpretation \mathcal{I} , and a Boolean CQ $q \leftarrow \alpha_1, \dots, \alpha_n$, where \mathbf{y} are the existential variables occurring in q , we say that \mathcal{I} satisfies q if there exists an assignment μ for the variables \mathbf{y} such that every atom α_i is satisfied by \mathcal{I}, μ . Given a Boolean UCQ Q , we say that \mathcal{I} satisfies Q if \mathcal{I} satisfies at least one CQ in Q . Given a CQ q of arity n , the evaluation of q in \mathcal{I} , denoted by $eval(q, \mathcal{I})$, is the set of n tuples of constants \mathbf{c} such that \mathcal{I} satisfies $q(\mathbf{c})$. The evaluation of a UCQ Q in \mathcal{I} , denoted by $eval(Q, \mathcal{I})$, is the set $\bigcup_{q \in Q} eval(q, \mathcal{I})$.

The set of *certain answers* to a UCQ Q over a *DL-Lite_A* ontology $\langle \mathcal{T}, \mathcal{A} \rangle$, denoted by $cert(Q, \langle \mathcal{T}, \mathcal{A} \rangle)$, is the set of tuples $\bigcap_{\mathcal{I} \in Mod(\langle \mathcal{T}, \mathcal{A} \rangle)} eval(Q, \mathcal{I})$.

Given a UCQ Q and a TBox \mathcal{T} , a UCQ Q' is a *perfect rewriting* of Q with respect to \mathcal{T} if, for every ABox \mathcal{A} such that $\langle \mathcal{T}, \mathcal{A} \rangle$ is consistent, $cert(Q, \langle \mathcal{T}, \mathcal{A} \rangle) = eval(Q', \mathcal{I}_{\mathcal{A}})$. The above notion of perfect rewriting immediately extends to any query language for which the evaluation $eval$ of queries on a first-order interpretation is defined. We remark that many algorithms are available to compute perfect rewritings in *DL-Lite* logics (e.g., [4,10,13,9,6,5]).

In the following, for ease of exposition, we will not consider attributes in *DL-Lite_A* ontologies. However, all the algorithms and results that we present in this paper can be

immediately extended to handle attributes (since attributes can essentially be treated in a way analogous to roles).

3 Extensional Constraints

We now define the notion of EBox, which constitutes a set of extensional constraints, i.e., constraints over the ABox. The idea of EBox has been originally introduced in [11], under the name of *ABox dependencies*.

The following definitions are valid for every DL, under the assumption that the assertions are divided into extensional assertions and intensional assertions, and extensional assertions correspond to atomic instance assertions.

Given a set of intensional assertions \mathcal{N} and an interpretation \mathcal{I} , we say that \mathcal{I} *satisfies* \mathcal{N} if \mathcal{I} satisfies every assertion in \mathcal{N} .

An *extensional constraint box*, or simply *EBox*, is a set of intensional assertions. Notice that, from the syntactic viewpoint, an EBox is identical to a TBox. Therefore, entailment of an assertion ϕ with respect to an EBox \mathcal{E} (denoted by $\mathcal{E} \models \phi$) is defined exactly in the same way as in the case of TBoxes.

Given an ABox \mathcal{A} and an EBox \mathcal{E} , we say that \mathcal{A} *is valid for* \mathcal{E} if $\mathcal{I}_{\mathcal{A}}$ satisfies \mathcal{E} .

Definition 1. (Admissible ABox) *Given a TBox \mathcal{T} and an EBox \mathcal{E} , an ABox \mathcal{A} is an admissible ABox for \mathcal{T} and \mathcal{E} if \mathcal{A} is consistent with \mathcal{T} and \mathcal{A} is valid for \mathcal{E} . We denote with $ADM(\mathcal{T}, \mathcal{E})$ the set of ABoxes \mathcal{A} that are admissible for \mathcal{T} and \mathcal{E} .*

Informally, an EBox acts as a set of *integrity constraints* over the ABox. Differently from other recent approaches that have proposed various forms of integrity constraints for DL ontologies (e.g., [8,14]), an EBox constrains the ABox while totally discarding the TBox, since the notion of validity with respect to an EBox only considers the ABox.

We are now ready to define the notion of perfect rewriting in the presence of both a TBox and an EBox.

Definition 2. (Perfect rewriting in the presence of an EBox) *Given a TBox \mathcal{T} , an EBox \mathcal{E} , and a UCQ Q , a FOL query ϕ is a perfect rewriting of Q with respect to $\langle \mathcal{T}, \mathcal{E} \rangle$ if, for every ABox $\mathcal{A} \in ADM(\mathcal{T}, \mathcal{E})$, $\langle \mathcal{T}, \mathcal{A} \rangle \models Q$ iff $\mathcal{I}_{\mathcal{A}} \models \phi$.*

The above definition establishes a natural notion of perfect rewriting in the presence of an EBox \mathcal{E} . Since \mathcal{E} constrains the admissible ABoxes, the more selective is \mathcal{E} (for the same TBox \mathcal{T}), the more restricted the set $ADM(\mathcal{T}, \mathcal{E})$ is. If for instance, $\mathcal{E}, \mathcal{E}'$ are two EBoxes such that $\mathcal{E} \subset \mathcal{E}'$, we immediately get from the above definitions that $ADM(\mathcal{T}, \mathcal{E}) \supseteq ADM(\mathcal{T}, \mathcal{E}')$. Now, let Q be a UCQ, let ϕ be a perfect rewriting of Q with respect to $\langle \mathcal{T}, \mathcal{E} \rangle$ and let ϕ' be a perfect rewriting of Q with respect to $\langle \mathcal{T}, \mathcal{E}' \rangle$: ϕ will have to satisfy the condition $\langle \mathcal{T}, \mathcal{A} \rangle \models Q$ iff $\mathcal{I}_{\mathcal{A}} \models \phi$ for more ABoxes \mathcal{A} than query ϕ' . Consequently, ϕ will have to be a more complex query than ϕ' . Therefore, larger EBoxes in principle allow for obtaining simpler perfect rewritings.

4 Extensional Constraints and Query Rewriting

As already explained, the goal of this paper is to use extensional constraints to optimize query rewriting in *DL-Lite_A*. An intuitive explanation of how extensional constraints allow for simplifying query rewriting can be given by the following very simple example.

Suppose we are given a TBox $\{Student \sqsubseteq Person\}$, an empty EBox \mathcal{E}_0 , and an EBox $\mathcal{E}_1 = \{Student \sqsubseteq Person\}$. Now, given a query $q(x) \leftarrow Person(x)$, a perfect rewriting of this query with respect to $\langle \mathcal{T}, \mathcal{E}_0 \rangle$ is

$$\begin{aligned} q(x) &\leftarrow Person(x) \\ q(x) &\leftarrow Student(x) \end{aligned}$$

while a perfect rewriting of query q with respect to $\langle \mathcal{T}, \mathcal{E}_1 \rangle$ is the query q itself. Namely, under the EBox \mathcal{E}_1 we can ignore the TBox concept inclusion $Student \sqsubseteq Person$, since it is already satisfied by the ABox.

However, as already explained in [11], we can not always ignore TBox assertions that also appear in the EBox (and are thus already satisfied by the ABox). For instance, let q be the query $q \leftarrow C(x)$. If the TBox \mathcal{T} contains the assertions $\exists R \sqsubseteq C$ and $D \sqsubseteq \exists R^-$ and the EBox \mathcal{E} contains the assertion $\exists R \sqsubseteq C$, we cannot ignore this last inclusion when computing a perfect rewriting of q (or when answering query q). In fact, suppose the ABox is $\{D(a)\}$: then $\mathcal{A} \in ADM(\mathcal{T}, \mathcal{E})$ and query q is entailed by $\langle \mathcal{T}, \mathcal{A} \rangle$. But actually q is not entailed by $\langle \mathcal{T}', \mathcal{A} \rangle$ where $\mathcal{T}' = \mathcal{T} - \mathcal{E}$.

From the query rewriting viewpoint, a perfect rewriting of q with respect to \mathcal{T} is

$$\begin{aligned} q &\leftarrow C(x) \\ q &\leftarrow R(x, y) \\ q &\leftarrow D(y) \end{aligned}$$

while a perfect rewriting of q with respect to \mathcal{T}' is

$$q \leftarrow C(x)$$

And of course, the ABox \mathcal{A} shows that this last query is not a perfect rewriting of q with respect to $\langle \mathcal{T}, \mathcal{E} \rangle$. Therefore, also when computing a perfect rewriting, we cannot simply ignore the inclusions of the TBox that are already satisfied by the ABox (i.e., that belong to the EBox).

The example above shows that we need to understand under which conditions we are allowed to use extensional constraints to optimize query rewriting.

5 Prexto

In this section we present the algorithm **Prexto** (Perfect Rewriting under EXTensional cOnstraints). **Prexto** makes use of the algorithm **Presto**, originally defined in [13], which computes a nonrecursive datalog program constituting a perfect rewriting of a UCQ Q with respect to a *DL-Lite_A* TBox \mathcal{T} . The algorithm **Presto** is reported in Figure 1. We refer the reader to [13] for a detailed explanation of the algorithm. For our purposes, it suffices to remind that the program returned by **Presto** uses auxiliary datalog predicates, called ontology-annotated (OA) predicates, to represent every basic concept and basic role that is involved in the query rewriting. E.g., the basic concept


```

Algorithm Presto( $Q, \mathcal{T}$ )
Input: UCQ  $Q$ ,  $DL\text{-Lite}_R$  TBox  $\mathcal{T}$ 
Output: nr-datalog query  $Q'$ 
begin
   $Q' = \text{Rename}(Q)$ ;
   $Q' = \text{DeleteUnboundVars}(Q')$ ;
   $Q' = \text{DeleteRedundantAtoms}(Q', \mathcal{T})$ ;
   $Q' = \text{Split}(Q')$ ;
  repeat
    if there exist  $r \in Q'$  and ej-var  $x$  in  $r$ 
      such that  $\text{Eliminable}(x, r, \mathcal{T}) = \text{true}$ 
      and  $x$  has not already been eliminated from  $r$ 
    then begin
       $Q'' = \text{EliminateEJVar}(r, x, \mathcal{T})$ ;
       $Q'' = \text{DeleteUnboundVars}(Q'')$ ;
       $Q'' = \text{DeleteRedundantAtoms}(Q'', \mathcal{T})$ ;
       $Q' = Q' \cup \text{Split}(Q'')$ 
    end
  until  $Q'$  has reached a fixpoint;
  for each OA-predicate  $p_\alpha^n$  occurring in  $Q'$ 
  do  $Q' = Q' \cup \text{DefineAtomView}(p_\alpha^n, \mathcal{T})$ 
end

```

Fig. 1. The original Presto algorithm [13]

B is represented by the OA-predicate p_B^1 , while the basic role R is represented by the OA-predicate p_R^2 (the superscript represents the arity of the predicate) [1].

In the following, we modify the algorithm Presto. In particular, we make the following changes:

1. the final **for each** cycle of the algorithm (cf. Figure 1) is not executed: i.e., the rules defining the OA-predicates are not added to the returned program;
2. the algorithm **DeleteRedundantAtoms** is modified to take into account the presence of disjointness assertions and role functionality assertions in the TBox. More precisely, the following simplification rules are added to algorithm **DeleteRedundantAtoms**(Q', \mathcal{T}) (in which we denote basic concepts by B, C , basic roles by R, S , and datalog rules by the symbol r):
 - (a) if $p_R^2(t_1, t_2)$ and $p_S^2(t_1, t_2)$ occur in r and $\mathcal{T} \models R \sqsubseteq \neg S$, then eliminate r from Q' ;
 - (b) if $p_R^2(t_1, t_2)$ and $p_S^2(t_2, t_1)$ occur in r and $\mathcal{T} \models R \sqsubseteq \neg S^-$, then eliminate r from Q' ;
 - (c) if $p_B^1(t)$ and $p_C^1(t)$ occur in r and $\mathcal{T} \models B \sqsubseteq \neg C$, then eliminate r from Q' ;
 - (d) if $p_R^2(t_1, t_2)$ and $p_C^1(t_1)$ occur in r and $\mathcal{T} \models \exists R \sqsubseteq \neg C$, then eliminate r from Q' ;

¹ Actually, to handle Boolean subqueries, also 0-ary OA-predicates (i.e., predicates with no arguments) are defined: we refer the reader to [13] for more details.

- (e) if $p_R^2(t_1, t_2)$ and $p_C^1(t_2)$ occur in r and $\mathcal{T} \models \exists R^- \sqsubseteq \neg C$, then eliminate r from Q' ;
- (f) if p_α^0 and p_β^0 occur in r and $\mathcal{T} \models \alpha^0 \sqsubseteq \neg\beta^0$, then eliminate r from Q' ;
- (g) if $p_B^1(t)$ and p_α^0 occur in r and $\mathcal{T} \models B^0 \sqsubseteq \neg\alpha^0$, then eliminate r from Q' ;
- (h) if $p_R^2(t_1, t_2)$ and p_α^0 occur in r and $\mathcal{T} \models R^0 \sqsubseteq \neg\alpha^0$, then eliminate r from Q' ;
- (i) if $p_R^2(t_1, t_2)$ and $p_R^2(t_1, t'_2)$ (with $t_2 \neq t'_2$) occur in r and $(\text{funct } R) \in \mathcal{T}$, then, if t_2 and t'_2 are two different constants, then eliminate r from Q' ; otherwise, replace r with the rule $\sigma(r)$, where σ is the substitution which poses t_2 equal to t'_2 ;
- (j) if $p_R^2(t_2, t_1)$ and $p_R^2(t'_2, t_1)$ (with $t_2 \neq t'_2$) occur in r and $(\text{funct } R^-) \in \mathcal{T}$, then, if t_2 and t'_2 are two different constants, then eliminate r from Q' ; otherwise, replace r with the rule $\sigma(r)$, where σ is the substitution which poses t_2 equal to t'_2 .

Example 1. Let us show the effect of the new transformations added to `DeleteRedundantAtoms` through two examples. First, suppose $\mathcal{T} = \{B \sqsubseteq \neg B', (\text{funct } R)\}$ and suppose r is the rule

$$q(x) \leftarrow p_B^1(y), p_R^2(x, y), p_R^2(x, z), p_{B'}^1(z)$$

Then, the above case (i) of algorithm `DeleteRedundantAtoms` can be applied, which transforms r into the rule

$$q(x) \leftarrow p_B^1(y), p_R^2(x, y), p_{B'}^1(y)$$

Now, the above case (c) of algorithm `DeleteRedundantAtoms` can be applied, hence this rule is deleted from the program. Intuitively, this is due to the fact that this rule looks for elements belonging both to concept B and to concept B' , which is impossible because the disjointness assertion $B \sqsubseteq \neg B'$ is entailed by the TBox \mathcal{T} . Therefore, it is correct to delete the rule from the program. \square

From now on, when we speak about `Presto` we refer to the above modified version of the algorithm, and when we speak about `DeleteRedundantAtoms` we refer to the above modified version which takes into account disjointness and functionality assertions.

The `Prexto` algorithm is defined in Figure 2. The algorithm is constituted of the following four steps:

1. the nonrecursive datalog program P is computed by executing the `Presto` algorithm. This program P is not a perfect rewriting of Q yet, since the definition of the intermediate OA-predicates is missing;
2. the program P' is then constructed (by the three **for each** cycles of the program). This program contains rules defining the intermediate OA-predicates, i.e., the concept and role assertions used in the program P . To compute such rules, the algorithm makes use of the procedure `MinimizeViews`, reported in Figure 3. This procedure takes as input a basic concept (respectively, a basic role) B and computes a minimal subset Φ'' of the set Φ of the subsumed basic concepts (respectively, subsumed basic roles) of B which *extensionally cover* the set Φ , as explained below.

Algorithm $\text{Prexto}(Q, \mathcal{T}, \mathcal{E})$

Input: UCQ Q , $DL\text{-Lite}_A$ TBox \mathcal{T} , $DL\text{-Lite}_A$ EBox \mathcal{E}

Output: UCQ Q'

begin

$P = \text{Presto}(Q, \mathcal{T});$

$P' = \emptyset;$

for each OA-predicate P_R^2 occurring in P **do**

$\Phi = \text{MinimizeViews}(R, \mathcal{E}, \mathcal{T});$

$P' = P' \cup \{p_B^2(x, y) \leftarrow S(x, y) \mid S \text{ is a role name and } S \in \Phi\}$

$\cup \{p_B^2(x, y) \leftarrow S(y, x) \mid S \text{ is a role name and } S^- \in \Phi\};$

for each OA-predicate P_B^1 occurring in P **do**

$\Phi = \text{MinimizeViews}(B, \mathcal{E}, \mathcal{T});$

$P' = P' \cup \{p_B^1(x) \leftarrow C(x) \mid C \text{ is a concept name and } C \in \Phi\}$

$\cup \{p_B^1(x) \leftarrow R(x, y) \mid \exists R \in \Phi\} \cup \{p_B^1(x) \leftarrow R(y, x) \mid \exists R^- \in \Phi\};$

for each OA-predicate P_N^0 occurring in P **do**

$\Phi = \text{MinimizeViews}(N^0, \mathcal{E}, \mathcal{T});$

$P' = P' \cup \{p_N^0 \leftarrow C(x) \mid C \text{ is a concept name and } C^0 \in \Phi\}$

$\cup \{p_N^0 \leftarrow R(x, y) \mid R \text{ is a role name and } R^0 \in \Phi\};$

$P'' = P \cup P';$

$Q' = \text{Unfold}(P'');$

$Q' = \text{DeleteRedundantAtoms}(Q', \mathcal{E});$

return Q'

end

Fig. 2. The Prexto algorithm

3. then, the overall nonrecursive datalog program $P \cup P'$ is unfolded, i.e., turned into a UCQ Q' . This is realized by the algorithm Unfold which corresponds to the usual unfolding of a nonrecursive program;
4. finally, the UCQ Q' is simplified by executing the algorithm $\text{DeleteRedundantAtoms}$ which takes as input the UCQ Q' and the EBox \mathcal{E} (notice that, conversely, the first execution of $\text{DeleteRedundantAtoms}$ within the Presto algorithm uses the TBox \mathcal{T} as input).

Notice that the bottleneck of the whole process is the above step 3, since the number of conjunctive queries generated by the unfolding may be exponential with respect to the length of the initial query Q (in particular, it may be exponential with respect to the maximum number of atoms in a conjunctive query of Q). As shown by the following example, the usage of extensional constraints done at step 2 through the MinimizeViews algorithm is crucial to handle the combinatorial explosion of the unfolding.

Example 2. Let \mathcal{T} be the following $DL\text{-Lite}_A$ TBox:

$\text{Company} \sqsubseteq \exists \text{givesHighSalaryTo}^-$

$\exists \text{givesHighSalaryTo}^- \sqsubseteq \text{Manager}$

$\text{Manager} \sqsubseteq \text{Employee}$

$\text{Employee} \sqsubseteq \text{HasJob}$

$\exists \text{receivesGrantFrom} \sqsubseteq \text{StudentWithGrant}$

$\text{StudentWithGrant} \sqsubseteq \text{FulltimeStudent}$

$\text{FulltimeStudent} \sqsubseteq \text{Unemployed}$

$\text{FulltimeStudent} \sqsubseteq \text{Student}$

$\text{isBestFriendOf} \sqsubseteq \text{knows}$

(funct isBestFriendOf)

(funct isBestFriendOf^-)

$\text{HasJob} \sqsubseteq \neg \text{Unemployed}$

Algorithm `MinimizeViews`($B, \mathcal{E}, \mathcal{T}$)

Input: basic concept (or basic role, or 0-ary predicate) B ,

 $DL-Lite_{\mathcal{A}}$ EBox \mathcal{E} , $DL-Lite_{\mathcal{A}}$ TBox \mathcal{T}
Output: set of basic concepts (or basic roles, or 0-ary predicates) Φ''
begin
 $\Phi = \{B' \mid \mathcal{T} \models B' \sqsubseteq B\};$
 $\Phi' = \emptyset;$
for each $B' \in \Phi$ **do**

 if there exists $B'' \in \Phi$ such that $\mathcal{E} \models B' \sqsubseteq B''$ and $\mathcal{E} \not\models B'' \sqsubseteq B'$

 then $\Phi' = \Phi' \cup \{B'\};$
 $\Phi'' = \Phi - \Phi';$
while there exist $B, B' \in \Phi'$

 such that $B \neq B'$ and $\mathcal{E} \models B \sqsubseteq B'$ and $\mathcal{E} \models B' \sqsubseteq B$
do $\Phi'' = \Phi'' - \{B'\};$
return Φ''
end
Fig. 3. The `MinimizeViews` algorithm

 Moreover, let E_1, \dots, E_4 be the following concept inclusions:

$$E_1 = FulltimeStudent \sqsubseteq StudentWithGrant$$

$$E_2 = \exists receivesGrantFrom \sqsubseteq StudentWithGrant$$

$$E_3 = HasJob \sqsubseteq Employee$$

$$E_4 = Manager \sqsubseteq Employee$$

 and let $\mathcal{E} = \{E_1, E_2, E_3, E_4\}$.

 Finally, let q_1 be the following query:

$$q_1(x) \leftarrow Student(x), knows(x, y), HasJob(y)$$

 Let us first consider an empty EBox. In this case, during the execution of `Prexto`($q_1, \mathcal{T}, \emptyset$) the algorithm `MinimizeViews` simply computes the subsumed sets of *Student*, *knows*, *HasJob*, which are, respectively:

$$\text{MinimizeViews}(Student, \emptyset, \mathcal{T}) = \{Student, FulltimeStudent, StudentWithGrant, \exists receivesGrantFrom\}$$

$$\text{MinimizeViews}(knows, \emptyset, \mathcal{T}) = \{knows, knows^-, isBestFriendOf, isBestFriendOf^-\}$$

$$\text{MinimizeViews}(HasJob, \emptyset, \mathcal{T}) = \{HasJob, Employee, Manager, \exists givesHighSalaryTo^-\}$$

 Since every such set is constituted of four predicates, the UCQ returned by the unfolding step in `Prexto`($q_1, \mathcal{T}, \mathcal{E}$) contains 64 CQs. This is also the size of the final UCQ, since in this case no optimizations are computed by the algorithm `DeleteRedundantAtoms`, because both the disjointness assertion and the role functionality assertions of \mathcal{T} have no impact on the rewriting of query q_1 .

Conversely, let us consider the EBox \mathcal{E} : during the execution of $\text{Prexto}(q_1, \mathcal{T}, \mathcal{E})$, we obtain the following sets from the execution of the algorithm MinimizeViews :

$$\begin{aligned} \text{MinimizeViews}(\textit{Student}, \mathcal{E}, \mathcal{T}) &= \\ &\{\textit{Student}, \textit{StudentWithGrant}\} \\ \text{MinimizeViews}(\textit{knows}, \mathcal{E}, \mathcal{T}) &= \\ &\{\textit{knows}, \textit{knows}^-, \textit{isBestFriendOf}, \textit{isBestFriendOf}^-\} \\ \text{MinimizeViews}(\textit{HasJob}, \mathcal{E}, \mathcal{T}) &= \\ &\{\textit{Employee}, \exists \textit{givesHighSalaryTo}^-\} \end{aligned}$$

Thus, the algorithm MinimizeViews returns only two predicates for *Student* and only two predicates for *HasJob*. Therefore, the final unfolded UCQ is constituted of 16 CQs (since, as above explained, the final call to $\text{DeleteRedundantAtoms}$ does not produce any optimization). \square

We now focus on the proof of correctness of Prexto , which is based on the known results about the Presto algorithm. Indeed, to prove correctness of Prexto , essentially we have to show that the modifications done with respect to the Presto algorithm preserve correctness.

In particular, it is possible to prove the following properties:

1. The additional simplification rules added to the $\text{DeleteRedundantAtoms}$ algorithm preserve completeness of the algorithm. More specifically, it can be easily shown that, in every execution of the algorithm $\text{DeleteRedundantAtoms}$ within Presto , every additional rule transformation either produces a rule that is equivalent (with respect to the TBox \mathcal{T}) to the initial rule, or deletes a rule which is actually empty, i.e., which does not contribute to any nonempty conjunctive query in the final UCQ.
2. The optimization realized by the MinimizeViews algorithm is correct. More precisely, the following property can be easily shown:

Lemma 1. *Let \mathcal{T} be a TBox, let \mathcal{E} be an EBox, let B be a basic concept and let Φ be the set of basic concepts subsumed by B in \mathcal{T} and let Φ'' be the set returned by $\text{MinimizeViews}(B, \mathcal{E}, \mathcal{T})$. Then, the following property holds:*

$$\bigcup_{B \in \Phi} B^{\mathcal{I}_A} = \bigcup_{B \in \Phi''} B^{\mathcal{I}_A}$$

An analogous property can be shown when B is a basic role (or a 0-ary predicate). From the above lemma, it easily follows that step 2 of Prexto is correct.

3. In step 4 of Prexto , the final simplification of conjunctive queries realized by the execution of the algorithm $\text{DeleteRedundantAtoms}$ over the EBox \mathcal{E} is correct. This immediately follows from the correctness of $\text{DeleteRedundantAtoms}$ shown in the above point 1 and from the fact that the final UCQ is executed on the ABox, i.e., it is evaluated on the interpretation \mathcal{I}_A .

Therefore, from the above properties and the correctness of the original Presto algorithm, we are able to show the correctness of Prexto .

Theorem 1. *Let \mathcal{T} be a $DL\text{-Lite}_A$ TBox, let \mathcal{E} be a $DL\text{-Lite}_A$ EBox, let Q be a UCQ and let Q' be the UCQ returned by $\text{Prexto}(Q, \mathcal{T}, \mathcal{E})$. Then, for every ABox \mathcal{A} such that $\mathcal{A} \in \text{ADM}(\mathcal{T}, \mathcal{E})$, $\text{cert}(Q, \langle \mathcal{T}, \mathcal{A} \rangle) = \text{eval}(Q', \mathcal{I}_{\mathcal{A}})$.*

Finally, it is easy to verify the following property, which states that the computational cost of Prexto is no worse than all known query rewriting techniques for $DL\text{-Lite}_A$ which compute UCQs.

Theorem 2. *$\text{Prexto}(Q, \mathcal{T}, \mathcal{E})$ runs in polynomial time with respect to the size of $\mathcal{T} \cup \mathcal{E}$, and in exponential time with respect to the maximum number of atoms in a conjunctive query in the UCQ Q .*

6 Comparison

We now compare the optimizations introduced by Prexto with the current techniques for query rewriting in *DL-Lite*.

In particular, we consider the simple $DL\text{-Lite}_A$ ontology of Example 2 and compare the size of the UCQ rewritings generated by the current techniques (in particular, Presto and the rewriting based on the TBox minimization technique TBox-min shown in [11]) with the size of the UCQ generated by Prexto . To single out the impact of the different optimizations introduced by Presto , we present three different execution modalities for Prexto : without considering the EBox (we call this modality Prexto-noEBox); (ii) without considering disjointness axioms and role functionality axioms in the TBox (we call this modality Prexto-noDisj); (iii) and considering all axioms both in the TBox and in the EBox (we call this modality Prexto-full). Moreover, we will consider different EBoxes of increasing size, to better illustrate the impact of the EBox on the size of the rewriting.

Let \mathcal{T} be the $DL\text{-Lite}_A$ ontology of Example 2 and let $\mathcal{E}_1, \dots, \mathcal{E}_4$ be the following EBoxes:

$$\begin{aligned} \mathcal{E}_1 &= \{E_1\} \\ \mathcal{E}_2 &= \{E_1, E_2\} \\ \mathcal{E}_3 &= \{E_1, E_2, E_3\} \\ \mathcal{E}_4 &= \{E_1, E_2, E_3, E_4\} \end{aligned}$$

where E_1, \dots, E_4 are the concept inclusion assertions defined in Example 2. Finally, let q_0, q_1, q_2, q_3 be the following simple queries:

$$\begin{aligned} q_0(x) &\leftarrow \text{Student}(x) \\ q_1(x) &\leftarrow \text{Student}(x), \text{knows}(x, y), \text{HasJob}(y) \\ q_2(x) &\leftarrow \text{Student}(x), \text{knows}(x, y), \text{HasJob}(y), \text{knows}(x, z), \text{Unemployed}(z) \\ q_3(x) &\leftarrow \text{Student}(x), \text{knows}(x, y), \text{HasJob}(y), \text{knows}(x, z), \text{Unemployed}(z), \\ &\quad \text{knows}(x, w), \text{Student}(w) \end{aligned}$$

The table reported in Figure 4 shows the impact on rewriting (and answering) queries q_0, q_1, q_2 and q_3 of: (i) the disjointness axiom and the functional role axioms in \mathcal{T} ; (ii) the EBoxes $\mathcal{E}_1, \dots, \mathcal{E}_4$. In the table, we denote by Presto+unfolding the UCQ obtained by unfolding the nonrecursive datalog program returned by the Presto algorithm, and

query	algorithm	$\mathcal{E} = \emptyset$	$\mathcal{E} = \mathcal{E}_1$	$\mathcal{E} = \mathcal{E}_2$	$\mathcal{E} = \mathcal{E}_3$	$\mathcal{E} = \mathcal{E}_4$
q_0	Presto+unfolding	4	4	4	4	4
q_0	TBox-min	4	4	4	4	4
q_0	Prexto-noEBox	4	4	4	4	4
q_0	Prexto-noDisj	4	3	2	2	2
q_0	Prexto-full	4	3	2	2	2
q_1	Presto+unfolding	64	64	64	64	64
q_1	TBox-min	64	64	64	64	64
q_1	Prexto-noEBox	64	64	64	64	64
q_1	Prexto-noDisj	64	48	32	24	16
q_1	Prexto-full	64	48	32	24	16
q_2	Presto+unfolding	1024	1024	1024	1024	1024
q_2	TBox-min	1024	1024	1024	1024	1024
q_2	Prexto-noEBox	896	896	896	896	896
q_2	Prexto-noDisj	1024	576	256	192	128
q_2	Prexto-full	896	504	224	168	112
q_3	Presto+unfolding	16384	16384	16384	16384	16384
q_3	TBox-min	16384	16384	16384	16384	16384
q_3	Prexto-noEBox	12672	12672	12672	12672	12672
q_3	Prexto-noDisj	16384	6912	2048	1536	1024
q_3	Prexto-full	12672	5660	1504	1128	752

Fig. 4. Comparison of query rewriting techniques on \mathcal{T} , \mathcal{E} and queries q_0, q_1, q_2, q_3

denote by TBox-min the execution of Presto+unfolding which takes as input the TBox minimized by the technique presented in [11] using the extensional inclusions in the EBox. These two rows can be considered as representative of the state of the art in query rewriting in *DL-Lite* (with and without extensional constraints): indeed, due to the simple structure of the TBox and the queries, every existing UCQ query rewriting technique for plain *DL-Lite* ontologies (i.e., ontologies without EBoxes) would generate UCQs of size analogous to Presto+unfolding (of course, we are not considering the approaches where the ABox is preprocessed, in which of course much more compact query rewritings can be defined [71]).

The third column of the table displays the results when the empty EBox was considered, while the fourth, fifth, sixth, and seventh column respectively report the results when the EBox E_1, E_2, E_3, E_4 , was considered. The numbers in these columns represent the size of the UCQ generated when rewriting the query with respect to the TBox \mathcal{T} and the EBox \mathcal{E} : more precisely, this number is the number of CQs which constitute the generated UCQ. We refer to Example 2, for an explanation of the results obtained in the case of query q_1 .

The results of Figure 4 clearly show that even a very small number of EBox axioms may have a dramatic impact on the size of the rewritten UCQ, and that this is already the case for relatively short queries (like query q_2): this behavior is even more apparent for longer queries like q_3 . In particular, notice that, even when only two extensional inclusions are considered (case $\mathcal{E} = \mathcal{E}_2$), the minimization of the UCQ is already very significant. Moreover, for the queries under examination, extensional inclusions are more

effective than disjointness axioms and role functionality axioms on the minimization of the rewriting size.

The results also show that the technique presented in [11] for exploiting extensional inclusions does not produce any effect in this case. This is due to the fact that the extensional inclusions considered in our experiment do not produce any minimization of the TBox according to the condition expressed in [11]. Conversely, the technique for exploiting extensional constraints of **Prexto** is very effective. For instance, notice that this technique is able to use extensional constraints (like E_2 and E_3) which have no counterpart in the TBox, in the sense that such concept inclusions are not entailed by the TBox \mathcal{T} .

Finally, we remark that the above simple example shows a situation which is actually not favourable for the algorithm, since there are very few extensional constraints and short (or even very short) queries: nevertheless, the experimental results show that, even in this setting, our algorithm is able to produce very significant optimizations. Indeed, the ideas which led to the **Prexto** algorithm came out of a large OBDA project that our research group is currently developing with an Italian Ministry. In this project, several relevant user queries could not be executed by our ontology reasoner (Quonto [3]) due to the very large size of the rewritings produced. For such queries, the minimization of the rewriting produced by the usage of the **Prexto** optimizations is actually much more dramatic than the examples reported in the paper, because the queries are more complex (at least ten atoms) and the number of extensional constraints is larger than in the example. As a consequence, **Prexto** was able to lower the number of conjunctive queries generated, and thus the total query evaluation time, typically by two to three orders of magnitude: e.g., for one query, the total evaluation time passed from more than 11 hours to 42 seconds; other seven queries, whose rewritings could not even be computed or executed because of memory overflow of either the query rewriter or the DBMS query parser, could be executed in few minutes, or even in a few seconds, after the optimization.

7 Conclusions

In this paper we have presented a query rewriting technique for fully exploiting the presence of extensional constraints in a *DL-Lite_A* ontology. Our technique clearly proves that extensional constraints may produce a dramatic improvement of query rewriting, and consequently of query answering over *DL-Lite_A* ontologies.

We remark that it is immediate to extend **Prexto** to OWL2 QL: the main features of OWL2 QL that are not covered by *DL-Lite_A* mainly consist of the presence of additional role assertions (symmetric/asymmetric/reflexive/irreflexive role assertions). These aspects can be easily dealt with by **Prexto** through a simple extension of the algorithm.

We believe that the present approach can be extended in several directions. First, it would be extremely interesting to generalize the **Prexto** technique to ontology-based data access (OBDA), where the ABox is only virtually specified through declarative mappings over external data sources: as already mentioned in the introduction, in this scenario extensional constraints would be a very natural notion, since they could be automatically derived from the mapping specification. Then, it would be very interesting to extend the usage of extensional constraints beyond *DL-Lite_A* ontologies: in this

respect, a central question is whether existing query rewriting techniques for other description logics (e.g., [9][12]) can be extended with optimizations analogous to the ones of *Prexto*. Finally, we plan to fully implement our algorithm within the *Quonto/Mastro* system [3] for *DL-Lite_A* ontology management.

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. OWL 2 web ontology language profiles (2009), <http://www.w3.org/TR/owl-profiles/>
2. Artale, A., Calvanese, D., Kontchakov, R., Zakharyashev, M.: The *DL-Lite* family and relations. *J. of Artificial Intelligence Research* 36, 1–69 (2009)
3. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Poggi, A., Rodriguez-Muro, M., Rosati, R., Ruzzi, M., Savo, D.F.: The *Mastro* system for ontology-based data access. *Semantic Web J.* 2(1), 43–53 (2011)
4. 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)
5. Gottlob, G., Schwentick, T.: Rewriting ontological queries into small nonrecursive datalog programs. In: Proc. of the 24th Int. Workshop on Description Logic, DL 2011 (2011)
6. Kikot, S., Kontchakov, R., Zakharyashev, M.: On (in)tractability of OBDA with OWL2QL. In: Proc. of the 24th Int. Workshop on Description Logic, DL 2011 (2011)
7. Kontchakov, R., Lutz, C., Toman, D., Wolter, F., Zakharyashev, M.: The combined approach to query answering in *DL-Lite*. In: Proc. of the 12th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2010), pp. 247–257 (2010)
8. Motik, B., Horrocks, I., Sattler, U.: Bridging the gap between OWL and relational databases. *J. of Web Semantics* 7(2), 74–89 (2009)
9. Pérez-Urbina, H., Motik, B., Horrocks, I.: Tractable query answering and rewriting under description logic constraints. *J. of Applied Logic* 8(2), 186–209 (2010)
10. 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)
11. Rodriguez-Muro, M., Calvanese, D.: Dependencies: Making ontology based data access work in practice. In: Proc. of the 5th Alberto Mendelzon Int. Workshop on Foundations of Data Management, AMW 2011 (2011)
12. Rosati, R.: On conjunctive query answering in \mathcal{EL} . In: Proc. of the 20th Int. Workshop on Description Logic (DL 2007). *CEUR Electronic Workshop Proceedings*, vol. 250, pp. 451–458 (2007), <http://ceur-ws.org/>
13. Rosati, R., Almatelli, A.: Improving query answering over *DL-Lite* ontologies. In: Proc. of the 12th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2010), pp. 290–300 (2010)
14. Tao, J., Sirin, E., Bao, J., McGuinness, D.L.: Integrity constraints in OWL. In: Proc. of the 24th AAAI Conf. on Artificial Intelligence, AAAI 2010 (2010)
15. Thomas, E., Pan, J.Z., Ren, Y.: TrOWL: Tractable OWL 2 Reasoning Infrastructure. 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. 431–435. Springer, Heidelberg (2010)

Semi-automatically Mapping Structured Sources into the Semantic Web*

Craig A. Knoblock¹, Pedro Szekely¹, José Luis Ambite¹,
Aman Goel¹, Shubham Gupta¹, Kristina Lerman¹, Maria Muslea¹,
Mohsen Taheriyani¹, and Parag Mallick²

¹ University of Southern California

Information Sciences Institute and Department of Computer Science

{knoblock,pszekely,ambite,amangoel,shubhamg,lerman,mariam,mohsen}@isi.edu

² Stanford University

Department of Radiology

paragm@stanford.edu

Abstract. Linked data continues to grow at a rapid rate, but a limitation of a lot of the data that is being published is the lack of a semantic description. There are tools, such as D2R, that allow a user to quickly convert a database into RDF, but these tools do not provide a way to easily map the data into an existing ontology. This paper presents a semi-automatic approach to map structured sources to ontologies in order to build semantic descriptions (source models). Since the precise mapping is sometimes ambiguous, we also provide a graphical user interface that allows a user to interactively refine the models. The resulting source models can then be used to convert data into RDF with respect to a given ontology or to define a SPARQL end point that can be queried with respect to an ontology. We evaluated the overall approach on a variety of sources and show that it can be used to quickly build source models with minimal user interaction.

1 Introduction

The set of sources in the Linked Data cloud continues to grow rapidly. Many of these sources are published directly from existing databases using tools such as D2R [8], which makes it easy to convert relational databases into RDF. This conversion process uses the structure of the data as it is organized in the database, which may not be the most useful structure of the information in RDF. But either way, there is often no explicit semantic description of the contents of a source and it requires a significant effort if one wants to do more than simply

* This research is based upon work supported in part by the Intelligence Advanced Research Projects Activity (IARPA) via Air Force Research Laboratory (AFRL) contract number FA8650-10-C-7058. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, AFRL, or the U.S. Government.

convert a database into RDF. The result of the ease with which one can publish data into the Linked Data cloud is that there is lots of data published in RDF and remarkably little in the way of semantic descriptions of much of this data.

In this paper, we present an approach to semi-automatically building source models that define the contents of a data source in terms of a given ontology. The idea behind our approach is to bring the semantics into the conversion process so that the process of converting a data source produces a source model. This model can then be used to generate RDF triples that are linked to an ontology and to provide a SPARQL end point that converts the data on the fly into RDF with respect to a given ontology. Users can define their own ontology or bring in an existing ontology that may already have been used to describe other related data sources. The advantage of this approach is that it allows the source to be transformed in the process of creating the RDF triples, which makes it possible to generate RDF triples with respect to a specific domain ontology.

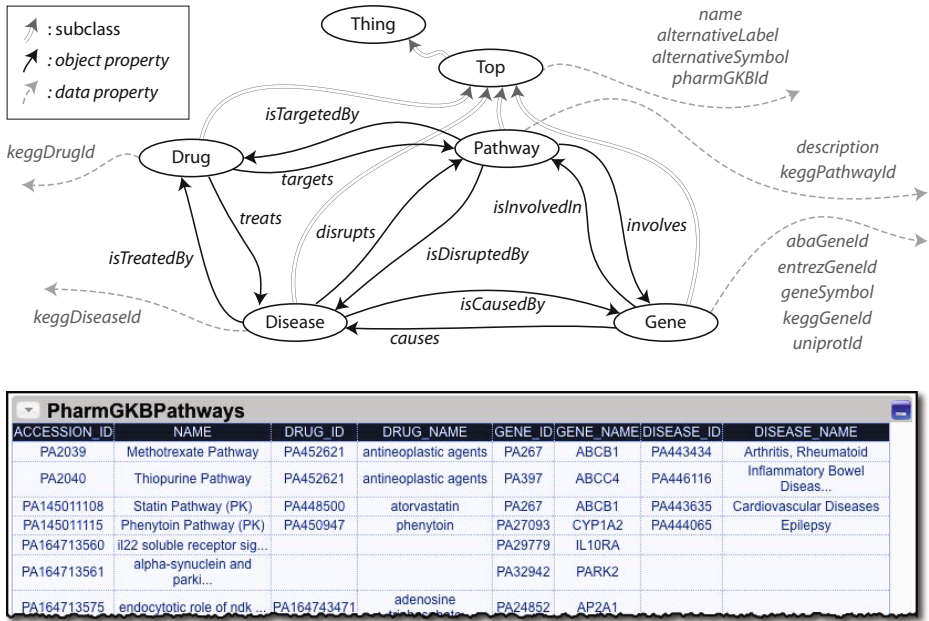
The conversion to RDF is a critical step in publishing sources into the Linked Data cloud and this work makes it possible to convert sources into RDF with the underlying semantics made explicit. There are other systems, such as R2R [7] and W3C's R2RML [9], that define languages for specifying mappings between sources, but none of this work provides support for defining these mappings. This paper describes work that is part of our larger effort on developing techniques for performing data-integration tasks by example [23]. The integrated system is available as an open-source tool called Karma¹.

2 Motivating Example

The bioinformatics community has produced a growing collection of databases with vast amounts of data about diseases, drugs, proteins, genes, etc. Nomenclatures and terminologies proliferate and significant efforts have been undertaken to integrate these sources. One example is the Semantic MediaWiki Linked Data Extension (SMW-LDE) [5], designed to support unified querying, navigation, and visualization through a large collection of neurogenomics-relevant data sources. This effort focused on integrating information from the Allen Brain Atlas (ABA) with standard neuroscience data sources. Their goal was to “bring ABA, Uniprot, KEGG Pathway, PharmGKB and Linking Open Drug Data [16] data sets together in order to solve the challenge of finding drugs that target elements within a disease pathway, but are not yet used to treat the disease.”

We use the same scenario to illustrate and evaluate our contributions, comparing our results to the published SMW-LDE results (see Figure 1). We use logical rules to formally define the mapping between data sources and an ontology. Specifically, we use global-local-as-view (GLAV) rules [13] commonly used in data integration [15] and data exchange [3] (i.e., rules whose antecedent and consequent are conjunctive formulas). The rule antecedent is the source relation that defines the columns in the data source. The rule consequent specifies how the source data elements are defined using the ontology terms. For example, the

¹ <https://github.com/InformationIntegrationGroup/Web-Karma-Public>



PharmGKBPathways($ACCESSION_ID$, $NAME$, $DRUG_ID$, $DRUG_NAME$, $GENE_ID$, $GENE_NAME$, $DISEASE_ID$, $DISEASE_NAME$) \rightarrow
 $Pathway(uri(ACCESSION_ID)) \wedge name(uri(PATHWAY_ID), NAME) \wedge$
 $involves(uri(PATHWAY_ID), uri(GENE_ID)) \wedge$
 $isTargetedBy(uri(PATHWAY_ID), uri(DRUG_ID)) \wedge$
 $isDisruptedBy(uri(PATHWAY_ID), uri(DISEASE_ID)) \wedge$
 $Gene(uri(GENE_ID)) \wedge geneSymbol(uri(GENE_ID), GENE_NAME) \wedge$
 $Drug(uri(DRUG_ID)) \wedge name(uri(DRUG_ID), DRUG_NAME) \wedge$
 $Disease(uri(DISEASE_ID)) \wedge name(uri(DISEASE_ID), DISEASE_NAME)$

Fig. 1. The ontology used in the SMW-LDE study, one of the KEGG Pathway sources used, and the source model that defines the mapping of this source to the ontology

first term, $Pathway(uri(ACCESSION_ID))$ specifies that the values in the $ACCESSION_ID$ column are mapped to the $Pathway$ class, and that these values should be used to construct the URIs when the source description is used to generate RDF. The second term, $name(uri(ACCESSION_ID), NAME)$ specifies that the values in the $ACCESSION_ID$ are related to the values in the $NAME$ column using the $name$ property.

The task in the SMW-LDE scenario is to define source models for 10 data sources. Writing these source models by hand, or the equivalent R2R rules is laborious and requires significant expertise. In the next sections we describe how our system can generate source models automatically and how it enables users to intervene to resolve ambiguities.

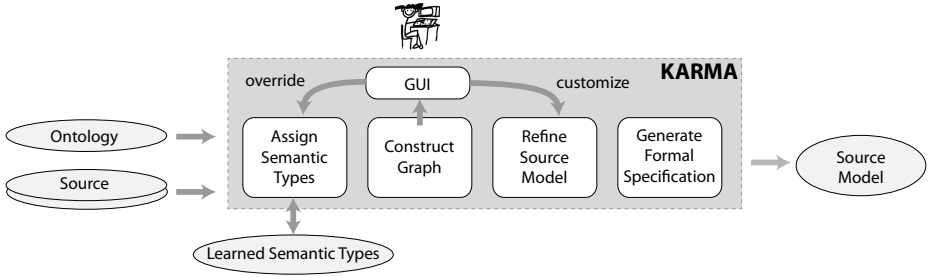


Fig. 2. The Karma process to model structured sources

3 Modeling Structured Sources

Figure 2 illustrates our approach for modeling data sources. The inputs to the process are an OWL ontology, the collection of data sources that the user wants to map to the ontology, and a database of semantic types that the system has learned to recognize based on prior use of the tool. The main output is the model that specifies, for each source, the mapping between the source and the ontology. A secondary output is a refined database of semantic types, updated during the process to incorporate semantic types learned using the data contained in the sources being mapped.

As shown in Figure 2, the modeling process consists of four main steps. The first step, *Assign Semantic Types*, involves mapping each column of a source to a node in the ontology. This is a user-guided process where the system assigns types automatically based on the data values in each column and a set of learned probabilistic models constructed from assignments done in prior sessions. If the semantic type assigned by the system is incorrect, the user can select from a menu the correct node in the graph. The system learns from this assignment and records the learned assignment in its database. The second step, *Construct Graph*, involves constructing a graph that defines the space of all possible mappings between the source and the ontology. At a high level, the nodes in the graph represent classes in the ontology, and the edges represent properties that relate these classes. The mapping from the ontology to the graph is not one-to-one given that, for example, several columns may contain instances of the same class (Section 3.2). The third step, *Refine Source Model*, updates the graph to refine the model based on user input. The graph is constructed so that the mapping between the source and the ontology can be computed using a Steiner tree algorithm (Section 3.3). The final fourth step, *Generate Formal Specification*, generates a formal specification of the source model from the Steiner tree computed in the prior step (Section 3.5). An example of this formal specification appears in the bottom part of Figure 1.

In general, it is not always possible to automatically compute the *desired* mapping between a source and an ontology since there may not be enough information in the source to determine the mapping. So, the automated process

computes the most succinct mapping, and the user interface allows the user to guide the process towards the desired interpretation (Section 3.4).

3.1 Inferring the Semantic Types

Semantic types characterize the type of data that appears in a column of data. For example, in the table shown in Figure 1, the first column contains PharmGKB identifiers of pathways, the second one contains names of pathways, etc. In some cases, semantic types correspond to classes in an OWL ontology, but in most cases, they could be most naturally thought of as the ranges of data properties. It is possible to define semantically meaningful RDFS types in OWL and use them as the ranges of data properties. However, few ontologies define such types. The ranges of data properties are almost always missing, or they are defined using syntactic types such as String or Integer.

In our modeling framework, a semantic type can be either an OWL class or a pair consisting of a data property and an OWL class (the property domain or a subclass of it). We use OWL classes to define the semantic types of columns of data that contain automatically-generated database keys or foreign keys (during RDF generation, these keys are used to generate URIs). We use semantic types defined in terms of data properties and their domain for columns containing meaningful data. In our example, the first column contains PharmGKB identifiers of pathways, so the values can be characterized by the semantic type consisting of the data property `pharmGKBId` and the class `Pathway`, or `Pathway.pharmGKBId` for short.

Karma provides a user interface to let users assign semantic types to the columns of a data source. In this section we present our approach for automating the assignment of semantic types by learning from prior assignments defined in the user interface. The objective is to learn a labeling function $\phi(n, \{v_1, v_2, \dots\}) = t$ so that given n , the name of a column, and $\{v_1, v_2, \dots\}$, the values in that column, it assigns a semantic type $t \in T$, where T is the set of semantic types used during training. The training data consists of a set of prior assignments of semantic types t_i to columns of data: $\{(n_1, \{v_{11}, v_{12}, \dots\}, t_1), (n_2, \{v_{21}, v_{22}, \dots\}, t_2), \dots\}$.

We use a conditional random field (CRF) [18] to learn the labeling function. Before giving the details of how we build the feature vectors to train the CRF, we first explain how we define ϕ in terms of a function $\hat{\phi}$ that we use to label individual values in a column of data. Given a column name n and a single value v in that column, $\hat{\phi}(n, v) = \{(v, t_k, p_k), t_k \in T\}$ gives for each t_k in T the probability p_k that the semantic type of v is t_k . To label a column of data $(n, \{v_1, v_2, \dots\})$, we compute $\hat{\phi}(n, v_i)$ for each value $v_i \in \{v_1, v_2, \dots\}$, and then compute the average probability over all values in a column. The result is a set of pairs $\bar{\phi}(n, \{v_1, v_2, \dots\}) = \{(t_1, p_1), (t_2, p_2), \dots\}$. Based on this set, we define $\phi(n, \{v_1, v_2, \dots\}) = t_m$, the type with maximum probability, i.e., t_m is such that $(t_m, p_m) \in \bar{\phi}(n, \{v_1, v_2, \dots\})$ and $p_m \geq p_i$ for all $(t_k, p_k) \in \bar{\phi}(n, \{v_1, v_2, \dots\})$. When users load a source, Karma automatically labels every column using $\phi(n, \{v_j\})$ as long as the probability p_m is above a certain threshold.

The task is now to learn the labeling function $\hat{\phi}(n, v)$. As mentioned above, users label columns of data, but to learn $\hat{\phi}(n, v)$ we need training data that assigns semantic types to each value in a column. We assume that columns contain homogeneous values, so from a single labeled column $(n, \{v_1, v_2, \dots\}, t)$ we generate a set of training examples $\{(n, v_1, t), (n, v_2, t), \dots\}$ as if each value in the column had been labeled using the same semantic type t .

For each triple (n, v, t) we compute a feature vector (f_i) that characterizes the syntactic structure of the column name n and the value v . To compute the feature vector, we first tokenize the name and the value. Our tokenizer uses white space and symbol characters to break strings into tokens, but identifies numbers as single tokens. For example, the name `ACCESSION_ID` produces the tokens (“Accession”, “_”, “Id”), the value `PA2039` produces the tokens (“PA”, “2039”), and the value `72.5°F` produces the tokens (72.5, °, F).

Each f_i is a Boolean feature function $f_i(n, v)$ that tests whether the name, value or the resulting tokens have a particular feature. For example, `valueStartsWithA`, `valueStartsWithB`, `valueStartsWithPA` are three different feature functions that test whether the value starts with the characters ‘A’, ‘B’ or the substring “PA”; `hasNumericTokenWithOrderOfMagnitude1`, `hasNumericTokenWithOrderOfMagnitude10` are feature functions that test whether the value contains numeric tokens of order of magnitude 1 and 10 respectively. In general, features are defined using templates of the form `predicate(X)`, and are instantiated for different values of X that occur within the training data. In our scenario, `valueStartsWith(X)` is instantiated with $X=‘P’$ and $X=‘A’$ because “PA2039” is in the first column and “Arthritis, Rheumatoid” is in the last column; however, there will be no `valueStartsWithB` feature because no value starts with the character ‘B’. Our system uses 21 predicates; the most commonly instantiated ones are:

`nameContainsToken(X)`, `nameStartsWith(X)`, `valueContainsToken(X)`, `valueStartsWith(X)`, `valueHasCapitalizedToken()`, `valueHasAllUppercaseToken()`, `valueHasAlphabeticalTokenOfLength(X)`, `valueHasNumericTokenWithOrderOfMagnitude(X)`, `valueHasNumericTokenWithPrecision(X)`, `valueHasNegativeNumericToken()`.

A CRF is a discriminative model, and it is practical to construct feature vectors with hundreds or even thousands of overlapping features. The model learns the weight for each feature based on how relevant it is in identifying the semantic types by optimizing a log-linear objective function that represents the joint likelihood of the training examples. A CRF model is useful for this problem because it can handle large numbers of features, learn from a small number of examples, and exploit the sequential nature of many structured formats, such as dates, temperatures, addresses, etc. To control execution times, our system labels and learns the labeling function using at most 100 randomly selected values from a column. With 100 items, labeling is instantaneous and learning takes up to 10 seconds for sources with over 50 semantic types.

3.2 Constructing the Graph

The central data structure to support the mapping of sources to the ontology is a graph computed from the semantic types of the source and the domain

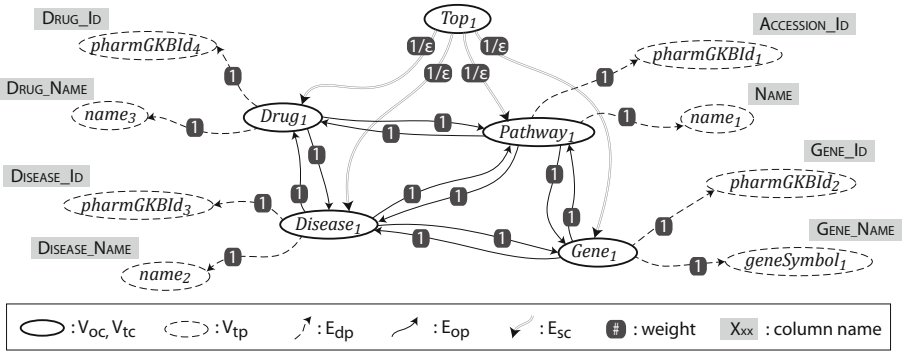


Fig. 3. The graph defines the search space for source models and provides the information for the user interface to enable users to refine the computed source model

ontology. The algorithm for building the graph has three sequential steps: graph initialization, computing nodes closure, and adding the links.

Graph Initialization: We start with an empty graph called G . In this step, for each semantic type assigned to a column, a new node with a unique label is added to the graph. A semantic type is either a class in the ontology or a pair consisting of the name of a datatype property and its domain. We call the corresponding nodes in the graph V_{tc} and V_{tp} respectively. Applying this step on the source shown in Figure 3 results in $V_{tc} = \{\}$ and $V_{tp} = \{pharmGKBId_1, pharmGKBId_2, pharmGKBId_3, pharmGKBId_4, name_1, name_2, name_3, geneSymbol_1\}$.

Computing Nodes Closure: In addition to the nodes that are mapped from semantic types, we have to find nodes in the ontology that relate those semantic types. We search the ontology graph and for every class node that has a path to the nodes corresponding to semantic types, we create a node in the graph. In other words, we get all the class nodes in the ontology from which the semantic types are reachable. To compute the paths, we consider both properties and *isa* relationships. The nodes added in this step are called V_{oc} . In the example, we would have $V_{oc} = \{Thing_1, Top_1, Gene_1, Pathway_1, Drug_1, Disease_1\}$. In Figure 3, solid ovals represent $\{V_{tc} \cup V_{oc}\}$, which are the nodes mapped from classes of ontology, and the dashed ovals represent V_{tp} , which are the semantic types corresponding to datatype properties.

Adding the Links: The final step in constructing the graph is adding the links to express the relationships among the nodes. We connect two nodes in the graph if there is a datatype property, object property, or *isa* relationship that connects their corresponding nodes in the ontology. More precisely, for each pair of nodes in the graph, u and v :

- If $v \in V_{tp}$, i.e., v is a semantic type mapped from a datatype property, and u corresponds to the domain class of that semantic type, we create a directed weighted link (u, v) with a weight equal to one ($w = 1$). For example, there

would be a link from $Pathway_1$ to $pharmGKBId_1$, because $pharmGKBId_1$ corresponds to the semantic type $Pathway.pharmGKBId$.

- If $u, v \in \{V_{tc} \cup V_{oc}\}$, which means both of them are mapped from ontology classes, we put a weighted link (u, v) with $w = 1$ in the graph only if there is an object property such as p in the ontology whose domain includes the class of u and whose range includes class of v . These links are called E_{op} . Note that the properties inherited from parents are also considered in this part, but to prioritize direct properties in the algorithm, we consider a slightly higher weight to the inherited properties. In other words, if p is defined such that its domain contains one of the superclasses of u (at any level) and its range contains one of the superclasses of v , we add the link (u, v) with $w = 1 + \epsilon$.

- If $u, v \in \{V_{tc} \cup V_{oc}\}$ and v is a direct or indirect subclass of u , a link (u, v) with $w = 1/\epsilon$ is added to the graph, in which ϵ is a very small value. We call these links E_{sc} . Subclass links have a large weight so that relationships mapped from properties are preferred over the relationships through the class hierarchy.

The final graph is a directed weighted graph $G = (V, E)$ in which $V = \{V_{tp} \cup V_{tc} \cup V_{oc}\}$ and $E = \{E_{dp} \cup E_{op} \cup E_{sc}\}$. Figure 3 shows the final graph.

3.3 Generating Source Models

Source models must explicitly represent the relationships between the columns of a source. For example, after mapping columns to the **Gene** and **Drug** classes, we want to explicitly represent the relationship between these two classes. The graph we constructed in the previous section explicitly represents all possible relationships among the semantic types. We construct a source model as the minimal tree that connects the semantic types. The minimal tree corresponds to the most succinct model that relates all the columns in a source, and this is a good starting point for refining the model. To compute the minimal tree, we use one of the variations of the known Steiner Tree algorithm. Given an edge-weighted graph and a subset of the vertices, called Steiner nodes, the goal is to find the minimum-weight tree in the graph that spans all Steiner nodes. In our graph, the Steiner nodes are the semantic type nodes, i.e., the set $\{V_{tc} \cup V_{tp}\}$. The Steiner tree problem is NP-complete, but we use a heuristic algorithm [17] with an approximation ratio bounded by $2(1 - 1/l)$, where l is the number of leaves in the optimal Steiner tree. The time complexity of the algorithm is $O(|V_{tc} \cup V_{tp}| |V|^2)$. Figure 4(a) shows the resulting Steiner tree.

It is possible that multiple minimal trees exist, or that the correct interpretation of the data is specified by a non-minimal tree. In these cases, Karma allows the user to interactively impose constraints on the algorithms that lead to the correct model. We enforce these constraints on G by transforming it into a new graph G' , and using G' as the input to the Steiner tree algorithm. User actions can have three types of effects on the algorithm:

Changing the Semantic Types: If the user changes the semantic type of one or more columns, we re-construct the graph G and repeat all the steps mentioned before to get the final Steiner tree.

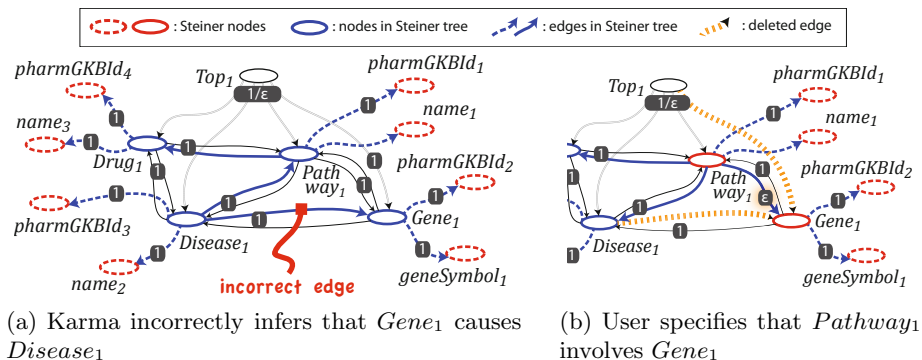


Fig. 4. Interactive refinement of the automatically computed Steiner trees

Specifying a Relationship: In the Steiner tree shown in Figure 4(a), *Disease* is related to *Gene* through the *isCausedBy* property. However, in the correct model of the data, *Gene* is related to *Pathway* through the *involves* property. Karma allows the user to correct the model and change the relationship from *isCausedBy* to *involves*. To force the Steiner tree algorithm to select the new link, we first add the source (*Pathway₁*) and target (*Gene₁*) of the link to the Steiner nodes. Then we remove all the incoming links to the target except the link selected by the user. This means that *involves* would be the only link in the graph going to *Gene₁*. Finally, we reduce the weight of the user link to ϵ . These steps guarantee that the user link will be chosen by the Steiner algorithm. Note that forcing a link by the user does not change graph G and it only affects G' and the Steiner nodes. Figure 4(b) illustrates the new G' and Steiner tree after selecting the *involves* relationship by the user.

Generating Multiple Instances of a Class: Consider the case that in the source table, in addition to information about the genes involved in pathway, we also have the data about genes that cause specific diseases. This means that, for example, we have two columns *GENE_NAME1* and *GENE_NAME2* referring to different genes. Suppose that the CRF model has assigned the *Gene.geneSymbol* semantic type to both columns and their corresponding nodes in the graph are *geneSymbol₁* and *geneSymbol₂*. After constructing the graph, we would have two outgoing links from *Gene₁* to *geneSymbol₁* and *geneSymbol₂*, indicating that *GENE_NAME1* and *GENE_NAME2* are different symbols of the same *Gene*. However, the correct model is the one in which *GENE_NAME1* and *GENE_NAME2* are symbols for two different genes. That is, there should be two instances of the *Gene* class, *Gene₁* and *Gene₂* that are separately connected to *geneSymbol₁* and *geneSymbol₂*. To solve this problem, Karma gives the option to the user to generate multiple instances of a class in the GUI. The user selects the *Gene₁* node and splits it based on the *geneSymbol* property. Then G' and the Steiner tree are re-computed to produce the correct model.

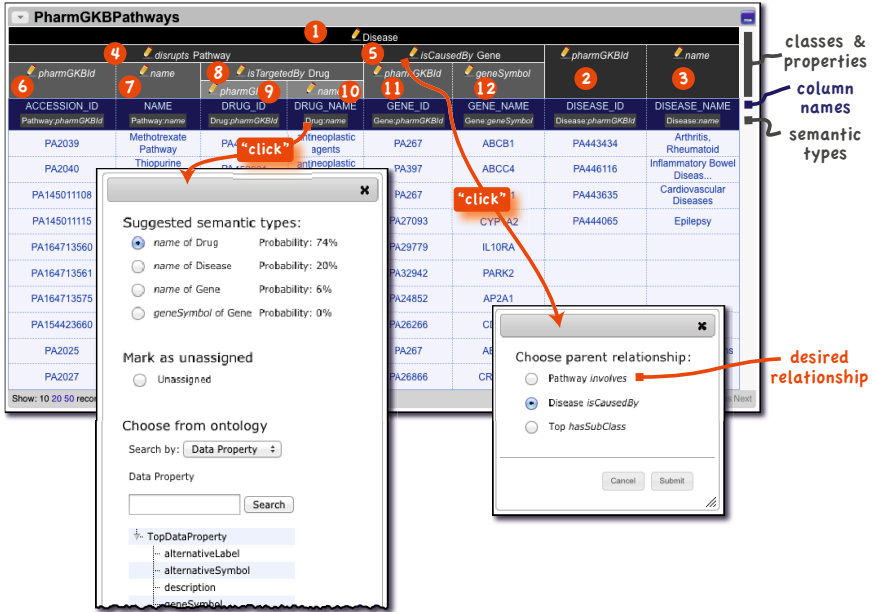


Fig. 5. Karma screen showing the PharmGKBPathways source. Clicking on the pencil icon brings up a menu where users can specify alternative relationships between classes. Clicking on a semantic type brings up a menu where the user can select the semantic types from the ontology. A movie showing the user interface in action is available at <http://isi.edu/integration/videos/karma-source-modeling.mp4>

3.4 User Interface for Refining Semantic Models

Karma visualizes a source model as a tree of nodes displayed above the column headings of a source. Figure 5 shows the visualization of the source model corresponding to the Steiner tree shown in Figure 4(a). The root of the Steiner tree appears at the top, and shows the name of the class of objects that the table is about (in our example the table is about diseases²). The Steiner nodes corresponding to the semantic types are shown just below the column headings. The nodes between the root and the semantic types show the relationships between the different objects represented in the table. Internal nodes of the Steiner tree (e.g., nodes 4, 5 and 8) consist of the name of an object property, shown in italics and a class name (a subclass of the range of the property). The property defines the relationship between the class named in the parent node and the class of the current node. For example, node 4 is “*disrupts Pathway*”, which means that the Disease (node 1) disrupts the Pathway represented by the columns under node 4. The leaves of the tree (nodes 6, 7, 9, etc.) show the name of data properties. For example, node 6 is *pharmGKBId*, meaning that the column contains the *pharmGKBId* of the Pathway in node 4.

² Selection of the root is not unique for ontologies that declare property inverses. In this example, any of the classes could have been selected as the root yielding equivalent models.

PharmGKBPathways						Disease	
pharmGKBId	name	isTargetedBy Drug		involves Gene		pharmGKBId	name
		pharmGKBId	name	pharmGKBId	geneSymbol		
ACCESSION_ID	NAME	DRUG_ID	DRUG_NAME	GENE_ID	GENE_NAME	DISEASE_ID	DISEASE_NAME
Pathway.pharmGKBId	Pathway.name	Drug.pharmGKBId	Drug.name	Gene.pharmGKBId	Gene.geneSymbol	Disease.pharmGKBId	Disease.name
PA2039	Methotrexate Pathway	PA452621	antineoplastic agents	PA267	ABCB1	PA443434	Arthritis, Rheumatoid

Fig. 6. Karma screen showing the user interaction to change the model of a column from a Pathway label to a Drug label

According to the model shown in Figure 5, the table contains information about diseases (1): the last column contains the disease names (3) and the next to last column contains their identifiers (2). The Disease *disrupts* a Pathway (4), and *isCausedBy* a Gene (5). The Pathway is identified using its *pharmGKBId* in the first column (6), and its *name* appears in the second column (7). The Pathway *isTargeted* by the Drug (8) whose identifier (9) and label (10) appear in the third and fourth columns. The gene that causes the disease (5) is identified using its *pharmGKBId* (11) and its *geneSymbol* (12).

This is a plausible model, but it is incorrect because the table lists the genes involved in the pathways that are disrupted by the disease instead of the genes that cause the disease; in other words, the *isCausedBy* property in cell 5 is incorrect. Users can edit the model to adjust the relationships between columns by clicking on the pencil icons. When the pop-up in Figure 5 appears, the user clicks on the pencil icon on the Gene cell (5): it shows the possible relationships corresponding to all incoming edges to the $Gene_1$ node in the graph shown in Figure 3. Figure 6 shows the adjusted model after the user selects the “Pathway Involves” option in Figure 5 to specify the correct relationship between the disease and the gene. The Gene cell (5) is now below Pathway (4) related using the *involves* property.

Karma also provides capabilities to clean, normalize and transform data before modeling it. For example, a source in our scenario contained alternative symbols for genes as comma-separated values stored in individual cells (e.g., “CP12, P3-450, P450(PA)”). Karma provides a “split cell” command to break the value into multiple cells so that each value can be modeled as a separate alternative symbol. These commands can be saved in scripts to enable automatic preprocessing of sources when source models are used to generate RDF.

3.5 Generation of Formal Source Model Specification

After users have (optionally) imposed constraints to reflect the correct semantics, the system processes the resulting Steiner tree to generate GLAV rules that provide a formal specification of (1) how the sources are combined and which attributes of the source are relevant, (2) how the source data maps to the ontology, and (3) how URIs for objects in the ontology are generated. We illustrate the algorithm that generates the GLAV rule of Figure 11 based on the Steiner tree from Figure 4(b), which corresponds to the user interface shown in Figure 6.

Class nodes generate unary predicates corresponding to classes in the ontology. The `uri` function builds URIs for class instances based on the key(s), or foreign key(s), in the source tables. For example, the `Pathway` node in Figures 3 and 6 generates the predicate `Pathway(uri(ACCESSION_ID))` because the values in the `ACCESSION_ID` column represent instances of `Pathway`.

The system also supports class nodes that are not associated with a source column. These correspond to existentially quantified variables in the rule consequent and would generate blank nodes in RDF. However, we generate unique regular URIs to support linking (`owl:sameAs`) into these URIs at a later stage. For example, assume that the ontology included a `Mutation` class, where a `Gene` has a `Mutation` that causes a `Disease`, then the corresponding fragment of the rule consequent would be: `hasMutation(uri(GENE_ID), uri(1)) ^ Mutation(uri(1)) ^ causes(uri(1), uri(DISEASE_ID))`. The index in the `uri` function is used to identify different existentially quantified variables.

Data property nodes generate binary predicates corresponding to data properties in the ontology. For example, the `name1` node associated with `Pathway` in Figure 3 generates the binary predicate `name(uri(ACCESSION_ID), NAME)`, specifying that instances of `Pathway` have the `name` data property filled with values from the `NAME` column.

Edges between class nodes generate binary predicates corresponding to object properties in the ontology. For example, the edge between `Pathway1` and `Gene1` in Figure 4(b) generates the predicate `involves(uri(ACCESSION_ID), uri(GENE_ID))`.

The resulting GLAV rules can now be used to generate the appropriate RDF for a source in terms of the domain ontology, as in data exchange 3. Alternatively, the mappings can be interpreted dynamically by a mediator, as in data integration 15. The mediator would provide a SPARQL endpoint exposing the ontology and executing queries directly over the original sources.

4 Evaluation

We evaluated our approach by generating source models for the same set of sources integrated by Becker et al. 5, as described in Section 2. The objective of the evaluation was 1) to assess the ability of our approach to produce source models equivalent to the mappings Becker et al. defined for these sources, and 2) to measure the effort required in our approach to create the source models. Becker et al. defined the mappings using R2R, so we used their R2R mapping files as a specification of how data was to be mapped to the ontology. Our objective was to replicate the effect of the 41 R2R mapping rules defined in these files. Each R2R mapping rule maps a column in our tabular representation. We measured effort in Karma by counting the number of user actions (number of menu choices to select correct semantic types or adjust paths in the graph) that the user had to perform. Effort measures for the R2R solution are not available, but appears to be substantial given that the rules are expressed in multiple pages of RDF.

Using Karma we constructed 10 source models that specify mappings equivalent to all of the 41 R2R mapping rules. Table 1 shows the number of actions

Table 1. Evaluation Results for Mapping the Data Sources using Karma

Source	Table Name	# Columns	# User Actions		
			Assign Semantic Type	Specify Relationship	Total
PharmGKB	Genes	8	8	0	8
	Drugs	3	3	0	3
	Diseases	4	4	0	4
	Pathways	5	2	1	3
ABA	Genes	6	3	0	3
KEGG Pathway	Drugs	2	2	0	2
	Diseases	2	2	0	2
	Genes	1	1	0	1
	Pathways	6	3	1	4
UniProt	Genes	4	1	0	1
		Total: 41	Total: 29	Total: 2	Total: 31
		Avg. # User Actions/Column = $31/41 = 0.76$			
Events database	19 Tables	Total: 64	Total: 43	Total: 4	Total: 47
		Avg. # User Actions/Column = $47/64 = 0.73$			

required to map all the data sources. The Assign Semantic Type column shows the number of times we had to manually assign a semantic type. We started this evaluation with no training data for the semantic type identification. Out of the 29 manual assignments, 24 were for specifying semantic types that the system had never seen before, and 5 to fix incorrectly inferred types.

The Specify Relationship column shows the number of times we had to select alternative relationships using a menu (see Figure 5). For the PharmGKB and KEGG Pathway sources, 1 action was required to produce a model semantically equivalent to the R2R mapping rule. The total number of user actions was 31, 0.76 per R2R mapping rule, a small effort compared to writing R2R mapping rules in RDF. The process took 11 minutes of interaction with Karma for a user familiar with the sources and the ontology.

In a second evaluation, we mapped a large database of events into the ACE OWL Ontology [12]. The ontology has 127 classes, 74 object properties, 68 data properties and 122 subclass axioms. The database contains 19 tables with a total of 64 columns. We performed this evaluation with no training data for the semantic type identification. All 43 manual semantic type assignments were for types that the system had not seen before, and Karma was able to accurately infer the semantic types for the 21 remaining columns. Karma automatically computed the correct source model for 15 of 19 tables and required one manual relationship adjustment for each of the remaining 4 tables. The average number of nodes in our graph data structure was 108, less than the number of nodes in the ontology (127 classes and 68 types for data properties). The average time for graph construction and Steiner tree computation across the 19 tables was 0.82 seconds, which suggests that the approach scales to real mid-size ontologies. The process took 18 minutes of interaction with Karma.

5 Related Work

There is significant work on schema and ontology matching and mapping [21,6]. An excellent recent survey [22] focuses specifically on mapping relational

databases into the semantic web. Matching discovery tools, such as LSD [10] or COMA [20], produce element-to-element matches based on schemas and/or data. Mapping generation tools, such as Clio [11] and its extensions [2], Altova MapForce (altova.com), or NEON's ODEMapster [4], produce complex mappings based on correspondences manually specified by the user in a graphical interface or produced by matching tools. Most of these tools are geared toward expert users (ontology engineers or DB administrators). In contrast, Karma focuses on enabling *domain* experts to model sources by automating the process as much as possible and providing users an intuitive user interface to resolve ambiguities and tailor the process. Karma produces complex GLAV mappings under the hood, but users do not need to be aware of the logical complexities of data integration/exchange. They see the source data in a familiar spreadsheet format annotated with hierarchical headings, and they can interact with it to correct and refine the mappings.

Alexe et al. [1] elicit complex data exchange rules from examples of source data tuples and the corresponding tuples over the target schema. Karma could use this approach to explain its model to users via examples, and as an alternative method for users to customize the model by editing the examples.

Schema matching techniques have also been used to identify the semantic types of columns by comparing them with labeled columns [10]. Another approach [19] is to learn regular expression-like rules for data in each column and use these expressions to recognize new examples. Our CRF approach [14] improves over these approaches by better handling variations in formats and by exploiting a much wider range of features to distinguish between semantic types that are very similar, such as those involving numeric values.

The combination of the D2R [8] and R2R [7] systems can also express GLAV mappings as Karma. D2R maps a relational database into RDF with a schema closely resembling the database. Then R2R can transform the D2R-produced RDF into a target RDF that conforms to a given ontology using an expressive transformation language. R2RML [9] directly maps a relational database to the desired target RDF. In both cases, the user has to manually write the mapping rules. In contrast, Karma automatically proposes a mapping and lets the user correct/refine the mapping interactively. Karma could easily export its GLAV rules into the R2RML or D2R/R2R formats.

6 Discussion

A critical challenge of the Linked Data cloud is understanding the semantics of the data that users are publishing to the cloud. Currently, users are linking their information at the entity level, but to provide deeper integration of the available data, we also need semantic descriptions in terms of shared ontologies. In this paper we presented a semi-automated approach to building the mappings from a source to a domain ontology.

Often sources require complex cleaning and transformation operations on the data as part of the mapping. We plan to extend Karma's interface to express

these operations and to include them in the source models. In addition, we plan to extend the approach to support modeling a source in which the relationships among columns contain a cycle.

References

1. Alexe, B., ten Cate, B., Kolaitis, P.G., Tan, W.C.: Designing and refining schema mappings via data examples. In: SIGMOD, Athens, Greece, pp. 133–144 (2011)
2. An, Y., Borgida, A., Miller, R.J., Mylopoulos, J.: A semantic approach to discovering schema mapping expressions. In: Proceedings of the 23rd International Conference on Data Engineering (ICDE), Istanbul, Turkey, pp. 206–215 (2007)
3. Arenas, M., Barcelo, P., Libkin, L., Murlak, F.: Relational and XML Data Exchange. Morgan & Claypool, San Rafael (2010)
4. Barrasa-Rodriguez, J., Gómez-Pérez, A.: Upgrading relational legacy data to the semantic web. In: Proceedings of WWW Conference, pp. 1069–1070 (2006)
5. Becker, C., Bizer, C., Erdmann, M., Greaves, M.: Extending smw+ with a linked data integration framework. In: Proceedings of ISWC (2010)
6. Bellahsene, Z., Bonifati, A., Rahm, E.: Schema Matching and Mapping, 1st edn. Springer (2011)
7. Bizer, C., Schultz, A.: The R2R Framework: Publishing and Discovering Mappings on the Web. In: Proceedings of the First International Workshop on Consuming Linked Data (2010)
8. Bizer, C., Cyganiak, R.: D2R Server—publishing relational databases on the semantic web. Poster at the 5th International Semantic Web Conference (2006)
9. Das, S., Sundara, S., Cyganiak, R.: R2RML: RDB to RDF Mapping Language, W3C Working Draft (March 24, 2011), <http://www.w3.org/TR/r2rml/>
10. Doan, A., Domingos, P., Levy, A.Y.: Learning source descriptions for data integration. In: Proceedings of WebDB, pp. 81–86 (2000)
11. Fagin, R., Haas, L.M., Hernández, M.A., Miller, R.J., Popa, L., Velegrakis, Y.: Clio: Schema mapping creation and data exchange. In: Conceptual Modeling: Foundations and Applications - Essays in Honor of John Mylopoulos, pp. 198–236 (2009)
12. Fink, C., Finin, T., Mayfield, J., Piatko, C.: Owl as a target for information extraction systems (2008)
13. Friedman, M., Levy, A.Y., Millstein, T.D.: Navigational plans for data integration. In: Proceedings of AAAI, pp. 67–73 (1999)
14. Goel, A., Knoblock, C.A., Lerman, K.: Using conditional random fields to exploit token structure and labels for accurate semantic annotation. In: Proceedings of AAAI 2011 (2011)
15. Halevy, A.Y.: Answering queries using views: A survey. The VLDB Journal 10(4), 270–294 (2001)
16. Jentzsch, A., Andersson, B., Hassanzadeh, O., Stephens, S., Bizer, C.: Enabling tailored therapeutics with linked data. In: Proceedings of the WWW Workshop on Linked Data on the Web, LDOW (2009)
17. Kou, L., Markowsky, G., Berman, L.: A fast algorithm for steiner trees. Acta Informatica 15, 141–145 (1981)
18. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282–289 (2001)

19. Lerman, K., Plangrasopchok, A., Knoblock, C.A.: Semantic labeling of online information sources. IJSWIS, special issue on Ontology Matching (2006)
20. Massmann, S., Raunich, S., Aumueller, D., Arnold, P., Rahm, E.: Evolution of the coma match system. In: Proceedings of the Sixth International Workshop on Ontology Matching, Bonn, Germany (2011)
21. 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)
22. Spanos, D.E., Stavrou, P., Mitrou, N.: Bringing relational databases into the semantic web: A survey. In: Semantic Web. IOS Pre-press (2011)
23. Tuchinda, R., Knoblock, C.A., Szekely, P.: Building mashups by demonstration. ACM Transactions on the Web (TWEB) 5(3) (2011)

Castor: A Constraint-Based SPARQL Engine with Active Filter Processing

Vianney le Clément de Saint-Marcq^{1,2,3}, Yves Deville¹,
Christine Solnon^{2,4}, and Pierre-Antoine Champin^{2,3}

¹ Université catholique de Louvain, ICTEAM institute,
Place Sainte-Barbe 2, 1348 Louvain-la-Neuve Belgium
{vianney.leclement,yves.deville}@uclouvain.be

² Université de Lyon, LIRIS, CNRS UMR5205, 69622 Villeurbanne France
{christine.solnon,pierre-antoine.champin}@liris.cnrs.fr

³ Université Lyon 1, F-69622 Villeurbanne France

⁴ INSA Lyon, F-69622 Villeurbanne France

Abstract. Efficient evaluation of complex SPARQL queries is still an open research problem. State-of-the-art engines are based on relational database technologies. We approach the problem from the perspective of Constraint Programming (CP), a technology designed for solving NP-hard problems. Such technology allows us to exploit SPARQL filters early-on during the search instead of as a post-processing step. We propose Castor, a new SPARQL engine based on CP. Castor performs very competitively compared to state-of-the-art engines.

1 Introduction

As semantic web technologies adoption grows, the fields of application become broader, ranging from general facts from Wikipedia, to scientific publications metadata, government data, or biochemical interactions. The Resource Description Framework (RDF) [9] provides a standard knowledge representation model, a key component for interconnecting data from various sources. SPARQL [12] is the standard language for querying RDF data sources. Efficient evaluation of such queries is important for many applications.

State-of-the-art SPARQL engines (e.g., Sesame [5], Virtuoso [6] or 4store [7]) are based on relational database technologies. They are mostly designed for scalability, i.e., the ability to handle increasingly large datasets. However, they have difficulties to solve complex queries, even on small datasets.

We approach SPARQL queries from a different perspective. We propose Castor, a new SPARQL engine based on Constraint Programming (CP). CP is a technology for solving NP-hard problems. It has been shown to be efficient for graph matching problems [20,15], which are closely related to SPARQL [3]. Castor is very competitive with the state-of-the-art engines and outperforms them on complex queries.

Contributions. A first technical description of this work has been published in [13]. The present paper presents a number of enhancements of the first Castor prototype, namely:

- more efficient data structures based on a total ordering of RDF values,
- the translation of solution modifiers to the constraints framework,
- the replacement of the SQLite backend by native triple indexes based on the RDF-3x engine.

Finally, we have conducted more comprehensive benchmarks, now comparing Castor with Virtuoso and 4store.

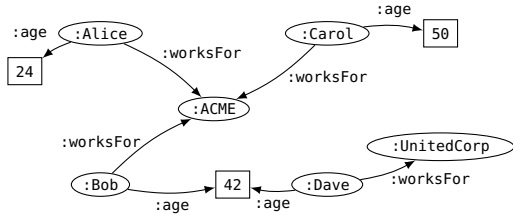
Outline. The next section describes the SPARQL language and how it is implemented in state-of-the-art engines. Section 3 presents our CP approach of SPARQL queries. Section 4 shows the major parts of our system. Section 5 contains the experimental results.

2 Background

Data in the semantic web are represented by a graph [9]. Nodes are identified by URIs¹ and literals, or they may be blank. Edges are directed and labeled by URIs. We will call such a graph an RDF dataset. Figure 1b shows an example of RDF dataset. Note that we can equivalently represent the dataset as a set of triples (Fig. 1a). Each triple describes an edge of the graph. The components of a triple are respectively the source node identifier (the subject), the edge label (the predicate) and the destination node identifier (the object).

```
:Alice :worksFor :ACME .
:Alice :age 24 .
:Bob :worksFor :ACME .
:Bob :age 42 .
:Carol :worksFor :ACME .
:Carol :age 50 .
:Dave :worksFor :UnitedCorp .
:Dave :age 42 .
```

(a) Triple set



(b) Graph representation

Fig. 1. RDF dataset example representing fictive employees

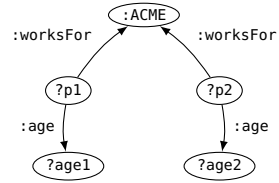
SPARQL [12] is the standard query language for RDF. The basis of a query is a triple pattern, i.e., a triple whose components may be variables. A set of triple patterns is called a basic graph pattern (BGP) as it can be represented by a pattern graph to be matched in the dataset. A solution of a BGP is an assignment of every variable to an RDF value, such that replacing the variables by their assigned values in the BGP yields a subset of the dataset viewed as a triple set. From now on, we will use the triple set representation of the dataset. More complex patterns can be obtained by composing BGPs together and by adding filters. Figure 2 shows an example SPARQL query with one BGP and one filter.

¹ For the sake of readability, throughout the paper we abbreviate URIs to CURIEs (<http://www.w3.org/TR/curie/>).

```

SELECT * WHERE {
  ?p1 :worksFor :ACME .    (P1)
  ?p1 :age ?age1 .        (P2)
  ?p2 :worksFor :ACME .    (P3)
  ?p2 :age ?age2 .        (P4)
  FILTER(?age1 < ?age2)  (F)
}
    
```

(a) SPARQL query



(b) Associated pattern graph

Fig. 2. SPARQL query example on the dataset shown in Fig. 1. The query returns all pairs of employees working at ACME, the first one being younger than the second one.

Formally, let U , B , L and V be pairwise disjoint infinite sets representing URIs, blank nodes, literals and variables, respectively. An RDF dataset is a finite set of triples $G \subset (U \cup B) \times U \times (U \cup B \cup L)$. We respectively denote U_G , B_G and L_G the finite set of URIs, blank nodes and literals occurring in G .

A SPARQL query consists of two parts: a graph pattern and solution modifiers. The graph pattern is defined recursively as follows.

- A basic graph pattern is a set of triple patterns $P = BGP \subset (U \cup V) \times (U \cup V) \times (U \cup L \cup V)$. Without loss of generality, that definition forbids blank nodes from appearing in a graph pattern: blank nodes can be replaced by variables (at least as long as we do not use any SPARQL entailment regime). We denote V_P the set of variables in P .
- Let P be a graph pattern and c be a SPARQL expression such that every variable of c occurs in P . $P \text{ FILTER } c$ is a constrained pattern. We denote V_c the set of variables in c .
- Let P_1 and P_2 be graph patterns. $P_1 \cdot P_2$, $P_1 \text{ OPTIONAL } P_2$ and $P_1 \text{ UNION } P_2$ are compound patterns. We will ignore compound patterns as they are not relevant for the contributions of this paper. However, they are handled by Castor as discussed in [13].

A solution of a graph pattern P with respect to a dataset G is a mapping $\mu : V_P \rightarrow U_G \cup B_G \cup L_G$. We denote $\llbracket P \rrbracket_G$ the set of all solutions. Let $\mu(P)$ (resp. $\mu(c)$) be the pattern (resp. expression) obtained by replacing every occurrence of a variable $?x \in V_P$ (resp. V_c) with its value $\mu(?x)$. The solutions of a basic graph pattern BGP are

$$\llbracket BGP \rrbracket_G = \{ \mu \mid \mu(BGP) \subseteq G \} .$$

The solutions of a graph pattern P constrained by an expression c are

$$\llbracket P \text{ FILTER } c \rrbracket_G = \{ \mu \in \llbracket P \rrbracket_G \mid \mu(c) \text{ evaluates to true} \} .$$

When evaluating a SPARQL query, the solution set of the graph pattern is transformed into a list. Solution modifiers are then applied in the following order.

² The condition on the variables appearing in c restricts the language to *safe* filters, without limiting its expressive power [2].

1. `ORDER BY` sorts the list of solutions,
2. `SELECT` projects the solution on a set of variables, i.e., the domain of the solution mappings are restricted to the specified set of variables,
3. `DISTINCT` removes duplicate solutions,
4. `OFFSET n` removes the n first solutions of the list,
5. `LIMIT n` keeps only the n first solutions of the list.

State-of-the-art SPARQL engines rely on relational database technologies to store the datasets and execute the queries. Such systems can be divided in three categories [8].

- *Triple stores* store the whole dataset in one three-column table. Each row represents one triple. Examples in this category are Sesame, 4store [7], Virtuoso [6], RDF-3x [10] and Hexastore [19].
- *Vertically partitioned tables* maintain one two-column table for each predicate. The resulting smaller tables are sometimes more convenient than the single large table of triple stores. However, there is a significant overhead when variables appear in place of predicates in the query. An example is SW-Store [1].
- *Schema-specific systems* map legacy relational databases to RDF triples using a user-specified ontology. Queries are translated to SQL and performed on the relational tables. Native RDF datasets can be transformed to relational tables if the user provides the structure. Thus, such systems do not handle well the schema-less nature of RDF.

For the purpose of this paper, we will focus on triple stores, which are very popular and well-performing generic engines.

The solutions for a single triple pattern can be retrieved efficiently from a triple store using redundant indexes. Combining multiple triple patterns however involves joining the solution sets together, i.e., merging mappings that assign the same value to common variables. Such operations can be more or less expensive depending on the order in which they are performed. Query engines carefully construct a join graph optimizing the join order. The join graph is then executed bottom-up, starting from the leaves, the triple patterns, and joining the results together. Filters are applied once all their variables appear in a solution set.

The join graph optimization problem has been largely studied for relational databases (e.g., [11][16]). Many results were also adapted to semantic web databases (e.g., [17]).

Figure 3 shows an example of executing a query in a triple store. Here, the filter can only be applied at the very last stage of the evaluation, as it involves variables from different parts of the query.

3 Constraint-Based View of SPARQL Queries

The relational database approach to SPARQL queries focuses on the triple patterns to build the solutions. We propose another view focusing on the variables.

A solution to a query is an assignment of the variables of the query to values of the dataset. The set of values that can be assigned to a variable is called its *domain*. The domain of a variable is initially the set of all URIs, blank nodes and literals occurring

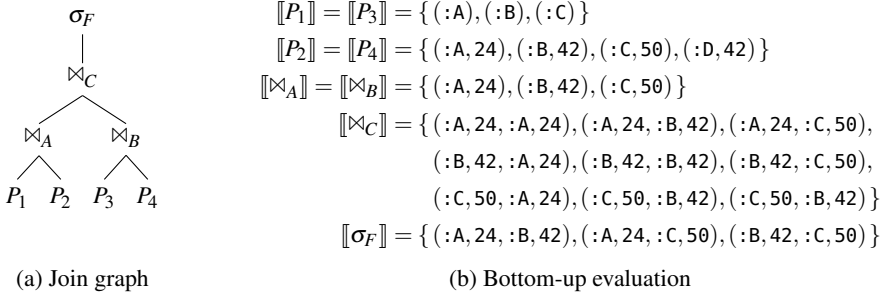


Fig. 3. Executing the query from Fig. 2 in a triple store evaluates the join graph bottom-up. Note that, depending on the used join algorithms, some intermediate results may be produced lazily and need not be stored explicitly. The URIs of the employees are abbreviated by their first letter.

in the dataset. We construct solutions by selecting for each variable a value from its domain and checking that the obtained assignment satisfies the triple patterns and the filters (i.e., the *constraints*).

Constructing all solutions can be achieved by building a search tree. Each node contains the domains of the variables. The root node contains the initial domains. At each node of the tree, a variable is assigned to a value in its domain (i.e., its domain is reduced to a singleton), and constraints are propagated to reduce other variable domains. Whenever a domain becomes empty, the branch of the search tree is pruned. The form of the search tree thus depends on the choice of variable at each node and the order of the children (i.e., how the values are enumerated in the domain of the variables). Let us consider for example the query of Fig. 2. When assigning ?age1 to 42, we can propagate the constraint ?age1 < ?age2 to remove from the domain of ?age2 every value which is not greater than 42 and, if all values are removed, we can prune this branch.

This is the key idea of constraint programming: prune the search tree by using the constraints to remove inconsistent values from the domains of the variables. Each constraint is used successively until the fix-point is reached. This process, called *propagation*, is repeated at every node of the tree. There are different levels of propagation. An algorithm with higher complexity will usually be able to prune more values. Thus, a trade-off has to be found between the achieved pruning and the time taken.

Figure 4 shows the search tree for the running example. At the root node, the triple patterns restrict the domains of ?p1 and ?p2 to only :Alice, :Bob and :Carol, i.e., the employees working at ACME, and ?age1 and ?age2 to { 24, 42, 50 }. The filter removes value 50 from ?age1, as there is no one older than 50. Similarly, 24 is removed from ?age2. Iterating the process, we can further remove :Carol from ?p1 and :Alice from ?p2. Compared to the relational database approach, we are thus able to exploit the filters at the beginning of the search.

The tree is explored in a depth-first strategy. Hence only the path from the root to the current node is kept in memory. In most constraint programming systems, instead of keeping copies of the domains along the path to the root, one maintains the domains of the current node and a *trail*. The trail contains the minimal information needed to restore the current domains to any ancestor node.

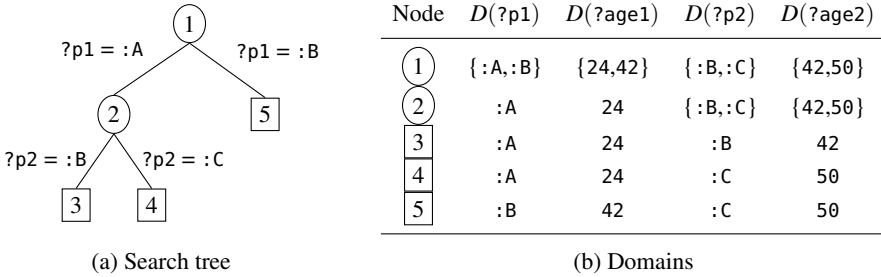


Fig. 4. Executing the query in Fig. 2 with constraint programming explores the search tree top-down. The triple patterns and filters are used at every node to reduce the domains of the variables. The URIs of the employees are abbreviated by their first letter.

4 Implementation

We evaluated the constraint-based approach using a state-of-the-art CP solver in [13]. While such implementation delivered some results, the cost of restoring the domains in generic solvers is too high for large datasets. Hence, we have built a specialized lightweight solver called Castor.

Castor is a prototype SPARQL engine based on CP techniques. When executing a query, a domain is created for every variable of the query, containing all values occurring in the dataset. For efficiency, every value is represented by an integer. Constraints correspond to the triple patterns, filters and solution modifiers. The associated pruning functions, called *propagators* are registered to the domains of the variables on which the constraints are stated. The propagators will then be called whenever the domains are modified. The search tree is explored in a depth-first strategy. A leaf node where every domain is a singleton is a solution, which is returned by the engine.

In this section, we describe the major components of Castor: the constraints and their propagators, the representation of the domains, the mapping of RDF values to integers, and the triple indexes used to store the dataset and propagating the triple pattern constraints.

4.1 Constraints

Constraints and propagators are the core of a CP solver. SPARQL queries have three kinds of constraints: triple patterns, filters and solution modifiers. The associated propagators can achieve different levels of consistency, depending on their complexity and properties of the constraint. We first show the different levels of consistency that are achieved by Castor. Then, we explain the propagators for the different constraints.

To be correct, a propagator should at least ensure that the constraint is satisfied once every variable in the constraint is bound (i.e., its domain is a singleton). However, to reduce the search space, propagators can prune the domains when variables are unbound. Propagators can be classified by their achieved level of consistency [4], i.e., the amount of pruning they can achieve.

- A propagator achieving forward-checking consistency does nothing until all variables in the constraint are bound, except one. It then iterates over the domain of the unbound variable, removing all values that do not satisfy the constraint. The required operation on the domains is the removal of a value.
- Bound consistency ensures that the bounds (i.e., the minimum and maximum value) of the domains of the variables in the constraint are consistent. A value is consistent if there exists a solution of the constraint with that value. The required operation on the domains is the update of a bound (i.e., increasing the lower bound or decreasing the upper bound).
- Domain consistency ensures that every value in the domains of the variables in the constraint are consistent. The required operation on the domains is the removal of a value. Domain consistency is the strongest level of consistency we consider. However, propagators achieving domain consistency usually have a higher complexity and could require maintaining auxiliary data structures.

Triple Patterns. A triple pattern is a constraint involving three variables, one for each component. For ease of reading, we consider constants to be variables whose domains are singletons. The pruning is performed by retrieving all the triples from the dataset where the components of the bound variables correspond to the assigned value. Values of domains of unbound variables that do not appear in the resulting set of triples are pruned. If the pruning is performed with only one unbound variable, we achieve forward-checking consistency. Castor achieves more pruning by performing the pruning when one or two variables are unbound. If all three variables are bound, the propagator checks if the triple is in the dataset and empties a domain if this is not the case.

Filters. Castor has a generic propagator for filters achieving forward-checking consistency. When traversing the domain of the unbound variable, we can check if the filter is satisfied by evaluating the SPARQL expression as described by the W3C recommendation [12]. It provides a fallback to easily handle any filter, but is not very efficient. When possible, specialized algorithms are preferred.

For example, the propagator for the `sameTERM(?x, ?y)` filter can easily achieve domain consistency. The constraint states that `?x` and `?y` are the same RDF term. The domains of both variables should be the same. Hence, when a value is removed from one domain, the propagator removes that value from the other domain.

Propagators for monotonic constraints [18], e.g., `?x < ?y`, can easily achieve bound consistency. Indeed, for constraint `?x < ?y`, we have $\max(?x) < \max(?y)$ and $\min(?x) < \min(?y)$. The pruning is performed by adjusting the upper bound of `?x` and the lower bound of `?y`.

Solution Modifiers. The `DISTINCT` keyword in SPARQL removes duplicates from the results. Such operation can also be handled by constraints. When a solution is found, a new constraint is added stating the any further solution must be different from the current one. The propagator achieves forward-checking consistency, i.e., when all variables but one are bound, we remove the value of the already found solution from the domain of the unbound variable.

When the `ORDER BY` and `LIMIT` keywords are used together, the results shall only include the n best solutions according to the specified ordering. After n solutions have been found, we add a new constraint stating that any new solution must be “better” than the worst solution so far. Such technique is known as branch-and-bound.

4.2 Mapping RDF Values to Integers

To avoid juggling with heavy data structures representing the RDF values, we map every value occurring in the dataset to a numerical identifier. Such mapping is also common in triple stores. The domains in Castor contain only those identifiers. An on-disk dictionary allows the retrieval of the associated value when needed.

Let $\text{id}(v)$ be the identifier mapped to the RDF value v . To efficiently implement a bounds consistent propagator for the $<$ filter, we want $v_1 <_F v_2 \Rightarrow \text{id}(v_1) < \text{id}(v_2)$, where $<_F$ is the $<$ operator of SPARQL expressions. The SPARQL specification only defines $<_F$ between numerical values, between simple literals, between strings, between Boolean values, and between timestamps. The $<_F$ operator thus defines a partial order.

To efficiently implement the `ORDER BY` solution modifier, we also want $v_1 <_O v_2 \Rightarrow \text{id}(v_1) < \text{id}(v_2)$, where $<_O$ is the partial order defined in the SPARQL specification. This order introduces a precedence between blank nodes, URIs and literals. Literals are ordered with $<_F$. Hence, $v_1 <_F v_2 \Rightarrow v_1 <_O v_2$.

To map each RDF value to a unique identifier, we introduce a total order $<_T$ that is compatible with both partial orders, i.e.,

$$\forall (v_1, v_2) \in (U \cup B \cup L) \times (U \cup B \cup L), v_1 <_O v_2 \Rightarrow v_1 <_T v_2 .$$

Values are partitioned into the following classes, shown in ascending order. The ordering of the values inside each class is also given. When not specified, or to solve ambiguous cases, the values are ordered by their lexical form.

1. Blank nodes: ordered by their internal identifier
2. URIs
3. Plain literals without language tags
4. `xsd:string` literals
5. Boolean literals: first false, then true values
6. Numeric literals: ordered first by their numerical value, then by their type URI
7. Date/time literals: ordered chronologically
8. Plain literals with language tags: ordered first by language tag, then by lexical form
9. Other literals: ordered by their type URI

We map the values of a dataset to consecutive integers starting from 1, such that $v_1 <_T v_2 \Leftrightarrow \text{id}(v_1) < \text{id}(v_2)$.

4.3 Variables and Domains

A domain is associated with every variable, representing the set of values that can be assigned to the variable. During the search, the domain gets reduced and restored. The data

structures representing the domain should perform such operations efficiently. There are two kinds of representations. The *discrete* representation keeps track of every single value in the domain. The *bounds* representation only keeps the lowest and highest value of the domain according to the total order defined in Section 4.2. We propose a dual view, leveraging the strengths of both representations.

Discrete Representation. The domain is represented by its size and two arrays dom and map. The size first elements of dom are in the domain of the variable, the others have been removed (see Fig. 5). The map array maps values to their position in the dom array.

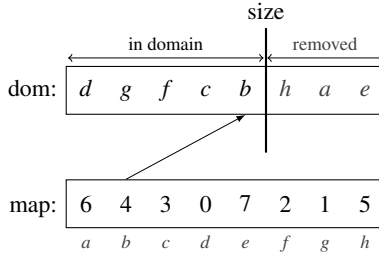


Fig. 5. Example representation of the domain $\{b, c, d, f, g\}$, such that $size = 5$, when the initial domain is $\{a, \dots, h\}$. The size first values in dom belong to the domain; the last values are those which have been removed. The map array maps values to their position in dom. For example, value b has index 4 in the dom array. In such representation, only the size needs to be kept in the trail.

To remove a value, we swap it with the last value of the domain (i.e., the value directly to the left of the *size* marker), reduce *size* by one and update the map array. Such operation is done in constant time.

Alternatively, we can restrict the domain to the intersection of itself and a set S . We move all values of S which belong to the *size* first elements of dom at the beginning of dom and set *size* to the size of the intersection. Such operation is done in $O(|S|)$, with $|S|$ the size of S . Castor uses the restriction operation in propagators achieving forward-checking consistency.

Operations on the bounds however are inefficient. This major drawback is due to the unsorted dom array. Searching for the minimum or maximum value requires the traversal of the whole domain. Increasing the lower bound or decreasing the upper bound involves removing every value between the old and new bound one by one.

As the order of the removed values is not modified by any operation, the domain can be restored in constant time by setting the *size* marker back to its initial position. The trail, i.e., the data structure needed to restore the domain to any ancestor node of the search tree, is thus a stack of the sizes.

Note that this is not the standard representation of discrete domains in CP. However, the trail of standard representations is too heavy for our purpose and size of data.

Bounds Representation. The domain is represented by its bounds, i.e., its minimum and maximum values. In contrast to the discrete representation, the bound representation is

an approximation of the exact domain. We assume all values between the bounds are present in the domain.

In such a representation, we cannot remove a value in the middle of the domain as we cannot represent a hole inside the bounds. However, increasing the lower bound or decreasing the upper bound is done in constant time.

The data structure for this representation being small (only two numbers), the trail contains copies of the whole data structure. Restoring the domains involves restoring both bounds.

Dual View. Propagators achieving forward-checking or domain consistency remove values from the domains. Thus, they require a discrete representation. However, propagators achieving bounds consistency only update the bounds of the domains. For them to be efficient, we need a bounds representation. Hence, Castor creates two variables x_D and x_B (resp. with discrete and bound representation) for every SPARQL variable $?x$. Constraints are stated using only one of the two variables, depending on which representation is the most efficient for the associated propagator. In particular, monotonic constraints are stated on bounds variables whereas triple pattern constraints are stated on discrete variables.

An additional constraint $x_D = x_B$ ensures the correctness of the dual approach. Achieving domain consistency for this constraint is too costly, as it amounts to perform every operation on the bounds on the discrete representation. Instead, the propagator in Castor achieves forward-checking consistency, i.e., once one variable is bound the other will be bound to the same value. As an optimization, when restricting a domain to its intersection with a set S , we filter out values of S which are outside the bounds and update the bounds of x_B . Such optimization does not change the complexity of the operation, as it has to traverse the whole set S anyway.

4.4 Triple Indexes

The propagator of the triple pattern constraint needs to retrieve a subset of triples from the dataset, where some components have a specific value (see Section 4.1). The efficiency of such operation depends on the way the triples are stored on the disk. Castor makes use of indexes to retrieve the triples. An index sorts the triples in lexicographical order and provides efficient retrieval of triples with a fixed prefix. For example, the SPO index sorts the triples first by subject, then by predicate and last by object. It can be used to retrieve all triples with a specific subject or all triples with specific subject and predicate. It can also be used to check whether a triple is part of the dataset. Castor has three indexes: SPO, POS and OSP to cover all possible combinations.

The data structure underlying an index is based on the RDF-3x engine [10]. The sorted triples are compressed and packed in pages (i.e., a block of bytes of fixed size, in our case 16KB). Those pages are the leaves of a B+-tree. The tree allows one to find the leaf containing the first requested triple in logarithmic time. After the decompression, we can find the triple using a binary search algorithm.

Propagators are called multiple times at every node of the search tree. Thus, a set of triples can be requested many times. To reduce the overhead of decompressing the leaf pages containing the triples, we introduce a small least-recently-used cache of decompressed pages. The size of the cache is currently arbitrarily fixed to 100 pages.

5 Experimental Results

To assess the performances of our approach, we have run the SPARQL Performance Benchmark (SP²Bench) [14]. SP²Bench consists of a deterministic dataset generator, and 12 representative queries to be executed on the generated datasets. The datasets represent relationships between fictive academic papers and their authors, following the model of academic publications in the DBLP database.

We compare the performances of four engines: Sesame 2.6.1 [5], Virtuoso 6.1.4 [6], 4store 1.1.4 [7] and our own Castor. Sesame was configured to use its native on-disk store with three indexes (spoc, posc, ospc). The other engines were left in their default configuration. We did not include RDF-3x in the comparison as it is unable to handle the filters appearing in the queries. For queries involving filters, we have also tested a version of Castor that does not post them as constraints, but instead evaluate them in a post-processing step.

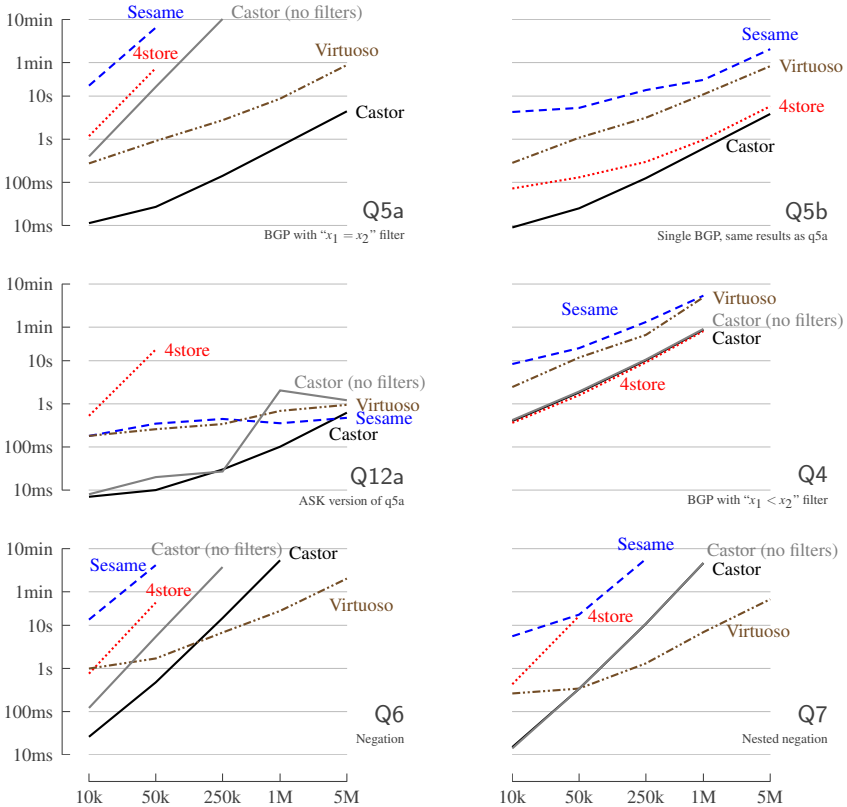


Fig. 6. Castor is competitive and often outperforms state-of-the-art SPARQL engines on complex queries. The x-axis represents the dataset size in terms of number of triples. The y-axis is the query execution time. Both axes have a logarithmic scale.

We have generated 6 datasets with 10k, 25k, 250k, 1M and 5M triples. We have performed three cold runs of each query over all the generated datasets, i.e., between two runs the engines were restarted and the system caches cleared with “`sysctl -w vm.drop_caches=3`”. We have set a timeout of 30 minutes. Please note that cold runs may not give the most significant results for some engines. E.g., Virtuoso aggressively fills its cache on the first query in order to perform better on subsequent queries. However, such setting corresponds to the one used by the authors of SP²Bench, so we have chosen to use it as well. All experiments were conducted on an Intel Pentium 4 3.2 GHz computer running ArchLinux 64bits with kernel 3.2.6, 3 GB of DDR-400 RAM and a 40 GB Samsung SP0411C SATA/150 disk with ext4 filesystem. We report the time spent to execute the queries, not including the time needed to load the datasets.

The authors of SP²Bench have identified four queries that are more challenging than the others: Q4, Q5a, Q6 and Q7. The execution time of those queries, along with two variations of Q5a, are reported in Figure 6.

Q5a and Q5b compute the same set of solutions. Q5a enforces the equality of two variables with a filter, whereas Q5b uses a single variable for both. Note that such optimization is difficult to do automatically, as equivalence does not imply identity in SPARQL. For example, “`42"^^xsd:integer`” and “`42.0"^^xsd:decimal`” compare equal in a filter, but are not the same RDF term and may thus not be matched in a BGP. Detecting whether one can replace the two equivalent variables by a single one requires a costly analysis of the dataset, which is not performed by any of the tested engines. Sesame and 4store timed out when trying to solve Q5a on the 250k and above datasets. Virtuoso does not differentiate equivalent values and treats equality as identity. Such behavior breaks the SPARQL standard and can lead to wrong results. Castor does no query optimization, but still performs equally well on both variants thanks to its ability to exploit the filter at every node of the search tree. Q12a replaces the SELECT keyword by ASK in Q5a. The solution is a boolean value reflecting whether there exists a solution to the query. Thus, we only have to look for the first solution. However, Castor still needs to initialize the search tree, which is the greatest cost. Virtuoso and 4store behave similarly to Q5a, but Sesame is able to find the answer much more quickly.

Executing Q4 results in many solutions (e.g., for the 1M dataset, Q4 results in 2.5×10^6 solutions versus 3.5×10^4 solutions for Q5a). The filter does not allow for much pruning, as shown by the very similar performances between the two variants of Castor. Nevertheless, Castor is still competitive with the other engines. None of the engines were able to solve the query for the 5M dataset in less than 30 minutes.

Table 1. Castor is the fastest or second fastest engine for nearly every query. The ranking of the engines is shown for each query. The last column is the average rank for every engine.

Query	1	2	3a	3b	3c	4	5a	5b	6	7	8	9	10	11	12a	12b	Mean
Castor	2	2	2	2	2	2	1	1	2	2	1	1	2	3	2	1	1.8
4store	1	1	1	1	1	1	3	2	3	4	3	2	1	1	4	3	2.0
Virtuoso	4	3	3	3	3	3	2	3	1	1	2	3	3	2	3	2	2.6
Sesame	3	4	4	4	4	4	4	4	4	3	4	4	4	4	1	4	3.7

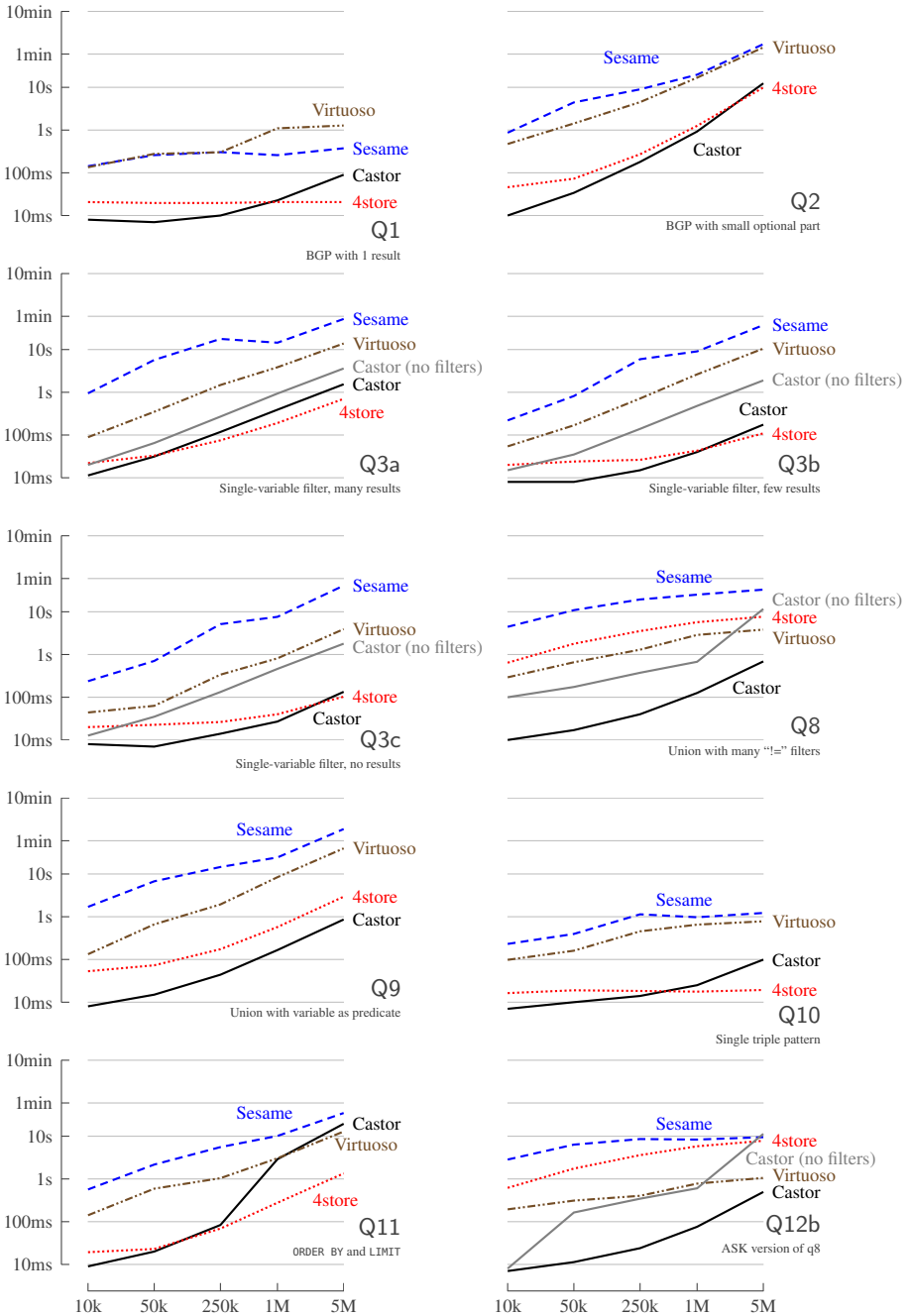


Fig. 7. On simpler queries, Castor is also very competitive with state-of-the-art SPARQL engines. The x-axis represents the dataset size in terms of number of triples. The y-axis is the query execution time. Both axes have a logarithmic scale.

Figure 7 shows the results for the other queries, except Q12c. Query Q12c involves an RDF value that is not present in the dataset. It is solved in constant time by all engines equally well. For all queries, Castor is competitive with the other engines or outperforming them. The sharp decrease of performances of Castor in Q11 between the 250k and the 1M datasets is due to the fixed size of the triple store cache. The hit ratio drops from 99.6% to 40.4%.

For each query, we sort the engines in lexicographical order, first by the largest dataset solved, then by the execution time on the largest dataset. The obtained ranks are shown in Table 1. Castor is ranked first for 5 queries out of 16, and second for all other queries but one. The 4store engine is ranked first on 8 queries, but does not fare as well on the other queries. In most of the queries where 4store is ranked first, the execution time of Castor is very close to the execution time of 4store. Virtuoso performs well on some difficult queries (Q6 and Q7), but is behind for the other queries. Sesame performs the worst of the tested engines.

6 Conclusion

We presented a Constraint Programming approach to solving SPARQL queries. In contrast to the relational database approach, we are able to exploit filters early-on during the search without requiring advanced query optimization. We showed the main design decision in the implementation of Castor, our prototype SPARQL engine based on CP techniques. We compared Castor with state-of-the-art engines, showing the feasibility and performance of our approach.

Castor is however still an early prototype. It has room for several improvements and extensions. For example, the form of the search tree is defined by basic heuristic functions. At each node, we select the variable with the smallest domain. An alternative might be selecting central variables of star-shaped queries first. Also no research has been done yet to find an optimal ordering of the propagators: they are simply called successively until a fix-point is reached. To reach the fix-point more quickly, it would be better to call first the propagators performing more pruning. Maybe selectivity estimates, a tool used in relational databases [17], could be used to order the propagators. This raises the more general question of whether and how optimization techniques used in relational databases can be combined with the CP approach.

Acknowledgments. The authors want to thank the anonymous reviewers for their constructive comments. The first author is supported as a Research Assistant by the Belgian FNRS (National Fund for Scientific Research). This research is also partially supported by the Interuniversity Attraction Poles Programme (Belgian State, Belgian Science Policy) and the FRFC project 2.4504.10 of the Belgian FNRS.

References

1. Abadi, D.J., Marcus, A., Madden, S.R., Hollenbach, K.: SW-Store: a vertically partitioned DBMS for semantic web data management. *The VLDB Journal* 18, 385–406 (2009)

2. Angles, R., Gutierrez, C.: The Expressive Power of SPARQL. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 114–129. Springer, Heidelberg (2008)
3. Baget, J.-F.: RDF Entailment as a Graph Homomorphism. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 82–96. Springer, Heidelberg (2005)
4. Bessiere, C.: Handbook of Constraint Programming, ch. 3. Elsevier Science Inc. (2006)
5. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 54–68. Springer, Heidelberg (2002)
6. Erling, O., Mikhailov, I.: RDF Support in the Virtuoso DBMS. In: Pellegrini, T., Auer, S., Tochtermann, K., Schaffert, S. (eds.) Networked Knowledge - Networked Media. Studies in Computational Intelligence, vol. 221, pp. 7–24. Springer, Heidelberg (2009)
7. Harris, S., Lamb, N., Shadbolt, N.: 4store: The design and implementation of a clustered RDF store. In: 5th International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2009), at ISWC 2009 (2009)
8. Hose, K., Schenkel, R., Theobald, M., Weikum, G.: Database foundations for scalable RDF processing. In: Polleres, A., d'Amato, C., Arenas, M., Handschuh, S., Kroner, P., Ossowski, S., Patel-Schneider, P. (eds.) Reasoning Web 2011. LNCS, vol. 6848, pp. 202–249. Springer, Heidelberg (2011)
9. Klyne, G., Carroll, J.J., McBride, B.: Resource description framework (RDF): Concepts and abstract syntax (2004), <http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>
10. Neumann, T., Weikum, G.: RDF-3X: a RISC-style engine for RDF. Proc. VLDB Endow. 1, 647–659 (2008)
11. Piatetsky-Shapiro, G., Connell, C.: Accurate estimation of the number of tuples satisfying a condition. In: Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data, SIGMOD 1984, pp. 256–276. ACM, New York (1984)
12. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF (January 2008), <http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/>
13. le Clément de Saint-Marcq, V., Deville, Y., Solnon, C.: An Efficient Light Solver for Querying the Semantic Web. In: Lee, J. (ed.) CP 2011. LNCS, vol. 6876, pp. 145–159. Springer, Heidelberg (2011)
14. Schmidt, M., Hornung, T., Lausen, G., Pinkel, C.: SP²Bench: A SPARQL performance benchmark. In: Proc. IEEE 25th Int. Conf. Data Engineering, ICDE 2009, pp. 222–233 (2009)
15. Solnon, C.: Alldifferent-based filtering for subgraph isomorphism. Artificial Intelligence 174(12-13), 850–864 (2010)
16. Steinbrunn, M., Moerkotte, G., Kemper, A.: Heuristic and randomized optimization for the join ordering problem. The VLDB Journal 6, 191–208 (1997)
17. Stocker, M., Seaborne, A., Bernstein, A., Kiefer, C., Reynolds, D.: SPARQL basic graph pattern optimization using selectivity estimation. In: Proceeding of the 17th International Conference on World Wide Web, WWW 2008, pp. 595–604. ACM, New York (2008)
18. Van Hentenryck, P., Deville, Y., Teng, C.M.: A generic arc-consistency algorithm and its specializations. Artificial Intelligence 57(2-3), 291–321 (1992)
19. Weiss, C., Karras, P., Bernstein, A.: Hexastore: sextuple indexing for semantic web data management. Proc. VLDB Endow. 1, 1008–1019 (2008)
20. Zampelli, S., Deville, Y., Solnon, C.: Solving subgraph isomorphism problems with constraint programming. Constraints 15, 327–353 (2010)

A Structural Approach to Indexing Triples

François Picalausa¹, Yongming Luo², George H.L. Fletcher²,
Jan Hidders³, and Stijn Vansummeren¹

¹ Université Libre de Bruxelles, Belgium
{fpicalau,stijn.vansummeren}@ulb.ac.be

² Eindhoven University of Technology, The Netherlands
{y.luo,g.h.l.fletcher}@tue.nl

³ Delft University of Technology, The Netherlands
{a.j.h.hidders}@tudelft.nl

Abstract. As an essential part of the W3C’s semantic web stack and linked data initiative, RDF data management systems (also known as triplestores) have drawn a lot of research attention. The majority of these systems use *value-based indexes* (e.g., B⁺-trees) for physical storage, and ignore many of the structural aspects present in RDF graphs. *Structural indexes*, on the other hand, have been successfully applied in XML and semi-structured data management to exploit structural graph information in query processing. In those settings, a structural index groups nodes in a graph based on some equivalence criterion, for example, indistinguishability with respect to some query workload (usually XPath). Motivated by this body of work, we have started the SAINT-DB project to study and develop a native RDF management system based on structural indexes. In this paper we present a principled framework for designing and using RDF structural indexes for practical fragments of SPARQL, based on recent formal structural characterizations of these fragments. We then explain how structural indexes can be incorporated in a typical query processing workflow; and discuss the design, implementation, and initial empirical evaluation of our approach.

1 Introduction

As an essential part of the W3C’s semantic web stack, the RDF data model is finding increasing use in a wide range of web data management scenarios, including linked data^[1]. Due to its increasing popularity and application, recent years have witnessed an explosion of proposals for the construction of native RDF data management systems (also known as triplestores) that store, index, and process massive RDF data sets.

While we refer to recent surveys such as [12] for a full overview of these proposals, we can largely discern two distinct classes of approaches. *Value-based approaches* focus on the use of robust relational database technologies such as B⁺-trees and column-stores for the physical indexing and storage of massive

¹ <http://linkeddata.org/>

RDF graphs, and employ established relational database query processing techniques for the processing of SPARQL queries [1, 7, 15, 18, 22]. While value-based triplestores have proven successful in practice, they mostly ignore the native graph structure like paths and star patterns that naturally occur in RDF data sets and queries. (Although some value-based approaches consider extensions to capture and materialize such common patterns in the data graph [1, 15].)

Graph-based approaches, in contrast, try to capture and exploit exactly this richer graph structure. Examples include GRIN [20] and DOGMA [6], that propose index structures based on graph partitioning and distances in the graphs, respectively. A hybrid approach is taken in dipLODocus[RDF], where value-based indexes are introduced for more or less homogeneous sets of subgraphs [23]. These somewhat ad-hoc approaches work well for an established query workload or class of graph patterns, but it is unclear how the indexed patterns can flexibly support general SPARQL queries outside of the supported set.

Structural indexes have been successfully applied in the semi-structured and XML data management context to exploit structural graph information in query processing. A structural index is essentially a reduced version of the original data graph where nodes have been merged according to some notion of structural similarity such as bisimulation [4, 5, 9, 13]. These indexes effectively take into account the structure of both the graph and query, rather than just the values appearing in the query as is the case for value-based indexes. Furthermore, the success of structural indexes hinges on a precise coupling between the expressive power of a general query language and the organization of data by the indexes [9]. The precise class of queries that they can support is therefore immediately clear, thereby addressing the shortcomings of other graph-based approaches.

While structural indexes have been explored for RDF data, for example in the Parameterizable Index Graph [19] and gStore [24] proposals, these proposals simplify the RDF data model to that of *resource-centric* edge-labeled graphs over a *fixed* property label alphabet (disallowing joins on properties), which is not well-suited to general SPARQL query evaluation where pattern matching is *triple-centric*, i.e., properties have the same status as subjects and objects. The relevance of such queries is observed by studies of the usage of SPARQL in practice [3, 16]. Furthermore, there is no tight coupling of structural organization of these indexes to the expressivity of a practical fragment of SPARQL.

Motivated by these observations, we have initiated the SAINT-DB (Structural Approach to INdexing Triples DataBase) project to study the foundations and engineering principles for native RDF management systems based on structural indexes that are faithful to both the RDF data model and the SPARQL query language. As a initial foundation, we have recently established a precise structural characterization of practical SPARQL fragments in terms of graph simulations [8]. Our goal in SAINT-DB is to leverage this characterization in the design of native structural indexing solutions for massive RDF data sets.

Contributions and Overview. In this article, we report on our first results in SAINT-DB. In particular, we make the following contributions: (1) A new notion of structural index for RDF data is introduced that, reflecting the SPARQL

t_1 : (<i>sue</i> , <i>type</i> , <i>CEO</i>)	t_{10} : (<i>reportsTo</i> , <i>type</i> , <i>socialRel</i>)
t_2 : (<i>crispin</i> , <i>type</i> , <i>VP</i>)	t_{11} : (<i>friendOf</i> , <i>type</i> , <i>socialRel</i>)
t_3 : (<i>sue</i> , <i>manages</i> , <i>joe</i>)	t_{12} : (<i>knows</i> , <i>type</i> , <i>socialRel</i>)
t_4 : (<i>joe</i> , <i>reportsTo</i> , <i>jane</i>)	t_{13} : (<i>bestFriendOf</i> , <i>type</i> , <i>socialRel</i>)
t_5 : (<i>jane</i> , <i>friendOf</i> , <i>lucy</i>)	t_{14} : (<i>dislikes</i> , <i>type</i> , <i>socialRel</i>)
t_6 : (<i>crispin</i> , <i>knows</i> , <i>larry</i>)	t_{15} : (<i>yonei</i> , <i>knows</i> , <i>yongsik</i>)
t_7 : (<i>larry</i> , <i>bestFriendOf</i> , <i>sarah</i>)	t_{16} : (<i>yongsik</i> , <i>reportsTo</i> , <i>tamae</i>)
t_8 : (<i>sarah</i> , <i>dislikes</i> , <i>hiromi</i>)	t_{17} : (<i>kristi</i> , <i>manages</i> , <i>filip</i>)
t_9 : (<i>manages</i> , <i>type</i> , <i>socialRel</i>)	t_{18} : (<i>filip</i> , <i>bestFriendOf</i> , <i>sriram</i>)

Fig. 1. A small RDF graph, with triples labeled for ease of reference

language, contains complete triple information and therefore allows for the retrieval of sets of triples rather than sets of resources. **(2)** A formalization of the structural index, coupled to the expressivity of practical fragments of SPARQL, is given, together with the algorithms for building and using it. **(3)** We demonstrate the effective integration of structural indexing into a state-of-the-art triple store with cost-based query optimization.

We proceed as follows. In Sec. **2** we introduce our basic terminology for querying RDF data. In Sec. **3** we present the principles behind triple-based structural indexes for RDF. In Sec. **4** we then discuss how these principles can be put into practice in a state of the art triple store. In Sec. **5**, we present an empirical study where the effectiveness of the new indices within this extended triple store is demonstrated. Finally, in Sec. **6** we present our main conclusions and give indications for further research.

2 Preliminaries

RDF. All information in RDF is uniformly represented by triples of the form (s, p, o) over some fixed but unspecified universe \mathcal{U} , $(s, p, o) \in \mathcal{U}^3$. Here, s is called the *subject*, p is called the *predicate*, and o is called the *object*. An *RDF graph* D is a finite set of *RDF triples*, $D \subseteq \mathcal{U}^3$. To illustrate, a small RDF graph of social relationships in a corporate setting is given in Fig. **1**.

BGP Queries. RDF comes equipped with the SPARQL **[17]** language for querying data in RDF format. Using so-called *basic graph patterns* (BGPs for short) as building blocks, SPARQL queries search for specified subgraphs of the input RDF graph. While SPARQL queries can be more complex in general, we will focus in this article on so-called *BGP queries*: SPARQL queries that consist of basic graph patterns only. The reason for this is threefold. First and foremost, the evaluation of basic graph patterns is a problem that occurs as a subproblem in all SPARQL query evaluation problems. Second, BGP queries correspond to the well-known class of conjunctive queries from relational databases. Third, recent analysis of real-world SPARQL query logs has illustrated that the majority of SPARQL queries posed in practice are BGP queries **[3, 16]**.

The formal definition of BGP queries is as follows. Let $\mathcal{V} = \{?x, ?y, ?z, \dots\}$ be a set of variables, disjoint from \mathcal{U} . A *triple pattern* is an element of $(\mathcal{U} \cup \mathcal{V})^3$. We write $\text{vars}(p)$ for the set of variables occurring in triple pattern p . A *basic graph pattern* (BGP for short) is a set of triple patterns. A *BGP query* (or simply *query* for short) is an expression Q of the form $\text{SELECT } X \text{ WHERE } P$ where P is a BGP and X is a subset of the variables mentioned in P .

Example 1. As an example, the following BGP query retrieves, from the RDF graph of Fig. 1, those pairs of people p_a and p_c such that p_a is a CEO and p_c has a social relationship with someone directly related to p_a .

```
SELECT ?pa, ?pc
WHERE { (?pa, type, CEO), (?pa, ?relab, ?pb), (?pb, ?relbc, ?pc),
        (?relab, type, socialRel), (?relbc, type, socialRel) } □
```

To formally define the semantics of triple patterns, BGPs, and BGP queries, we need to introduce the following concepts. A *mapping* μ is a partial function $\mu: \mathcal{V} \rightarrow \mathcal{U}$ that assigns values in \mathcal{U} to a finite set of variables. The domain of μ , denoted by $\text{dom}(\mu)$, is the subset of \mathcal{V} where μ is defined. The restriction $\mu[X]$ of μ to a set of variables $X \subseteq \mathcal{V}$ is the mapping with domain $\text{dom}(\mu) \cap X$ such that $\mu[X](?x) = \mu(?x)$ for all $?x \in \text{dom}(\mu) \cap X$. Two mappings μ_1 and μ_2 are *compatible*, denoted $\mu_1 \sim \mu_2$, when for all common variables $?x \in \text{dom}(\mu_1) \cap \text{dom}(\mu_2)$ it is the case that $\mu_1(?x) = \mu_2(?x)$. Clearly, if μ_1 and μ_2 are compatible, then $\mu_1 \cup \mu_2$ is again a mapping. We define the join of two sets of mappings Ω_1 and Ω_2 as $\Omega_1 \bowtie \Omega_2 := \{\mu_1 \cup \mu_2 \mid \mu_1 \in \Omega_1, \mu_2 \in \Omega_2, \mu_1 \sim \mu_2\}$, and the projection of a set of mappings Ω to $X \subseteq \mathcal{V}$ as $\pi_X(\Omega) := \{\mu[X] \mid \mu \in \Omega\}$. If p is a triple pattern then we denote by $\mu(p)$ the triple obtained by replacing the variables in p according to μ . Semantically, triple patterns, BGPs, and queries evaluate to a set of mappings when evaluated on an RDF graph D :

$$\begin{aligned} \llbracket p \rrbracket_D &:= \{\mu \mid \text{dom}(\mu) = \text{vars}(p) \text{ and } \mu(p) \in D\}, \\ \llbracket \{p_1, \dots, p_n\} \rrbracket_D &:= \llbracket p_1 \rrbracket_D \bowtie \dots \bowtie \llbracket p_n \rrbracket_D, \\ \llbracket \text{SELECT } X \text{ WHERE } P \rrbracket_D &:= \pi_X(\llbracket P \rrbracket_D). \end{aligned}$$

Example 2. Let Q be the query of Example 1 and D be the dataset of Fig. 1. Then $\llbracket Q \rrbracket_D = \{(?p_a \mapsto \text{sue}, ?p_c \mapsto \text{jane})\}$. In other words, Q evaluated on D contains a single mapping μ , where $\mu(?p_a) = \text{sue}$ and $\mu(?p_c) = \text{jane}$. □

3 Principles of Triple-Based Structural Indexing

To evaluate a BGP $P = \{p_1, \dots, p_n\}$ on an RDF graph D we need to perform $n - 1$ joins $\llbracket p_1 \rrbracket_D \bowtie \dots \bowtie \llbracket p_n \rrbracket_D$ between subsets of D . Since D is large in practice, we are interested in pruning as much as possible the subsets $\llbracket p_i \rrbracket_D$ of D that need to be joined, where $1 \leq i \leq n$. In particular, we are interested in efficiently removing from $\llbracket p_i \rrbracket_D$ any “dangling” triples that do not participate in the full join. Towards this purpose, we next introduce the notions of equality type and structural index.

Definition 1. An equality type is a set of pairs (i, j) with $1 \leq i, j \leq 3$. Intuitively, a pair (i, j) in an equality type indicates the position in which two triples share a common value. In particular, let $t = (t_1, t_2, t_3)$ and $u = (u_1, u_2, u_3)$ be two RDF triples or two triple patterns. Then the equality type of t and u , denoted $eqtp(t, u)$, is defined as $eqtp(t, u) := \{(i, j) \mid t_i = u_j \text{ and } 1 \leq i, j \leq 3\}$.

Essentially, the equality type of t and u specifies the kinds of natural joins that t and u can participate in. For example, when evaluating the BGP $\{(?x, ?y, 1), (?z, ?x, ?y)\}$ we are looking for triples t and u such that $\{(1, 2), (2, 3)\} \subseteq eqtp(t, u)$. This is a necessary condition in general for a mapping to be in a BGP result:

$$\mu \in \llbracket P \rrbracket_D \Rightarrow eqtp(p, q) \subseteq eqtp(\mu(p), \mu(q)) \text{ for all } p, q \in P.$$

Intuitively speaking, a structural index (defined below) groups triples into index blocks, and summarizes the equality types that exist between triples in those blocks. The necessary condition above can then be used to prune triples that can never realize the desired equality type, by looking at the structural index only.

Definition 2 (Structural Index). Let \mathcal{T} denote the set of all equality types and let D be an RDF graph. A structural index for D is an edge-labeled graph $I = (N, E)$ where N is a finite set of nodes, called the blocks of the index, and $E \subseteq N \times \mathcal{T} \times N$ is a set of edges labeled by equality types. The nodes N of I must be sets of triples in D (i.e., $N \subseteq 2^D$), and must form a partition of D . We write $[t]_I$ to denote the unique block of I that contains $t \in D$. Furthermore, it is required that E reflects the equality types between the triples in its blocks, in the sense that for all $t, u \in D$ we must have $([t]_I, eqtp(t, u), [u]_I) \in E$.

An embedding of a BGP P into a structural index I is a function $\alpha: P \rightarrow N$ that assigns to each triple pattern $p \in P$ a node $\alpha(p) \in N$ such that for every $p, q \in P$ there exists $\tau \in \mathcal{T}$ with $eqtp(p, q) \subseteq \tau$ and $(\alpha(p), \tau, \alpha(q)) \in E$.

Example 3. Consider the graph shown in Fig. 2, where nodes are labeled with triples from the dataset D of Fig. 1, and, for clarity of presentation, self-loops (all labeled by $\{(1, 1), (2, 2), (3, 3)\}$), symmetric edges (e.g., there is also a $\{(1, 3)\}$ edge from n_3 to n_2), and transitive edges (e.g., there are also $\{(2, 2)\}$ edges between n_1 and n_6 and n_1 and n_7) have been suppressed. The reader is invited to verify that this graph is indeed a structural index for D , and that there is only one embedding α of the BGP of query Q of Example 1 into this structural index. In particular, α assigns the triple patterns $(?p_a, ?rel_{ab}, ?p_b)$ and $(?p_b, ?rel_{bc}, ?p_c)$ of Q to index nodes n_2 and n_3 , respectively. Whereas in the absence of structural information, these triple patterns individually can match any triple of D , α restricts their possible bindings to a small fraction of D , a significant reduction in search space for evaluating Q on D . \square

3.1 Query Processing with Structural Indexes

The following proposition (proof omitted) establishes in general this connection between query embeddings in a structural index and query evaluation on a dataset.

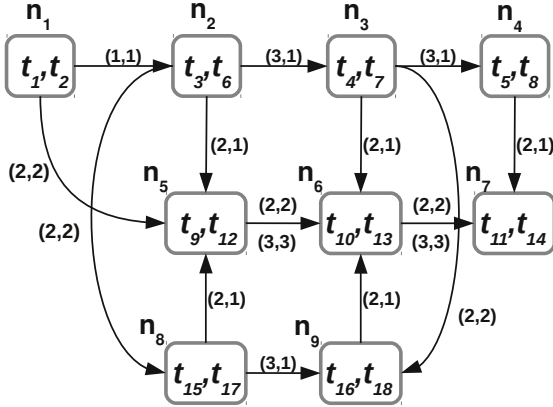


Fig. 2. A structural index for the RDF graph of Fig. 1. As described in Example 3, a few edges have been suppressed for clarity of presentation.

Proposition 1. Let $P = \{p_1, \dots, p_n\}$ be a BGP, let D be a dataset, let I be a structural index for D and let A be the set of all embeddings of P into I . Then

$$\begin{aligned} \llbracket P \rrbracket_D &= \bigcup_{\alpha \in A} \llbracket p_1 \rrbracket_{\alpha(p_1)} \bowtie \dots \bowtie \llbracket p_n \rrbracket_{\alpha(p_n)} \\ &= \llbracket p_1 \rrbracket_{\bigcup_{\alpha \in A} \alpha(p_1)} \bowtie \dots \bowtie \llbracket p_n \rrbracket_{\bigcup_{\alpha \in A} \alpha(p_n)} \end{aligned}$$

This proposition indicates two natural ways we can use a structural index I to alternatively compute $\llbracket P \rrbracket_D$:

- (M1). First compute the set A of all embeddings of P into I . Ideally, I is small enough so that finding these embeddings is computationally fast. For each $\alpha \in A$ we join $\llbracket p_1 \rrbracket_{\alpha(p_1)} \bowtie \dots \bowtie \llbracket p_n \rrbracket_{\alpha(p_n)}$, and add the result to the output. Note that, since $\alpha(p_i) \subseteq D$ for each $1 \leq n$, also $\llbracket p_i \rrbracket_{\alpha(p_i)} \subseteq \llbracket p_i \rrbracket_D$. Potentially, therefore, we compute joins on smaller relations than when computing $\llbracket p_1 \rrbracket_D \bowtie \dots \bowtie \llbracket p_n \rrbracket_D$. Nevertheless, we risk computing many such joins (as many as there are embeddings of P into I).
- (M2). To circumvent this problem, we can alternatively compute, for each i , the subset $D_i = \bigcup_{\alpha \in A} \alpha(p_i)$ of D , and then join $\llbracket p_1 \rrbracket_{D_1} \bowtie \dots \bowtie \llbracket p_n \rrbracket_{D_n}$. This requires computing the join only once, but on larger subsets of D .

We will empirically validate the effectiveness of these methods in Section 5.2.

3.2 Index Construction

A crucial assumption in the query processing strategies (M1) and (M2) outlined above is that the structural index I is small enough to efficiently compute embeddings on, yet detailed enough to ensure that candidate triples that cannot

participate in the required joins are pruned. Indeed, note that computing all embeddings of P in the trivial index I in which each block consists of a single triple will be as hard as computing the result of P on D itself. On the other hand, while it is trivial to compute all embeddings of P into the other trivial index J in which all triples are kept in a single block, we always have $\alpha(p_i) = D$ and hence no pruning is achieved.

We next outline a method for constructing structural indexes that are guaranteed to have *optimal* pruning power for the class of so-called *pure acyclic* BGPs, in the following sense.

Definition 3 (Pruning-optimal). *A structural index I for RDF graph D is pruning-optimal w.r.t. BGP P if, for every $p \in P$, we have $\pi_{vars(p)}\llbracket P \rrbracket_D = \llbracket p \rrbracket_{\bigcup_{\alpha \in A} \alpha(p)}$, where A is the set of all embeddings of P in I . Index I is pruning-optimal w.r.t. a class of BGPs \mathcal{C} if I is pruning-optimal w.r.t. every $P \in \mathcal{C}$.*

Note that the inclusion $\pi_{vars(p)}\llbracket P \rrbracket_D \subseteq \llbracket p \rrbracket_{\bigcup_{\alpha \in A} \alpha(p)}$ always holds due to Prop. [1](#). The converse inclusion does not hold in general, however.

Stated differently, pruning-optimality says that *every* element in $\llbracket p \rrbracket_{\alpha(p)}$ can be extended to a matching in $\llbracket P \rrbracket_D$, for every triple pattern $p \in P$ and every embedding α of P into I . Hence, when using Prop. [1](#) to compute $\llbracket P \rrbracket_D$ we indeed optimally prune each relation $\llbracket p_i \rrbracket_D$ to be joined.

As already mentioned, we will give a method for constructing structural indexes that are pruning-optimal w.r.t. the class of so-called *pure acyclic* BGPs. Here, purity and acyclicity are defined as follows.

Definition 4. *A BGP P is pure if it contains only variables, i.e., if $P \subseteq \mathcal{V}^3$.*

The restriction to pure BGPs is motivated by the following proposition, stating that pruning-optimal indexes do not always exist for non-pure BGPs. Intuitively, this is due to the fact that structural indexes only contain information about the joins that can be done on an RDF graph D , but do not contain any information about the universe values present in D . Since non-pure BGP *do* query for these universe values, structural indexes do not have enough information to ensure that for every α , every $p, q \in P$ and every $t \in \llbracket p \rrbracket_{\alpha(p)}$ there always exists a matching tuple $u \in \llbracket q \rrbracket_t$ that not only has the correct join type (i.e., $eqtp(p, q) \subseteq eqtp(t, u)$) but also fulfills the universe value constraints required by q (i.e., $u \in \llbracket q \rrbracket_D$).

Proposition 2. *Let p, q be distinct triple patterns with $eqtp(p, q) \neq \emptyset$ and $\{p, q\}$ not pure. There exists an RDF graph D such that any structural index I for D is not pruning-optimal w.r.t. P .*

The other restriction, acyclicity is a very well-known concept for relational select-project-join queries [\[2\]](#). Its adaption to BGP queries is as follows.

Definition 5 (Acyclicity). *A BGP P is acyclic if it has a join forest. A join forest for P is a forest F (in the graph-theoretical sense) whose set of nodes is exactly P such that, for each pair of triple patterns p and q in P that have variables in common the following two conditions hold:*

1. p and q belong to the same connected component of F ; and
2. all variables common to p and q occur in every triple pattern on the (unique) path in F from p to q .

The depth of F is the length of the longest path between any two nodes in F . The depth of an acyclic BGP P is the minimum depth of a join forest for P .

Recent analysis has illustrated that 99% of the BGP queries found in real-world SPARQL query logs are acyclic [16]. The class of acyclic BGPs is hence of practical relevance.

Similar to the way in which the concept of *graph bisimulation* (as used e.g., in modal logic and process calculi) is used to build structural indexes for semi-structured and XML databases and XPath-based query languages (e.g., [4, 9]), our pruning-optimal index is obtained by grouping triples that are equivalent under the following notion of *guarded simulation*.

Definition 6 (Guarded simulation). Let D be an RDF graph and let k be a natural number. We say that $u \in D$ simulates $t \in D$ guardedly up to depth k , denoted $t \preceq_k u$, if either (1) $k = 0$; or (2) if $k > 0$ there exists for every $t' \in D$ some $u' \in D$ such that $eqtp(t, u) \subseteq eqtp(t', u')$ and $t' \preceq_{k-1} u'$. We write $t \simeq_k u$ if $t \preceq_k u$ and $u \preceq_k t$. Finally, we write $t \simeq u$ if $t \simeq_k u$ for every k .

Although space constraints prohibit us from discussing the origin of the above definition in detail (cf. [8]), readers familiar with the notion of graph simulation may note that the above notion of guarded simulation is equivalent to the graph simulation (up to depth k) of the edge-labeled graph $G = (D, \{(t, \tau, u) \in D \times \mathcal{T} \times D \mid \tau \subseteq eqtp(t, u)\})$ to itself. It follows immediately that efficient main-memory algorithms for computing the relations \simeq_k and \simeq hence exist [10, 21].

Definition 7 (Simulation Index). The depth- k simulation index of RDF graph D , denoted $SIM_k(D)$, is the structural index $I = (N, E)$ for D such that

- N consists of the equivalence classes of \simeq_k , i.e., if we denote by $[t]_{\simeq_k}$ the set $\{u \in D \mid t \simeq_k u\}$ then $N = \{[t]_{\simeq_k} \mid t \in D\}$.
- $E = \{([t]_{\simeq_k}, \tau, [u]_{\simeq_k}) \mid t, u \in D, \tau = eqtp(t, u)\}$.

The simulation index of D , denoted $SIM(D)$ is defined similarly, but then using \simeq instead of \simeq_k .

The following proposition (proof omitted) shows that simulation indexes are pruning-optimal with respect to the class of pure acyclic BGPs.

Proposition 3. Let D be an RDF graph. $SIM(D)$ is pruning-optimal w.r.t. the class of pure acyclic BGPs. Moreover, $SIM_k(D)$ is pruning-optimal w.r.t. the class of pure acyclic BGPs of depth at most k , for each k .

Although pure BGPs are infrequent in practice, they are the only reasonable class of queries to couple queries with structural indexes from a theoretical point of view, as indicated by Prop. 2. This result hence shows that the $SIM(D)$ and $SIM_k(D)$ indexes allow one to take into account *precisely* the structural (join) information in the dataset.

4 Applying the Principles in Practice

In this section we discuss the design of SAINT-DB in which we have implemented the principles of Sec. 3. We start with a description of the triplestore upon which SAINT-DB is built.

RDF-3X. RDF-3X is a state-of-the-art, open source native RDF storage and retrieval system [15]. It is widely used by the research community and has, according to many previous studies, excellent query performance.

RDF-3X makes extensive use of B^+ -trees as its core underlying data structure. In particular, it stores all (s, p, o) triples of the RDF graph in a (compressed) clustered B^+ -tree in which the triples themselves act as search keys. This means that the triples are sorted lexicographically in the leaves of the B^+ -tree, which allows the conversion of triple patterns into efficient range scans. For example, to compute $[(jane, friendOf, ?x)]_D$ it suffices to search the B^+ -tree using the prefix search key $(jane, friendOf)$, and subsequently scan the relevant leaf pages to find all bindings for $?x$. RDF-3X actually employs this idea aggressively: to guarantee that not only triple patterns of the form $(jane, friendOf, ?x)$ can be answered by efficient range scans, but also triple patterns of the form $(?x, friendOf, lucy)$, $(jane, ?x, ?y)$, and so on, it maintains all six possible permutations of subject (S), predicate (P) and object (O) in six separate indexes (corresponding to the sort orders SPO, SOP, PSO, ...). Compression of the B^+ -tree leaf pages is used to minimize storage overhead. Since each possible way of lexicographically ordering the RDF graph (SPO, SOP, PSO, ...) is materialized in a separate index, joins can be answered using efficient merge-only joins during query processing (as opposed to the sort-merge joins that are normally required). We mention that in addition, RDF-3X also builds six so-called *aggregated* indexes and three so-called one-valued indexes, but refer to RDF-3X paper for full details [15].

The RDF-3X query optimizer uses detailed statistics (available, among others, in the aggregated and one-valued indexes) to efficiently generate bushy join orderings and physical query plans using an RDF-tailored cardinality and selectivity estimation algorithm [15].

SAINTDB. SAINT-DB represents structural indexes $I = (N, E)$ by assigning a unique integer $id(n) > 0$ to each index block $n \in N$. Both the partition N of D and D itself are represented by storing all triples $(s, p, o) \in D$ as *quads* of the form $(s, p, o, id([s, p, o]_I))$, where $id([s, p, o]_I)$ denotes the identifier of the index block containing (s, p, o) . The set of labeled edges E over N is represented by storing each $(m, \tau, n) \in E$ also as a quad $(id(m), \tau, id(n), 0)$, where the 0 in the fourth column allows us to distinguish quads that represent E -edges from quads representing D -triples. All of these quads are conceptually stored in a single quaternary relation.

Since SAINT-DB hence stores quads instead of triples, we have updated the complete RDF-3X infrastructure (B^+ -tree storage management and indexes, query optimization and compilation, query processing, data statistics, etc.) to reason about quads instead of triples. This effectively means that we save all permutations of subject (S), predicate (P), object (O), and block-id (B) (as well

as their aggregate and one-value versions) into B^+ -trees. As a consequence, it becomes possible to retrieve the set of all triples that (1) match a given triple pattern and (2) belong to a given index block by accessing the suitable B^+ -tree. For example, to compute $\llbracket(jane, friendOf, ?x)\rrbracket_n$ with n an index block we would search the *SPBO* B^+ -tree using the prefix key $(jane, friendOf, id(n))$ and find all bindings for $?x$ using a range scan over the corresponding leaves. This idea is easily extended to compute the set of all triples that (1) match a given triple pattern and (2) belong to a *set* of given index blocks. For example, to compute $\llbracket(jane, friendOf, ?x)\rrbracket_{n_1 \cup n_2}$ we would search and scan the *SPBO* B^+ -tree using the prefix key $(jane, friendOf, id(n_1))$; search and scan again using the $(jane, friendOf, id(n_2))$ prefix; and merge the two results to produce a sorted list of bindings for $?x$.

Adding Predicates to the Index. During our experiments we have noticed that the set A of all embeddings of BGP P into I frequently contains embeddings α that cannot contribute to $\llbracket P \rrbracket_D$ due to the fact that, for some triple pattern $p \in P$, there is actually no triple in $\alpha(p)$ that mentions the constants required in p . To remedy this deficiency while keeping the index small, we store, for each index block $n \in N$ the set $preds(n)$ of predicates mentioned, $preds(n) := \{pred \mid (s, pred, o) \in n\}$. Since the set of all predicates used in an RDF graph is typically quite small, each $preds(n)$ is also small and efficient to represent. By storing $preds(n)$ in the index we can then remove from A all embeddings α for which there is some triple pattern $(s, p, o) \in P$ with p a constant and $p \notin preds(n)$. Let us denote this reduced set of embeddings by A' .

SAINTDB Query Processing. We have implemented the following three query processing strategies in SAINT-DB. In each of these strategies, we first compute the reduced set A' of embeddings of the P into I , as described above.

The first two strategies corresponds to the methods (M1) and (M2) described in Sec. 3 where embeddings are only taken from A' and where the sets $\llbracket p_i \rrbracket_{\alpha(p_i)}$ and $\llbracket p_i \rrbracket_{\bigcup_{\alpha \in A'} \alpha(p)}$ are computed using the suitable B^+ -tree range scans, as outlined above. No join ordering is attempted; all joins are executed in the same order as when RDF-3X computes $\llbracket p_1 \rrbracket_D \bowtie \dots \bowtie \llbracket p_n \rrbracket_D$. Since the operands of the (M1) and (M2) may be smaller than the corresponding operands of the RDF-3X join, this order may not be optimal.

The third strategy, denoted (M3) in what follows, is a variant of (M2) that employs full quad-based query optimization to reach a suitable physical query plan. In particular, (M3) uses the statistics to estimate the cardinality of both $\llbracket p \rrbracket_D$ and $\llbracket p \rrbracket_{\bigcup_{\alpha \in A'} \alpha(p)}$, for each $p \in P$. In the event that the set $\{\alpha(p) \mid \alpha \in A\}$ contains multiple index blocks (and we hence have to do multiple B^+ -tree scans and merge the results) it uses these cardinalities to check that the costs for loading and merging $\llbracket p \rrbracket_{\bigcup_{\alpha \in A} \alpha(p)}$ is lower than the cost of simply loading $\llbracket p \rrbracket_D$. If not, the structural index information is thrown away, and $\llbracket p \rrbracket_D$ will be executed (but only for the triple pattern under consideration in isolation). Once it has determined, for each triple pattern, whether the available structural index embeddings should be used, it computes a bushy join ordering and physical plan, based on the quad cardinality statistics.

5 Experimental Validation

5.1 Experimental Setup

We have implemented SAINT-DB upon RDF-3X version 0.36². All experiments described in this section have been run on an Intel Core i7 (quad core, 3.06 GHz, 8MB cache) workstation with 8GB main memory and a three-disk RAID 5 array (750GB, 7200rpm, 32MB cache) running 64-bit Ubuntu Linux.

Our performance indicator is the number of I/O read requests issued by SAINT-DB and RDF-3X, measured by counting the number of calls to the buffer manager’s `readPage` function. Thereby, our measurements are independent of the page buffering strategies of the system. Since SAINT-DB currently does not yet feature compression of the B^+ -tree leaf pages, we have also turned off leaf compression in RDF-3X for fairness of comparison. During all of our experiments the structural indexes were small enough to load and keep in main memory. The computation of the set of all embeddings into the index hence does not incur any I/O read requests, and is not included in the figures mentioned.

Datasets, Queries, and Indexes. We have tested SAINT-DB on two synthetic datasets and one real-world dataset. The first synthetic dataset, denoted CHAIN, is used to demonstrate the ideal that SAINT-DB can achieve on highly graph-structured and repetitive data. It contains chains of triples of the form $(x_1, y_1, x_2), (x_2, y_2, x_3), \dots, (x_n, y_n, x_{n+1})$, with chain length n ranging from 3 to 50. Each chain is repeated 1000 times and CHAIN includes around 1 million triples in total. The full simulation index $\text{SIM}(\text{CHAIN})$ has been generated accordingly, and consists of 1316 index blocks, each consisting of 1000 triples. On CHAIN we run queries that also have a similar chain-shaped style $(?x_1, ?y_1, ?x_2), (?x_2, ?y_2, ?x_3), \dots, (?x_n, ?y_n, ?x_{n+1})$, with n varying from 4 to 7.

The second synthetic dataset, denoted LUBM, is generated by the Lehigh University Benchmark data generator [11] and contains approximately 2 million triples. For this dataset, we computed the depth-2 simulation index $\text{SIM}_2(\text{LUBM})$, which consists of 222 index blocks. Index blocks have varying cardinalities, containing as little as 1 triple to as many 190,000 triples.

The real-world RDF dataset, denoted SOUTHAMPTON, is published by the University of Southampton³. It contains approximately 4 million triples. For this dataset, we also computed the depth-2 simulation index $\text{SIM}_2(\text{SOUTHAMPTON})$, which consists of 380 index blocks. Index blocks have varying cardinalities, containing as little as 1 triple to as many 10^6 triples.

For all datasets the indexes in their current non-specialized form require only a few megabytes and therefore can be kept in main memory. A specialized in-memory representation could easily further reduce this footprint. The detailed description of the queries used can be found online⁴. In the rest of this section we denote queries related to the LUBM dataset as $L1, \dots, L16$, and those related to SOUTHAMPTON as $S1, \dots, S7$.

² <http://code.google.com/p/rdf3x/>

³ <http://data.southampton.ac.uk/>

⁴ <http://www.win.tue.nl/~yluo/saintdb/>

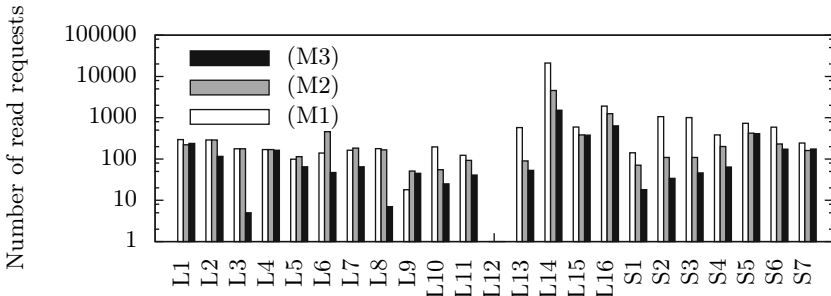


Fig. 3. Number of read requests for different query processing strategies

5.2 Experimental Analysis

We first examine the different query processing strategies implemented in SAINT-DB to exploit the structural index, and then compare the performance of SAINT-DB to that of RDF-3X. In the following, we assume that the database has loaded the partition blocks into main memory, and ensure that all queries are executed on a cold cache/buffer. The embeddings of the queries into the structural index can therefore be efficiently computed, and are available to the query optimizer.

Query Processing Strategies. Fig. 3 shows the number of I/O read requests issued to the buffer manager during query evaluation by each of the three different query processing strategies (M1), (M2), and (M3) introduced in Sec. 4.

As a general observation, (M1) requires more reads from the database than (M2), which in turn requires more reads than (M3). Conceptually, (M1) executes a different query for each embedding of the original BGP into the structural index, while (M2) executes the same queries in parallel, sharing evaluation costs. (M3) differs from both (M1) and (M2) by exploiting not only the structural index but also the selectivity of particular triple patterns. A lower number of reads can therefore be achieved, by accessing directly the relevant information when very specific triple patterns are issued. For example, L_4 asks for every undergraduate student with a single triple pattern. Hence, no structural information is available while the triple pattern itself selects the right data.

While this observation explains the general behavior, a more detailed analysis provides more insights. First, query L_{12} does not require any read. Indeed, this query does not produce any result, and the absence of results is identified at the index level: no embedding of the BGP of this query exist into the structural index. Second, queries L_6 and L_9 show that strategy (M1) can perform better than (M2) and (M3). This is due to different join orderings, which, along with sideways information passing [14], causes scans to skip different data sets. The current plan generation cannot identify these differences, as sideways information passing depends on runtime data.

SAINT-DB vs RDF-3X. We now turn to a comparison of the query evaluation costs of SAINT-DB and RDF-3X. For this comparison we use the (M3) strategy in SAINT-DB, given our observations above on the performance of this strategy.

Table 1. Read requests for SAINT-DB and RDF-3X on the CHAIN dataset. The columns denote the length of the chain in the query. Speed-up is the ratio of read requests of RDF-3X over those of SAINT-DB.

	4	5	6	7
SAINT-DB	306	350	393	438
RDF-3X	3864	4799	5734	6669
<i>Speed-up</i>	12.63	13.71	14.59	15.23

Table 2. Read requests for SAINT-DB and RDF-3X on the LUBM and SOUTHAMPTON datasets. Speed-up is the ratio of read requests of RDF-3X over those of SAINT-DB.

	C1			C2				C3				
	L2	L3	L4	L9	S1	S2	S4	L1	L5	L6	L7	L8
SAINT-DB	116	5	163	18	18	36	64	238	39	47	38	7
RDF-3X	89	5	123	12	16	35	53	194	132	39	268	7
<i>Speed-up</i>	0.77	1.00	0.75	0.67	0.89	0.97	0.83	0.82	3.38	0.83	7.05	1.00

	C3										
	L10	L11	L12	L13	L14	L15	L16	S3	S5	S6	S7
SAINT-DB	25	41	0	53	1519	352	288	48	410	173	175
RDF-3X	21	30	281	109	2668	2178	1224	33	424	316	236
<i>Speed-up</i>	0.84	0.73	∞	2.06	1.76	6.19	4.25	0.69	1.03	1.83	1.35

Table 1 shows the performance of RDF-3X and SAINT-DB on the CHAIN dataset. We see that SAINT-DB requires over 10 times less I/Os up compared to RDF-3X, and this reduction in I/Os increases as query length increases. This is due to the rich structures inside the data set and the queries. The structural index can hence significantly eliminate the search space of the later index scans. These results demonstrate the tremendous potential of structural indexing over value-based indexing.

Table 2 shows the I/O costs of RDF-3X and SAINT-DB, for the LUBM and SOUTHAMPTON datasets. We have grouped queries into three categories:

- (C1) Queries without structure. These queries consist of a single triple pattern, and hence do not exhibit any structural information to be exploited.
- (C2) Structured queries over highly specific information. These queries have many triple patterns, and at least one triple pattern is very selective.
- (C3) Structured queries. These queries have many triple patterns, with rich structural information.

By leveraging exhaustive value-based indexes and various optimization strategies, RDF-3X can efficiently answer queries in category C1 and C2. In particular, for C2 the technique of *sideways information passing* [14] allows efficient computation of bindings in RDF-3X for the less selective patterns. Hence, as we can expect, the structural indexes of SAINT-DB provide no advantage. Nevertheless, even though these queries represent the worst-case scenario for structural indexes, SAINT-DB generally exhibits comparable query evaluation costs.

Further study is nevertheless warranted to bridge the gap between value-based and structure-based indexing for such query types (e.g., compression techniques in index blocks, to offset the overhead introduced by moving from triples to quads).

For the queries of category C3, we see that structural information does increase selectivity significantly. For example, queries *L5*, *L7*, *L14*, *L15*, and *L16* do benefit from the increased selectivity, with SAINT-DB having as little as 15% of the query evaluation costs of RDF-3X. The query benefiting the most from structural information is *L12*. The result is detected as empty at the structural index level in SAINT-DB, and hence no read request is issued. For this same query, RDF-3X needs to perform a number of joins to find the same empty result. The structural-approach avoids these I/Os completely.

This initial empirical study indicates that there are indeed general situations where SAINT-DB can clearly leverage structural information for significant reduction in query evaluation costs. Furthermore, for queries without significant structure, or with highly selective triple patterns, SAINT-DB is competitive with RDF-3X. Our next steps in this study are a finer analysis of query categories, and their appropriate indexing and query evaluation strategies in SAINT-DB.

6 Concluding Remarks

In this paper, we have presented the first results towards triple-based structural indexing for RDF graphs. Our approach is grounded in a formal coupling between practical fragments of SPARQL and structural characterizations of their expressive power. An initial empirical validation of the approach shows that it is possible and profitable to augment current value-based indexing solutions with structural indexes for efficient RDF data management.

In this first phase of the SAINT-DB investigations, we have focused primarily on the formal framework and design principles. We are currently shifting our focus to a deeper investigation into the engineering principles and infrastructure necessary to put our framework into practice. Some basic issues for further study in this direction include: alternates to the B^+ -tree data structure for physical storage and access of indexes and data sets; more sophisticated optimization and query processing solutions for reasoning over both the index and data graphs; efficient external memory computation and maintenance of indexes; and, extensions to richer fragments of SPARQL, e.g., with the OPTIONAL and UNION constructs.

Acknowledgments. We thank Paul De Bra for his critical feedback on these investigations. The research of FP is supported by a FNRS/FRIA scholarship. The research of SV is supported by the OSCB project funded by the Brussels Capital Region. The research of GF, JH, and YL is supported by the Netherlands Organization for Scientific Research (NWO).

References

1. Abadi, D., et al.: SW-Store: a vertically partitioned DBMS for semantic web data management. *VLDB J.* 18, 385–406 (2009)
2. Abiteboul, S., Hull, R., Vianu, V.: *Foundations of Databases*. Addison-Wesley (1995)
3. Arias, M., Fernández, J.D., Martínez-Prieto, M.A., de la Fuente, P.: An empirical study of real-world SPARQL queries. In: USEWOD (2011)
4. Arion, A., Bonifati, A., Manolescu, I., Pugliese, A.: Path summaries and path partitioning in modern XML databases. *WWW* 11(1), 117–151 (2008)
5. Brenes Barahona, S.: *Structural summaries for efficient XML query processing*. PhD thesis, Indiana University (2011)
6. Bröcheler, M., Pugliese, A., Subrahmanian, V.S.: DOGMA: A Disk-Oriented Graph Matching Algorithm for RDF Databases. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) *ISWC 2009*. LNCS, vol. 5823, pp. 97–113. Springer, Heidelberg (2009)
7. Fletcher, G.H.L., Beck, P.W.: Scalable indexing of RDF graphs for efficient join processing. In: CIKM, Hong Kong, pp. 1513–1516 (2009)
8. Fletcher, G.H.L., Hidders, J., Vansummeren, S., Luo, Y., Picalausa, F., De Bra, P.: On guarded simulations and acyclic first-order languages. In: *DBPL*, Seattle (2011)
9. Fletcher, G.H.L., Van Gucht, D., Wu, Y., Gyssens, M., Brenes, S., Paredaens, J.: A methodology for coupling fragments of XPath with structural indexes for XML documents. *Information Systems* 34(7), 657–670 (2009)
10. Gentilini, R., Piazza, C., Policriti, A.: From bisimulation to simulation: Coarsest partition problems. *J. Autom. Reasoning* 31(1), 73–103 (2003)
11. Guo, Y., Pan, Z., Hefflin, J.: LUBM: A benchmark for OWL knowledge base systems. *J. Web Sem.* 3(2-3), 158 (2005)
12. Luo, Y., Picalausa, F., Fletcher, G.H.L., Hidders, J., Vansummeren, S.: Storing and indexing massive rdf datasets. In: De Virgilio, R., et al. (eds.) *Semantic Search over the Web, Data-Centric Systems and Applications*, pp. 29–58. Springer, Heidelberg (2012)
13. Milo, T., Suciu, D.: Index Structures for Path Expressions. In: Beeri, C., Bruneman, P. (eds.) *ICDT 1999*. LNCS, vol. 1540, pp. 277–295. Springer, Heidelberg (1998)
14. Neumann, T., Weikum, G.: Scalable join processing on very large RDF graphs. In: *SIGMOD*, pp. 627–640 (2009)
15. Neumann, T., Weikum, G.: The RDF-3X engine for scalable management of RDF data. *VLDB J.* 19(1), 91–113 (2010)
16. Picalausa, F., Vansummeren, S.: What are real SPARQL queries like? In: *Proceedings of the International Workshop on Semantic Web Information Management, SWIM 2011*, pp. 7:1–7:6. ACM, New York (2011)
17. Prud’hommeaux, E., Seaborne, A.: *SPARQL query language for RDF*. Technical report, W3C Recommendation (2008)
18. Sidiropoulos, L., et al.: Column-store support for RDF data management: not all swans are white. *Proc. VLDB Endow.* 1(2), 1553–1563 (2008)
19. Tran, T., Ladwig, G.: Structure index for RDF data. In: *Workshop on Semantic Data Management, SemData@ VLDB* (2010)
20. Udea, O., Pugliese, A., Subrahmanian, V.S.: GRIN: A graph based RDF index. In: *AAAI*, Vancouver, B.C., pp. 1465–1470 (2007)

21. van Glabbeek, R.J., Ploeger, B.: Correcting a Space-Efficient Simulation Algorithm. In: Gupta, A., Malik, S. (eds.) CAV 2008. LNCS, vol. 5123, pp. 517–529. Springer, Heidelberg (2008)
22. Weiss, C., Karras, P., Bernstein, A.: Hexastore: Sextuple Indexing for Semantic Web Data Management. In: VLDB, Auckland, New Zealand (2008)
23. Wylot, M., Pont, J., Wisniewski, M., Cudré-Mauroux, P.: dipLODocus[RDF]—Short and Long-Tail RDF Analytics for Massive Webs of 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. 778–793. Springer, Heidelberg (2011)
24. Zou, L., Mo, J., Chen, L., Özsu, M.T., Zhao, D.: gStore: Answering SPARQL queries via subgraph matching. Proc. VLDB Endow. 4(8), 482–493 (2011)

Domain Specific Data Retrieval on the Semantic Web

Tuukka Ruotsalo^{1,2,3}

¹ School of Information, University of California, Berkeley, USA

² Department of Media Technology, Aalto University, Finland

³ Helsinki Institute for Information Technology (HIIT), Finland

tuukka.ruotsalo@aalto.fi

Abstract. The Web content increasingly consists of structured domain specific data published in the Linked Open Data (LOD) cloud. Data collections in this cloud are by definition from different domains and indexed with domain specific ontologies and schemas. Such data requires retrieval methods that are effective for domain specific collections annotated with semantic structure. Unlike previous research, we introduce a retrieval framework based on the well known vector space model of information retrieval to fully support retrieval of Semantic Web data described in the Resource Description Framework (RDF) language. We propose an indexing structure, a ranking method, and a way to incorporate reasoning and query expansion in the framework. We evaluate the approach in ad-hoc retrieval using two domain specific data collections. Compared to a baseline, where no reasoning or query expansion is used, experimental results show up to 76% improvement when an optimal combination of reasoning and query expansion is used.

1 Introduction

Search engines have revolutionized the way we search and fetch information by being able to automatically locate documents on the Web. Search engines are mostly used to locate text documents that match queries expressed as a set of keywords. Recently, the document centric Web has been complemented with structured metadata, such as the Linked Open Data cloud (LOD) [3]. In such datasets structured and semantic data descriptions complement the current Internet infrastructure through the use of machine understandable information provided as annotations [2]. Annotations are produced manually in many organizations, but automatic annotation has also become mature enough to work on Web scale [13]. As a result, we are witnessing increasing amount of structured data published on the Web.

Standards such as the RDF(S) [5] and publishing practices for linked data have enabled seamless access to structured Web data, but the underlying collections remain indexed using domain specific ontologies and schemas. In fact, such domain specific structure is the underlying element empowering the Semantic Web. For example, the data from cultural heritage data providers is very different from the data by scientific literature publishers, indexed with different vocabularies, and in the end, serving different information needs. As a result, different data collections are being published as a linked open data and accessed on the Web, but each individual publisher can decide about the semantics used to annotate the particular data collection. This imposes specific challenges for retrieval methods operating on such dataspace:

1. *Structured object data.* Search is targeted to objects or entities that are increasingly described using a combination of structured information and free text descriptions. For example, a tourist attraction could be described with information about the location of the site as coordinate data, the categorization of the site through references to a thesauri or an ontology, and the description of the attraction in free text format.
2. *Recall orientation.* A subset of the linked data cloud identified relevant for a specific application is often limited in size. Data collections are in hundreds of thousands or millions as opposite to billions as in conventional Web search. This favors recall oriented retrieval methods.
3. *Semantic gap between search and indexing vocabulary.* Objects originate from domain specific curated collections and are described using expert vocabulary. For example, a user searching for scientific objects inside a museum could be interested on spheres, galvanometers, and optical instruments, but could use terms "science" and "object" to express her information need.

To address the former challenges, we propose an extension of the Vector Space Model (VSM) [15] adapted to the RDF data model. Unlike in previous approaches [8,6,16,10,11,7], in our extension the indexing is based on RDF triples instead of individual concepts detected from text documents. The novelty of our model is that we use RDF triples as the basis for our indexing and ranking models instead of using ontologies only to expand individual terms in queries or text document indexing. This has only been addressed in [4], where horizontal indices were used to index RDF data. While similar in nature, in addition, we compare different query types, query complexity levels, and query expansion levels for the triple-based model. We also consider more complex queries than keyword queries as used in the previous work. In our approach, the queries can be any combination of keywords, triples or resources. In addition, the effect of query and document expansion, that allow background knowledge to be used as basis to reduce data sparsity and enable semantic indexing, are key contributions in our study.

We evaluate our approach in domain specific data retrieval on cultural heritage data collections and show that our adaptation of the VSM, combined with reasoning and correct query expansion strategy, yields to superior retrieval performance with an increase of 76% in mean average precision compared to a baseline approach. The rest of the paper is structured as follows. In section 2 we present the retrieval framework including indexing, retrieval and query expansion methods. In section 3 we explain the experimental setup, data and evaluation measures. Section 4 presents the results. Finally, we conclude with discussion, related work and future research directions.

2 Retrieval Framework

We use a retrieval framework based on the VSM and extend it to utilize RDF triples as indexing features. We show how indexing can be done for RDF triples, cosine similarity computed over such data representation, and how reasoning and query expansion can be incorporated in to the retrieval framework.

2.1 Data Representation

We start with a retrieval method based on the well known Vector Space Model of information retrieval [15]. We use metadata expressed as ontology-based annotations and utilize RDF as a representation language. RDF describes data as triples, where each triple value can be either a resource R or literal L . The feasibility of the index can be problematic in terms of the size of the triple-space if the triples would be directly used as indexing features. Using the pure VSM of information retrieval would cause the dimensionality of the document representation to be vectors that have occurrences for every deduced triple. The maximum dimensionality being the number of possible triples on the domain $T \in R \times R \times (R \cup L)$. It is well known that high dimensionality often causes problems in similarity measurements and has been recognized to be problematic in ontology-based search [61]. This would hurt the performance of the VSM, because many of the matching concepts would be the same in the tail of super concepts, i.e. almost all documents would be indexed using the triples consisting of resources appearing in the upper levels of the ontology hierarchies.

We reformulate the indexing of the documents and the triples in the deductive closure of their annotations as vectors describing occurrences of each triple given the property of the triple. Splitting the vector space based on property is not a new idea, but has been recently used in RDF indexing [24]. An intuition behind this is that properties often specify the point of view to the entity. For example, annotating *Europe* as a manufacturing place or subject matter should lead to completely different weighting of the resource, depending on the commonality of *Europe* as a subject matter or as a manufacturing place. In addition, properties are not expected to be used as query terms alone, but only combined with either subjects or objects of the triple. For example, it is unlikely that a user would express her information need by inserting a query to return all documents with *dc:subject* in the annotations. However, a user could construct a query that would request all documents with *dc:subject* having a value *Europe*. Literals are treated separately from concepts. We tokenize literals to words and stem them using the Porter stemming method. After this they are stored in the same vectors as the concepts. In practice, the data is often described using a schema, where the subject of the triple is the identifier for the entity being described, as in the data used in our experiments. However, our indexing strategy enables indexing of arbitrary RDF graphs.

Accessing the correct index for each vector space fast in the query phase requires an external index. For this purpose we define a posting list that maps the index of the correct vector space to the query. We propose a model over possible vector spaces, first one for every possible property, and two additional vector spaces for subject and object. From now on we refer to these actually indexed subjects and objects as concepts to avoid mixing these with the subject of an RDF triple. Every concept is indexed in a vector space that defines the occurrence of the concept in an annotation of a specific entity. These vector spaces are referred as y and they form a set of vector spaces Y with a length x , i.e. $Y_x = \{y_1, \dots, y_x\}$.

This indexing strategy requires a large number of vector spaces, but the triple dimension of each matrix is lower because the maximum term length k for triples is the number of resources and literals R , and for the document dimension only the

documents that have triples in the particular vector space are indexed. This avoids the high dimensionality problem when computing similarity estimates.

2.2 Weighting

The purpose of the indexing strategy is not only to reduce the dimensionality to make computation faster, but also to enable more accurate weighting by avoiding the problems caused by the high dimensionality. Intuitively, some of the triples are likely to be much less relevant for the ranking than others. For example, matching a query only based on a triple $\langle rdf:Resource, rdf:Resource, rdf:Resource \rangle$ will lead to a match to all documents, but is meaningless for search purposes. On the other hand, a resource Helsinki, should be matched to all documents indexed with resource Helsinki, but also to the documents indexed with Europe, because they belong into the same deductive closure, but with smaller weight. For this purpose we use *tf-idf* weighting over the resources within a specific vector space. In normalized form *tf* is:

$$tf_{i,j} = \left(\frac{N_{i,j}}{\sum_k N_{k,j}} \right)^{\frac{1}{2}}, \quad (1)$$

where $N_{i,j}$ is the number of times a resource i is mentioned in the vector space of document j and $\sum_k N_{k,j}$ is the sum of the number of occurrences of all resources of the document j . In a similar way, inverse document frequency *idf* is defined as:

$$idf_i = 1 + \log\left(\frac{N}{n_i + 1}\right), \quad (2)$$

where n_i is the number of documents, where the resource i appears within the specific vector space and N is the total number of documents in the system. The weight of an individual resource in a specific vector space is given by:

$$w_{i,j} = tf_{i,j} \cdot idf_i. \quad (3)$$

The *tf-idf* effect in triple-space is achieved based on the annotation mass on resources, but also through reasoning. For example, the resource Europe is likely to have much more occurrences in the index than the resource Finland, since the index contains the deductive closures of the triples from annotations using resource identifiers also for other European countries. This makes the *idf* value for resource Finland higher than for resource Europe. A document annotated with resources Germany, France and Finland would increase the *tf* value for the resource Europe, because through deductive reasoning Germany, France and Finland are a part of Europe. Naturally, the *tf* could also be higher in case the document is annotated with several occurrences of the same resource, for example as a result of automatic annotation procedure based on text analysis [13].

2.3 Ranking

In the vector model the triple vectors can be used to compute the degree of similarity between each document d stored in the system and the query q . The vector model

evaluates the similarity between the vector representing an individual document V_{d_j} and a query V_q . We reformulate the cosine similarity to take into account a set of vector spaces, one for each possible combination of triples given the models $y \in Y$ as opposite to the classic VSM that would use only one vector space for all features. For this purpose, we adopt the modified cosine similarity ranking formula used in Apache Lucene open source search engine¹, where the normalization based on Euclidean distance is replaced with a length norm and a coord-factor. The length norm is computed as:

$$\ln(V_{d_j, y}) = \frac{1}{\sqrt{nf}}, \quad (4)$$

where nf is the number of features present used to index the document d_j in the vector space y under interest. The coord-factor is computed as:

$$cf(q, d_j) = \frac{mf}{k}, \quad (5)$$

where mf is the number of matching features in all vector spaces for document d_j and query q and k is the total number of features in the query.

In our use case, these have two clear advantages compared to the classic cosine similarity. First, the use of the length norm gives more value to documents with less triple occurrences within the vector space under interest. In our case this means that documents annotated with less triples within a particular vector space y get relatively higher similarity score. This is intuitive, because the knowledge-base could contain manually annotated documents with only few triples and automatically annotated documents with dozens of triples. In addition, some vector spaces can end up having more triples, as a result of reasoning or more intense annotation, than others. The number of matching features in queries also should increase the similarity of the query and document. This effect is captured by the coord-factor. We can now write the similarity as:

$$sim(q, d_j) = cf(q, d_j) \cdot \sum_{y=1}^x \sum_{i=1}^k (w_{i, y_j} \cdot \ln(V_{y_{d_j}})), \quad (6)$$

where the dot product of the vectors now determines the weight $w_{i, j}$ and is computed across all vector spaces y . In this way the ranking formula enables several vector spaces to represent a single document because length norm is computed for each vector space separately. This can be directly used to operate with our triple space indexing.

The model approximates the importance of all the different combinations of $y \in Y$ separately. Intuitively, this is a coherent approach: the importance of a concept in the domain is dependent on the use of the concept in a triple context. Note that our approach does not normalize across the vector spaces. This favors matches in several vector space instead of a number of matches in a single vector space. For example, a query with several triples with the property *dc:subject* and a single triple with the property *dc:creator* would favor queries that have both *dc:subject* and *dc:creator* present over queries that would have matches only for one of the properties.

¹ The features of the similarity computation that are not used in our method and experiments are omitted. The full description of the original ranking formula can be found at <http://lucene.apache.org/>

2.4 Reasoning and Query Expansion

The adaptation of the vector space model that we presented in the earlier section assumes the existence of document vectors that can be then stored in separate vector spaces. RDF(S) semantics enable deductive reasoning on the triple space. Using such information in the indexing phase is often called document expansion. This means that the document vectors are constructed based on the triple-space resulting from a deductive reasoning process.

For example, an annotation with an object Paris, could be predicated by different properties. One document could be created in Paris while another document could have Paris as a subject matter. Through deductive reasoning both of these annotation triples are deduced to a triple, where the property pointing to the concept Paris is *rdf:Resource*. In a similar way, the concept Paris can be deduced through subsumption reasoning to France, Europe, and so on.

If a search engine receives a query about Paris, it should not matter for the search engine whether the user is interested in Paris in the role of subject matter or place of creation. Therefore, the search engine should rank these cases equally based on only the information that the documents are somehow related to Paris. In other words, based on the triple, where the property is *rdf:Resource*. On the other hand, if the user specifies an interest in Paris as a subject matter, the documents annotated in such way can be ranked higher by matching them to a vector space of subject matters. This functionality is already enabled using the vector space model for triple space by indexing deductive closures along with the original triples.

Another way to improve the accuracy of the method is ontology-based query expansion. While deductive reasoning provides logical deduction based on the relations available in the ontologies, the user can be interested also in other related documents. For example, users interested in landscape paintings, could also be interested on seascape paintings, landscape photographs and so on. These can be related in the ontology further away or with different relationships that are included in the standard RDF(S) reasoning.

Ontologies can be very unbalanced and depending on the concepts used in the annotation, different level of query expansion may be necessary. For example, a document annotated with a concept Buildings may already be general enough and matches to many types of buildings, while a document annotated with the concept Churches might indicate user's interest, not only on churches, but also other types of religious buildings.

Measuring a concept to be semantically close to another concept, and therefore a good candidate for the query expansion, can be approximated using its position in the ontological hierarchy [14]. The more specific the concept is, more expansion can be allowed. We use the Wu-Palmer measure to measure the importance of a resource (subject, predicate, and object separately) given the original resource in the query triple. Formally, the Wu-Palmer measure for resources c and c' is:

$$rel_{WP}(c, c') = \frac{2l(s(c, c'), r)}{l(c, s(c, c')) + l(c', s(c, c')) + 2l(s(c, c'), r)}, \quad (7)$$

where $l(c, c')$ is a function that returns the smallest number of nodes on the path connecting c and c' (including c and c' themselves), $s(c, c')$ is a function that returns the

lowest common super-resource of resources c and c' , and r is the root resource of the ontology.

The resources having a Wu-Palmer value above a certain threshold are selected for query expansion. We construct all the possible triples that are possible based on the resources determined by the Wu-Palmer measure and select the most general triples as the expanded triples that are used in the actual similarity computation. This means that all subjects, properties, and objects of any triple in the query are included by using all permutations of the resources in these resulting sets and the most general combination is selected. By the most general combination, we mean triple that has the longest distance in terms of subsumption from the original triple in terms of the expanded subject, predicate and object, each measured individually. This also removes possible redundancy of the original query triples, such as inclusion of triples.

In case other relations are used in the expansion, all of the triples are included. In other words, we include only the most general case in terms of subsumption, but include related terms as new triples. The rationale behind including only the most general triple is that including all possible super-triples could lead to a substantial amount of matching triples and may hurt the accuracy of the similarity computation.

The Wu-Palmer measure can be used to dynamically control the query expansion level towards an index of concepts that form a tree. Such a tree can be constructed in many different ways. A trivial case is to use only subsumption hierarchies, a semantically coherent taxonomy of concepts. However, ontologies enable also other relations to be used in query expansion. We refer different combinations of such relations as the query expansion strategy.

We investigate the following query expansion strategies: related terms only, subsumption only, full expansion. Related terms only strategy means that a semantic clique is formed based on the nodes directly related to the concept being expanded (distance of arcs is one), but no subsumption reasoning is used. Subsumption only strategy means that the query is expanded using transitive reasoning in subsumption hierarchies. This means that additional query expansion to other concepts than those in the deductive closure can be done only using subsumption hierarchies. Full expansion means that both, subsumption and related terms are used in expansion and the tree index is built using subsumption relations and related terms of each concept achieved through subsumption. Related terms are not treated as transitive.

For example, using only the subsumption hierarchies, we could deduce the information that the concept "landscape paintings" is related to its superconcept "paintings" and through that to the concept "seascape paintings", because they have a common superconcept. Using the full expansion we could obtain an additional information that "seascape paintings" is further related to "seascapes", "marinas" and so on.

3 Experiments

We conducted a set of laboratory experiments to determine the retrieval performance of the method and the effect of different indexing and query expansion strategies.

3.1 Method Variants

We created different variants of the VSM method by varying indexing strategies and query expansion levels. These were used to study the effect of the different combinations to the retrieval performance. The performance of all different indexing strategies was measured separately: related terms only, subsumption only, and full expansion. All of the strategies were then measured using different levels of query expansion by varying the Wu-Palmer measure from 1.0 to 0.1. All of the strategies were implemented on top of the triple-space index.

3.2 Data

We used a dataset in the domain of cultural heritage, where the documents have high quality annotations. The dataset consists of documents that describe museum items, including artwork, fine arts and scientific instruments, and points of interest, such as visiting locations, statues, and museums. The data was obtained from the Museo Galileo in Florence, Italy, and from the Heritage Malta. The document annotations utilize the Dublin Core properties and required extensions for the cultural heritage domain, such as material, object type, and place of creation of the document described. An example annotation of a document describing a scientific instrument from the Museo Galileo is described in Figure 1.

```
<dc:identifier> <urn:imss:instrument:402015> .
<sm:physicalLocation> <http://www.imss.fi.it/> .
<dc:title> "Horizontal dial" .
<dc:subject> "Measuring time" .
<dc:description> "Sundial, complete with gnomon..." .
<dc:subject> <aat:300054534> . (Astronomy)
<sm:dateOfCreation> <sm:time_1501_1600> . (16th Century)
<sm:material> <aat:300010946> . (Gilt Brass)
<sm:objectType> <aat:300041614> . (Sundial)
<sm:placeOfCreation> <tgn:7000084> (Germany)
<sm:processesAndTechniques> <aat:300053789> . (Gilding)
<dc:terms/isPartOf> "Medici collections" .
<rdf:type> <sm:Instrument> .
```

Fig. 1. An example of the data used in the experiments. Subjects of the triples are all identifiers of the resource being describes and are therefore omitted. Description is shortened.

The documents are indexed with RDF(S) versions of Getty Vocabularies². The RDF(S) versions of the Getty Vocabularies are lightweight ontologies that are transformed to RDF(S) from the original vocabularies, where concepts are organized in subsumption hierarchies and have related term relations. Geographical instances are structured in meronymical hierarchies that represent geographical inclusion. Temporal data is described using a hand crafted ontology that has concepts for each year, decade,

² http://www.getty.edu/research/conducting_research/vocabularies/

century, and millennium organized in a hierarchy. Literal values are indexed in the VSM as Porter stemmed tokenized words.

3.3 Queries and Relevance Assessments

The query set consists of 40 queries that were defined by domain experts in the same museums where the datasets were curated. Figure 2 shows two example queries, one for astronomers and subject matter optics, and another for physicist Leopoldo Nobili and subject matter of galvanometers, batteries and electrical engineering. Relevance assessments corresponding to the query set were provided for a set of 500 documents in both museums. Museum professionals provided relevance assessments for the dataset by assessing each document either relevant or not relevant separately for all of the queries. The dataset and relevance assessment were carried out specifically for this study. This is a relatively large set of queries and relevance assessments for one-off experiment because the recall is analyzed with full coverage by domain experts meaning that all of the documents are manually inspected against all of the queries. Pooling or automatic pre-filtering was not used. This makes the relevance assessments highly reliable, avoids bias caused by automatic pre-filtering, and takes into account all possible semantic relevance, even non-trivial connections judged relevant by the domain experts.

The domain experts were asked to create queries typical for the domain, such that the queries would include also non-trivial queries considering the underlying collection. For example, a query containing the concept "seascapes" was judged relevant also for objects annotated with the concept "landscape paintings", and for objects annotated with "marinas", "boats", "harbors" and so on. The judges were allowed to inspect the textual description in addition to the image of the objects when assessing relevance.

```
<rdf:Resource> <aat:300025789> . (astronomers)
<dc:subject> <aat:300134506> . (astronomical photography)
<dc:subject> <aat:300211119> . (optical toys)
<dc:subject> <aat:300056210> . (optical properties)
<rdf:type> <sm:Instrument> .

<rdf:Resource> "Leopoldo Nobili" .
<dc:subject> <aat:300197519> . (galvanometers)
<dc:subject> <aat:300002501> . (batteries)
<dc:subject> <aat:300054490> . (electrical engineering)
<rdf:type> <sm:Instrument>
```

Fig. 2. An example of two sets of queries defined by experts in the Museo Galileo. The namespace *dc* and *sm* refer to the Dublin Core and a custom extension of the Dublin Core properties for the cultural heritage domain, and *aat* to the Art and Architecture Thesaurus of the Getty Foundation. The subject of each RDF triple is omitted, because it is *rdf:Resource* for these queries.

3.4 Evaluation Metrics

The accuracy of the retrieval methods was measured using Recall, Precision and Mean Average Precision. In addition, we plotted non-interpolated precision-recall curves on 11 recall levels to get an understanding of the performance differences between the methods. Recall R is defined as the number of relevant documents retrieved by a search method divided by the total number of relevant documents in the system, while precision P is defined as the number of relevant documents retrieved by a search method divided by the total number of documents retrieved.

Precision and recall are vulnerable measures because often when precision increases, recall decreases and vice versa. Therefore, a single measure that can be used to estimate a balanced performance in terms of precision and recall can be useful. We are interested also on ranking and a natural measure to be used to investigate the ranking along with the precision and recall tradeoff is Mean Average Precision (MAP). MAP for a set of relevant documents $\{d_1, \dots, d_{m_j}\}$ for a query q_j of total Q queries and a set of ranked retrieval results RA_{jk} from the top result until one gets to document d_k is:

$$MAP(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_k^{m_j} P(RA_{jk}). \quad (8)$$

When a relevant document is not retrieved at all, the precision value in the above equation is taken to be 0. For a single information need, the average precision is the average area under the precision-recall curve for a set of queries.

3.5 Statistical Significance

The statistical significance of the differences of the results obtained using different combinations of methods were ensured using the the Friedman test. The Friedman test is a non-parametric test based on ranks and is suitable for comparing more than two related samples. The statistical significance between method pairs was then analyzed using a paired Wilcoxon Signed-Rank test with Bonferonni correction as a post-hoc test. The differences between the method variants were found to be statistically significant ($p < 0.001$). The Friedman test was chosen because the data was not found to be normally distributed using the Shapiro and Wilk test.

4 Results

Figures 3 and 4 summarize the results. Figure 3 presents the precision - recall using each method variant when no query expansion was used. The curve on Figure 4 presents the same results for the best query expansion determined by the Wu-Palmer cutoff that was found to lead to best MAP for each method variant. In other words, the best achieved indexing strategy - query expansion combination. The following main findings can be observed. First, using the full indexing leads to the best overall performance. It performs equally good to subsumption indexing when a combination of query expansion and reasoning is used, but outperforms the subsumption indexing on low recall levels and

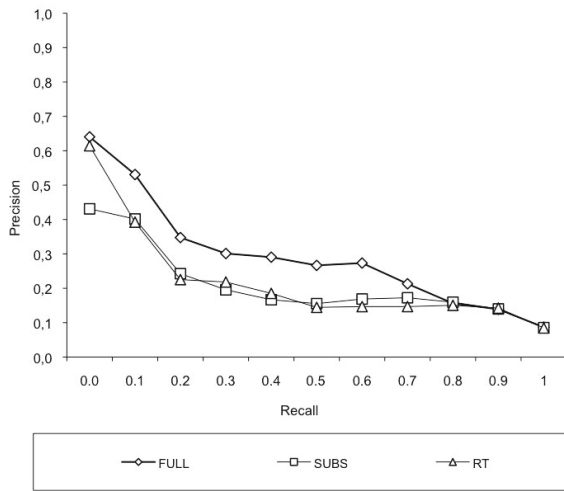


Fig. 3. Precision plotted on 11 recall levels for different reasoning and indexing strategies. No query expansion is used. The values are averaged over the 40 queries.

in the case where no query expansion is used. Second, the subsumption indexing and full indexing outperform related terms indexing in all tasks. The results show up to 76% improvement compared to a variation where no reasoning and query expansion are used.

Both, indexing using subsumption and full indexing, that also uses subsumption, seems to perform clearly better than related term indexing. The performance is increased by 0.15 (68%) in MAP compared to the baseline. The gain in performance achieved using subsumption and full indexing strategies imply that subsumption reasoning is the most important factor affecting the accuracy of the retrieval. Full indexing strategy clearly outperforms the other strategies on overall performance and performs best even on the lowest recall levels. An interesting finding is that subsumption reasoning and indexing strategy do perform worse than related terms strategy on the low recall levels, when reasoning is not complemented with query expansion.

Query expansion has a significant overall effect and, in addition to reasoning, is an important factor affecting the accuracy of the retrieval. Query expansion increases the accuracy up to 0.16 (76%) in terms of MAP when full expansion reasoning and indexing strategy is used. It is notable that the subsumption reasoning and indexing strategy actually performs only equally good compared to the baseline approach when no additional query expansion is used. This indicates that the combination of correct indexing strategy and query expansion is crucial to achieve optimal accuracy. An additional query expansion using super concepts from ontologies was found to be most effective when using cut-off value 0.9 to 0.7 of the Wu-Palmer measure. This means an expansion of zero to three nodes in the ontology graph in addition to the standard RDF(S) reasoning.

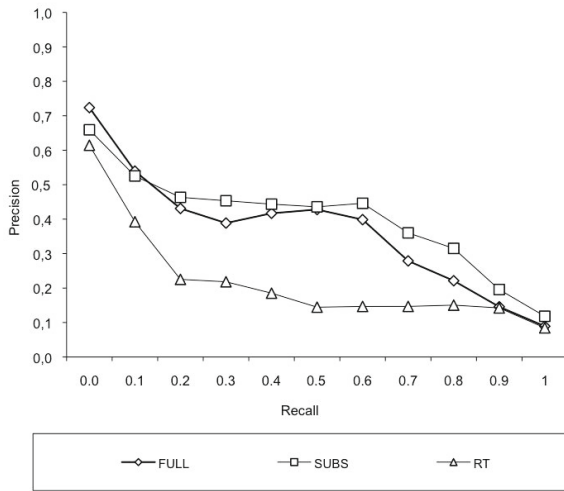


Fig. 4. Precision plotted on 11 recall levels for different reasoning and indexing strategies. The best combination of query expansion and reasoning is used. The values are averaged over the 40 queries.

It is observable in the results that the gold standard and queries favor recall-oriented methods, which was expected to be the case in domain specific setting. For example, the subsumption indexing strategy with the Wu-Palmer cut-off at 0.4 leads to an equally good performance as the cut-off 0.7, while cut-off values 0.5 and 0.6 perform worse. This indicates that using extensive query expansion compensates better semantic approximation achieved using related term relations together with subsumption reasoning. We believe that this is due to the fact that our data set consists of documents from a relatively specific domain and a collection of only 1000 documents. In additional runs we observed that precision - recall curves have different tradeoffs when varying the Wu-Palmer cut-off values. Using more query expansion increases recall, but does not hurt precision as extensively as could be expected. Our conclusion is that our dataset favors recall oriented approaches without a serious precision trade-off. This may not be the case in settings, where data is retrieved from a data cloud that is linked to other domains. Therefore, we believe that full expansion with mild query expansion leads to best overall performance.

5 Conclusions and Discussion

In this paper, we propose an indexing and retrieval framework for structured Web data to support domain specific retrieval of RDF data. The framework is computationally feasible because it avoids the high dimensionality of the triple space in similarity computation by using triple based indexing. We conducted a set of experiments to validate

the performance of the approach and combine different reasoning, indexing and query expansion strategies. We show that ontology-based query expansion and reasoning improves retrieval in Semantic Web data retrieval and can be effectively used in our adaptation of the vector space model. We also provide empirical evidence to support the effect of self-tuning query expansion method that is based on a metric that measures the depth of the ontology graphs. The experimental evaluation of the framework led to the following conclusions:

1. The best combination of reasoning and query expansion leads to improvement of accuracy up to 76%.
2. Full reasoning and indexing strategy improves accuracy of retrieval, with best results achieved when combined with query-expansion.
3. Query expansion that considers also other relations than those belonging to the standard RDF(S) reasoning improves results. The Wu-Palmer cut-off values around 0.7-0.9 when combined with full indexing leads to best results.
4. Using only subsumption indexing seems to work relatively well in our experiments, but requires extensive expansion. This can be problematic with more diverse datasets than the ones used in this study.

We conducted experiments that tested a number of different techniques and their combinations. However, the experimental setup leaves room for further research. While we used two separate collections and queries from different annotators and institutions, these were indexed using the same ontologies. The data used in the experiments is from the cultural heritage domain and may not generalize to other more open domains.

We measured the performance of the methods against expert created gold standard on a set of domain specific annotations on the cultural heritage domain. The relevance assessments are determined manually for the whole dataset, unlike in some other datasets proposed for semantic search evaluation, such as the Semantic Search Workshop data [9], where the relevance assessments were determined by assessing relevance for documents pooled from 100 top results from each of the participating systems, queries were very short, and in text format. This ensures that our dataset enables measuring recall and all of the query-document matches, even non-trivial, are present. The set of queries, for which the relevance assessments were created, are in the form of sets of triples. This avoids the problems of query construction and disambiguation of the terms, which means that we are able to measure the retrieval performance independently of the user interface or initial query construction method. While we realize that disambiguation and query construction are essential for search engines, we think that they are problems of their own to be tackled by the Semantic Search community. Our methods are therefore valid for information filtering scenarios and search scenarios that can use novel query construction methods, such as faceted search, or query suggestion techniques. The methods only operate on numerical space for triples and implements ranking independently from the specific RDF dataset it could also be implemented as a ranking layer under database management systems that support more formal query languages such as SPARQL. Our experiments were run on a gold standard acquired specifically for this study, that makes the results more reliable and the gold standard highly reliable. However, we used relatively small dataset of 1000 documents which

makes the task recall oriented. However, the methods themselves scale to large collections, because the indexing and retrieval framework does not make any assumptions over the classic VSM and are able to delimit the dimensionality of the VSM based on splitting the space separately for each property. However, small collections are typical in domain specific search and the results may not be directly comparable with results obtained for other collections. While full query expansion with subsumption reasoning works well for such a homogenous dataset, this might not be true for more varying datasets. This is due to the fact that the best performance was achieved with the Wu-Palmer cut-off value of 0.4 that allows traversing the supertree of a concept for several nodes. This could hurt the accuracy when applied to larger precision oriented datasets. However, our results are a clear indication of the effectiveness of both query expansion and reasoning.

Furthermore, ontologies are not the only source for semantic information. Our method operates in pure numerical vector space that makes it possible to apply standard dimensionality reduction and topic modeling methods that could reveal the semantics based on collection statistics. Since our experiments showed that maximal query expansion using ontologies leads to best retrieval accuracy, such methods are an interesting future research direction.

Acknowledgements. The work was conducted under the Academy of Finland Grant (135536), Fulbright Technology Industries of Finland Grant, and Finnish Foundation for Technology Promotion Grant.

References

1. Agirre, E., Arregi, X., Otegi, A.: Document expansion based on wordnet for robust ir. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, COLING 2010, pp. 9–17. Association for Computational Linguistics, Stroudsburg (2010)
2. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web: Scientific American. Scientific American (May 2001)
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. Blanco, R., Mika, P., Vigna, S.: Effective and Efficient Entity Search in RDF 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. 83–97. Springer, Heidelberg (2011)
5. Brickley, D., Guha, R.V.: RDF vocabulary description language 1.0: RDF Schema W3C recommendation. Recommendation, World Wide Web Consortium (February 10, 2004)
6. Castells, P., Fernandez, M., Vallet, D.: An adaptation of the vector-space model for ontology-based information retrieval. *IEEE Transactions on Knowledge and Data Engineering* 19(2), 261–272 (2007)
7. Fazzinga, B., Gianforme, G., Gottlob, G., Lukasiewicz, T.: Semantic web search based on ontological conjunctive queries. *Web Semantics: Science, Services and Agents on the World Wide Web* 9(4), 453–473 (2011)
8. Fernández, M., Cantador, I., López, V., Vallet, D., Castells, P., Motta, E.: Semantically enhanced information retrieval: An ontology-based approach. *Web Semantics: Science, Services and Agents on the World Wide Web* 9(4), 434–452 (2011)

9. Halpin, H., Herzig, D., Mika, P., Blanco, R., Pound, J., Thompon, H., Duc, T.T.: Evaluating ad-hoc object retrieval. In: Proceedings of the International Workshop on Evaluation of Semantic Technologies, Shanghai, China. CEUR, vol. 666 (November 2010)
10. Kiryakov, A., Popov, B., Ognyanoff, D., Manov, D., Kirilov, A., Goranov, M.: Semantic Annotation, Indexing, and Retrieval. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 484–499. Springer, Heidelberg (2003)
11. Ning, X., Jin, H., Jia, W., Yuan, P.: Practical and effective ir-style keyword search over semantic web. *Information Processing & Management* 45(2), 263–271 (2009)
12. Pérez-Agüera, J.R., Arroyo, J., Greenberg, J., Iglesias, J.P., Fresno, V.: Using bm25f for semantic search. In: Proceedings of the 3rd International Semantic Search Workshop, SEMSEARCH 2010, pp. 2:1–2:8. ACM, New York (2010)
13. Ruotsalo, T., Aroyo, L., Schreiber, G.: Knowledge-based linguistic annotation of digital cultural heritage collections. *IEEE Intelligent Systems* 24(2), 64–75 (2009)
14. Ruotsalo, T., Mäkelä, E.: A comparison of corpus-based and structural methods on approximation of semantic relatedness in ontologies. *International Journal on Semantic Web and Information Systems* 5(4), 39–56 (2009)
15. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. *Communications of the ACM* 18(11), 613–620 (1975)
16. Vallet, D., Fernández, M., Castells, P.: An Ontology-Based Information Retrieval Model. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 455–470. Springer, Heidelberg (2005)

Exchange and Consumption of Huge RDF Data

Miguel A. Martínez-Prieto^{1,2}, Mario Arias Gallego^{1,3},
and Javier D. Fernández^{1,2}

¹ Department of Computer Science, Universidad de Valladolid Spain

² Department of Computer Science, Universidad de Chile Chile

³ Digital Enterprise Research Institute, National University of Ireland Galway
{migumar2,jfergar}@infor.uva.es, mario.arias@deri.org

Abstract. Huge RDF datasets are currently exchanged on textual RDF formats, hence consumers need to post-process them using RDF stores for local consumption, such as indexing and SPARQL query. This results in a painful task requiring a great effort in terms of time and computational resources. A first approach to lightweight data exchange is a compact (binary) RDF serialization format called HDT. In this paper, we show how to enhance the exchanged HDT with additional structures to support some basic forms of SPARQL query resolution without the need of "unpacking" the data. Experiments show that i) with an exchanging efficiency that outperforms universal compression, ii) post-processing now becomes a fast process which iii) provides competitive query performance at consumption.

1 Introduction

The amount and size of published RDF datasets has dramatically increased in the emerging Web of Data. Publication efforts, such as Linked Open Data^[1] have “democratized” the creation of such structured data on the Web and the connection between different data sources^[7]. Several research areas have emerged alongside this; RDF indexing and querying, reasoning, integration, ontology matching, visualization, etc. A common Publication-Exchange-Consumption workflow (Figure 1) is involved in almost every application in the Web of Data.

Publication. After RDF data generation, publication refers to the process of making RDF data publicly available for diverse purposes and users. Besides RDF publication with dereferenceable URIs, data providers tend to expose their data as a file to download (RDF dump), or via a SPARQL endpoint, a service which interprets the SPARQL query language^[2].

Exchange. Once the consumer has discovered the published information, the exchange process starts. Datasets are serialized in traditional plain formats (*e.g.* RDF/XML^[5], N3^[4] or Turtle^[3]), and universal compressors (*e.g.* gzip) are commonly applied to reduce their size.

Consumption. The consumer has to post-process the information in several ways. Firstly, a decompression process must be performed. Then, the serialized

¹ <http://linkeddata.org/>

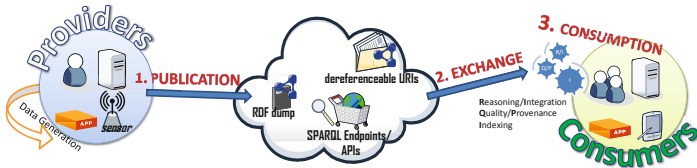


Fig. 1. Publication-Exchange-Consumption workflow in the Web of Data

RDF must be parsed and indexed, obtaining a data structure more suitable for tasks such as browsing and querying.

The scalability issues of this workflow arise in the following **running example**. Let us suppose that you publish a huge RDF dataset like Geonames (112 million triples about geographical entities). Plain data take up 12.07 GB (in Ntriples²), and compression should be applied. For instance, its gzip-compressed dump takes 0.69 GB. Thus, compression is necessary for efficient exchange (in terms of time) when managing huge RDF. However, after decompression, data remain in a plain format and an intensive post-processing is required³. Even when data is shared through a SPARQL endpoint, some queries can return large amounts of triples, hence the results must be compressed too.

Nowadays, the potential of huge RDF is seriously underexploited due to the large space they take up, the powerful resources required to process them, and the large consumption time. Similar problems arise when managing RDF in mobile devices; although the amount of information could be potentially smaller, these devices have more restrictive requirements for transmission costs/latency, and for post-processing due to their inherent memory and CPU constraints [14]. A first approach to lighten this workflow is a binary RDF serialization format, called HDT (*Header-Dictionary-Triples*) [11], recently accepted as a **W3C Member Submission** [6]. This proposal highlights the need to move forward plain RDF syntaxes to a data-centric view. HDT modularizes the data and uses the skewed structure of RDF graphs [9] to achieve compression. In practical terms, HDT-based representations take up to 15 times less space than traditional RDF formats [11].

Whereas publication and exchange were partially addressed in HDT, the consumption is underexploited; HDT provides basic retrieval capabilities which can be used for limited resolution of SPARQL triple patterns. This paper revisits these capabilities for speeding up consumption within the workflow above. We propose i) to publish and exchange RDF serialized in HDT, and then ii) to perform a lightweight post-process (at consumption) enhancing the HDT representation with additional structures providing a full-index for RDF retrieval. The resulting enhanced HDT representation (referred to as HDT-FoQ: *HDT Focused on Querying*) enables the exchanged RDF to be directly queryable with SPARQL, speeding up the workflow in several correlated dimensions:

² <http://www.w3.org/TR/rdf-testcases/#ntriples>

³ *Post-processing* is the computation needed at consumption (parsing+indexing) before any query can be issued.

- RDF datasets are exchanged in compact HDT, reducing transmission costs.
- HDT-FoQ is built on top of HDT, requiring little post-processing. It excels in consumption latency (the time awaited until the dataset can be consumed).
- HDT-FoQ provides efficient in-memory resolution of triple patterns and joins.

Our experimental results report figures on each of the achievements above. In particular, our HDT-driven approach completes the workflow 10 – 15 times faster than traditional solutions, outperforming them in the three processes. Query performance evaluation shows that i) the resultant indexed HDT-FoQ achieves the best overall performance for triple patterns resolution, and ii) an ad-hoc join implementation on top of HDT-FoQ reports competitive results with respect to optimized solutions within the state-of-the-art.

This paper is structured as follows. Section 2 reviews the state-of-the-art and sets HDT foundations. In Section 3, HDT is revisited for basic consumption, and Section 4 describes how HDT-FoQ enhances it to achieve efficient SPARQL resolution. Section 5 provides experimental results about the impact of HDT in the current scenario. Finally, Section 6 concludes and devises future work.

2 State-of-the-Art

Huge RDF datasets are currently serialized in verbose formats (RDF/XML [5], N3 [4] or Turtle [3]), originally designed for a document-centric Web. Although they compact some constructions, they are still dominated by a human-readable view which adds an unnecessary overhead to the final dataset representation. It increases transmission costs and delays final data consumption within the Publication-Exchange-Consumption workflow.

Besides serialization, the overall performance of the workflow is determined by the efficiency of the external tools used for post-processing and consuming huge RDF. Post-processing transforms RDF into any binary representation which can be efficiently managed for specific consumption purposes. Although it is performed once, the amount of resources required for it may be prohibitive for many potential consumers; it is specially significant for mobile devices comprising a limited computational configuration.

Finally, the consumption performance is determined by the mechanisms used for access and retrieval RDF data. These are implemented around the SPARQL [2] foundations and their efficiency depends on the performance yielded by RDF indexing techniques. Relational-based solutions such as Virtuoso [10] are widely accepted and used to support many applications consuming RDF. On the other hand, some stores build indexes for all possible combinations of elements in RDF (SPO, SOP, PSO, POS, OPS, OSP), allowing i) all triple patterns to be directly resolved in the corresponding index, and ii) the first join step within a BGP to be resolved through fast merge-join. Hexastore [18] performs a memory-based implementation which, in practice, is limited by the space required to represent and manage the index replication. RDF-3X [17] performs multi-indexing on a disk-resident solution which compresses the indexes within B⁺-trees. Thus, RDF-3X enables the management of larger datasets at the expense of overloading querying processes with expensive I/O transferences.

Speeding up consumption within this workflow is influenced by two factors: i) the RDF serialization format, as it should be compact for exchanging and friendly for consumption, and ii) efficient RDF retrieval. Scalability issues underlying to these processes justify the need for a binary RDF format like HDT [6].

2.1 Binary RDF Representation (HDT)

HDT is a binary serialization format which organizes RDF data in three logical components. The **Header** includes logical and physical metadata describing the RDF dataset and serves as an entry point to its information. The **Dictionary** provides a catalog of the terms used in the dataset and maps them to unique integer IDs. It enables terms to be replaced by their corresponding IDs and allows high levels of compression to be achieved. The **Triples** component represents the pure structure of the underlying graph after the ID replacement.

Publication and exchange processes are partially addressed by HDT. Although it is a machine-oriented format, the Header gathers human-friendly textual metadata such as the provenance, size, quality, or physical organization (subparts and their location). Thus, it is a mechanism to discover and filter published datasets. In turn, the Dictionary and Triples partition mainly aims at efficient exchange; it reduces the inherent redundancy to an RDF graph by isolating terms and structure. This division has proved effective in RDF stores [17].

3 Revisiting HDT for Basic Consumption

HDT allows different implementations for the dictionary and the triples. Besides achieving *compression*, some implementations can be optimized to support native data retrieval. The original HDT proposal [11] gains insights into this issue through a triples implementation called Bitmap Triples (BT). This section firstly gives basic notions of succinct data structures, and then revisits BT emphasizing how these structures can allow basic consumption.

3.1 Succinct Data Structures

Succinct data structures [16] represent data using as little space as possible and provide direct access. These savings allow them to be managed in faster levels of the memory hierarchy, achieving competitive performance. They provide three primitives (\mathcal{S} is a sequence of length n from an alphabet Σ):

- $\mathbf{rank}_a(\mathcal{S}, i)$ counts the occurrences of $a \in \Sigma$ in $\mathcal{S}[1, i]$.
- $\mathbf{select}_a(\mathcal{S}, i)$ locates the position for the i -th occurrence of $a \in \Sigma$ in \mathcal{S} .
- $\mathbf{access}(\mathcal{S}, i)$ returns the symbol stored in $\mathcal{S}[i]$.

In this paper, we make use of succinct data structures for representing sequences of symbols. We distinguish between binary sequences (*bitsequences*) and general sequences. i) **Bitsequences** are a special case drawn from $\Sigma = \{0, 1\}$. They can be represented using $n + o(n)$ bits of space while answering the three previous

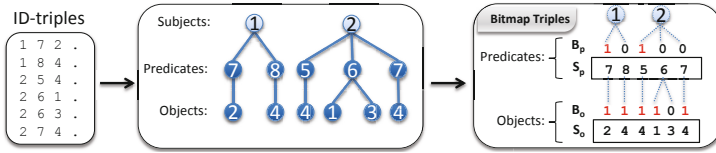


Fig. 2. Description of Bitmap Triples

operations in constant time. We use the implementation of González, *et al.* [12] which takes, in practice, 37.5% extra space on top of the original bitsequence size.

ii) **General sequences** are represented using *wavelet trees* [13]. A wavelet tree represents a general sequence as a balanced tree of height $h = \log \sigma$, comprising a total of h bitsequences of n bits. It uses $n \log \sigma + o(n) \log \sigma$ bits and answers **rank**, **select** and **access** in proportional time to its height h .

3.2 Bitmap Triples for SP-0 Indexing

HDT describes *Bitmap Triples* (BT) as a specific triples encoding which represents the RDF graph through its adjacency matrix. In practice, BT slices the matrix by subject and encodes the *predicate-object* lists for each subject in the dataset. Let us suppose that the triples below comprise all occurrences of subject s :

$$\{(s, p_1, o_{11}), \dots, (s, p_1, o_{1n_1}), (s, p_2, o_{21}), \dots, (s, p_2, o_{2n_2}), \dots, (s, p_k, o_{kn_k})\}$$

These triples are reorganized into *predicate-object* adjacency lists as follows:

$$s \rightarrow [(p_1, (o_{11}, \dots, o_{1n_1})), (p_2, (o_{21}, \dots, o_{2n_2})), \dots, (p_k, (o_{k1}, \dots, o_{kn_k}))].$$

Each list represents a predicate, p_j , related to s and contains all objects reachable from s through this predicate.

This transformation is illustrated in Figure 2; the ID-based triples representation (labeled as *ID-triples*) is firstly presented, and its reorganization in adjacency lists is shown on its right. As can be seen, adjacency lists draw tree-shaped structures containing the subject ID in the root, the predicate IDs in the middle level, and the object IDs in the leaves (note that each tree has as many leaves as occurrences of the subject in the dataset). For instance, the right tree represents the second adjacency list in the dataset, thus it is associated to the subject 2 (rooting the tree). In the middle level, the tree stores (in a sorted way) the three IDs representing the predicates related to the current subject: 5, 6, and 7. The leaves comprise, in a sorted fashion, all objects related to the subject 2 through a given predicate: e.g. objects 1 and 3 are reached through the path 2, 6; which means that the triples (2, 6, 1) and (2, 6, 3) are in the dataset.

BT implements a compact mechanism for modeling an RDF graph as a forest containing as many trees as different subjects are used in the dataset. This assumption allows subjects to be implicitly represented by considering that the i -th tree draws the adjacency list related to the i -th subject. Moreover, two integer sequences: \mathcal{S}_p and \mathcal{S}_o , are used for storing, respectively, the predicate and the object IDs within the adjacency lists. Two additional bitsequences: \mathcal{B}_p and \mathcal{B}_o

Table 1. Triple pattern resolution on BT (operations marked with * are performed as many times as predicates to be included in the list obtained from `findPred`)

Triple Pattern	Operations
(S, P, O)	<code>findPred(S)</code> , <code>filterPred(\mathcal{P}_s, P)</code> , <code>findObj(pos)</code> , <code>filterObj(\mathcal{O}_x, O)</code> .
$(S, P, ?O)$	<code>findPred(S)</code> , <code>filterPred(\mathcal{P}_s, P)</code> , <code>findObj(pos)</code> .
$(S, ?P, O)$	<code>findPred(S)</code> , <code>{findObj(pos)</code> , <code>filterObj(\mathcal{O}_x, O)}</code> *.
$(S, ?P, ?O)$	<code>findPred(S)</code> , <code>findObj(pos)</code> *.

(storing list cardinalities) are used for delimitation purposes. This is illustrated on the right side of Figure 2. As can be seen, \mathcal{S}_p stores the five predicate IDs involved in the adjacency lists: $\{7, 8, 5, 6, 7\}$ and \mathcal{B}_p contains five bits: $\{10100\}$, which are interpreted as follows. The *first* 1-bit (in $\mathcal{B}_p[1]$) means that the list for the *first subject* begins at $\mathcal{S}_p[1]$, and the *second* 1-bit (in $\mathcal{B}_p[3]$) means that the list for the *second subject* begins at $\mathcal{S}_p[3]$. The cardinality of the list is obtained by subtracting the positions, hence the adjacency lists for the first and the second subject contain respectively $3 - 1 = 2$ and $6 - 3 = 3$ predicates. The information stored in $\mathcal{S}_o = \{2, 4, 4, 1, 3, 4\}$ and $\mathcal{B}_o = \{111101\}$ is similarly interpreted, but note that adjacency lists, at this level, are related to subject-predicate pairs.

BT gives a practical representation of the graph structure which allows triples to be sequentially listed. However, direct accessing to the triples in the i -th list would require a sequential search until the i -th 1-bit is found in the bitsequence. Direct access (in constant time) to any adjacency list could be easily achieved with a little spatial $o(n)$ overhead on top of the original bitsequence sizes. It ensures constant time resolution for `rank`, `select`, and `access`, and allows efficient primitive operations to be implemented on the adjacency lists:

- `findPred(i)`: returns the list of predicates related to the subject i (\mathcal{P}_i), and the position pos in which this list begins in \mathcal{S}_p . This position is obtained as $pos = select_1(\mathcal{B}_p, i)$, and \mathcal{P}_i is retrieved from $\mathcal{S}_p[pos, select_1(\mathcal{B}_p, i + 1) - 1]$.
- `filterPred(\mathcal{P}_i, j)`: performs a binary search on \mathcal{P}_i and returns the position of the predicate j in \mathcal{P}_i , or 0 if it is not in the list.
- `findObj(pos)`: returns the list of objects (\mathcal{O}_x) related to the subject-predicate pair represented in $\mathcal{S}_p[pos]$. It positions the pair: $x = rank_1(\mathcal{B}_o, pos)$, and then extracts \mathcal{O}_x from $\mathcal{S}_o[select_1(\mathcal{B}_o, x), select_1(\mathcal{B}_o, x + 1) - 1]$.
- `filterObj(\mathcal{O}_j, k)`: performs a binary search on \mathcal{O}_j and returns the position of the object k in \mathcal{O}_j , or 0 if it is not in the list.

Table 1 summarizes how these primitives can be used to resolve some triple patterns in SPARQL: (S, P, O) , $(S, P, ?O)$, $(S, ?P, O)$, and $(S, ?P, ?O)$. Let us suppose that we perform the pattern $(2, 6, ?)$ over the triples in Figure 2. BT firstly retrieves (by `findPred(2)`) the list of predicates related to the subject 2: $\mathcal{P}_2 = \{5, 6, 7\}$, and its initial position in \mathcal{S}_p : $pos_{ini} = 3$. Then, `filterPred($\mathcal{P}_2, 6$)` returns $pos_{off} = 2$ as the position in which 6 is in \mathcal{P}_2 . This allows us to obtain the position in which the pair $(2, 6)$ is represented in \mathcal{S}_p : $pos = pos_{ini} + pos_{off} = 3 + 2 = 5$, due to \mathcal{P}_2 starts in $\mathcal{S}_p[3]$, and 6 is the second element in this list. Finally, `findObj(5)` is executed for retrieving the final result comprising the list of objects $\mathcal{O}_5 = \{1, 3\}$ related to the pair $(2, 6)$.

4 Focusing on Querying (*HDT-FoQ*)

HDT was originally intended for publication and exchange, but its triples component provides enough information for efficient RDF retrieval. The bitsequences delimiting adjacency lists provide an SP-0 index which allows some patterns to be efficiently resolved (row BT in Table 2). It enables HDT to be exploited as a basis for an indexed representation (called HDT-FoQ: *HDT Focused on Querying*) which allows exchanged RDF to be directly consumed using SPARQL.

This section presents how HDT is enhanced from an innovative perspective focused on querying. Three main issues must be addressed to obtain an efficient configuration for SPARQL resolution: i) The **dictionary** is serialized in a compressed way which allows it to be included as part of the original HDT representation. It must also be directly consumable to provide efficient operations for querying the mapping between each term and the corresponding ID. ii) The original **triples** component representation is enhanced to provide efficient RDF retrieval covering all possible triple patterns in SPARQL. iii) Efficient **join algorithms** are implemented to perform Basic Graph Patterns (BGPs) 2.

4.1 Functional Dictionary Serialization

The dictionary component contributes greatly to the HDT compactness because it enables triples to be modeled through three-ID groups. However, RDF dictionaries can suffer from scalability drawbacks because they take more space than ID-triples representations 15. An advanced serialization addresses this drawback while providing basic operations for consumption, *i.e.*, operations from *term to ID* (*locate*), and from *ID to term* (*extract*). HDT-FoQ relies on an HDT representation including such an advanced dictionary. It is organized as follows:

- **Common subjects and objects** (S0) maps to the range [1, |S0|] all terms playing *subject* and *object* roles.
- **Subjects** (S) maps to [|S0|+1, |S0|+|S|] all remaining terms playing as *subject*.
- **Objects** (0) maps to [|S0|+1, |S0|+|0|] all remaining terms playing as *object*.
- **Predicates** (P) maps terms playing as predicate to [1, |P|].

This four-subset partitioning fits the original HDT approach 11 and allows terms playing as subject and object to be represented only once. This dictionary organization is serialized through four independent streams which respectively concatenate, in lexicographic order, the terms within each subset. Each stream is finally encoded with *Plain Front-Coding* (PFC) 8. It adapts differential Front-Coding compression 19 to the case of string dictionaries and it provides, at consumption, efficient *locate* and *extract* resolution in compressed space.

4.2 A Wavelet Tree-Based Solution for PS-0 Indexing

Bitmap Triples (BT) represents the triples component through adjacency lists prioritized by subject. This decision addresses fast querying for patterns providing the subject, but makes retrieval by any other dimension difficult.

Table 2. Indexes and Triple Pattern resolution through incremental proposals

	Index Order			Triple Patterns						
	SP-O	PS-O	OP-S	SPO	SP?	S?O	S??	?PO	?P?	??O
BT	✓	-	-	SP-O	SP-O	SP-O	SP-O	-	-	-
BT+ \mathcal{W}_P	✓	✓	-	SP-O	SP-O	SP-O	SP-O	PS-O	PS-O	-
HDT-FoQ	✓	✓	✓	SP-O	SP-O	SP-O	SP-O	OP-S	PS-O	OP-S

Algorithm 1. `findSubj(i)`

```

1: occs ← ranki( $\mathcal{W}_p, n$ );
2: for ( $x = 1$  to occs) do
3:   pos[ $x$ ] ← selecti( $\mathcal{W}_p, x$ );
4:   S[ $x$ ] ← rank1( $\mathcal{B}_p, pos[x]$ );
5: end for
6: return pos; S

```

Algorithm 2. `filterSubj(i, j)`

```

1: posj ← select1( $\mathcal{B}_p, j$ ) - 1;
2: posj+1 ← select1( $\mathcal{B}_p, j + 1$ ) - 1;
3: occs ← ranki( $\mathcal{W}_p, pos_{j+1}$ ) - ranki( $\mathcal{W}_p, pos_j$ );
4: return occs

```

We firstly focus on predicate-based retrieval on top of BT. This requires the efficient resolution of patterns providing the predicate while leaving the subject as variable: $(?, P, 0)$ and $(?, P, ?)$. In both cases, all occurrences of P must be quickly located, but BT scatters them along the sequence of predicates (\mathcal{S}_p) and its sequential scan arises as the trivial solution. Thus, the predicate-based retrieval demands indexed access to \mathcal{S}_p , which can be provided by representing the sequence with the wavelet tree structure.

This new wavelet-tree based representation of \mathcal{S}_p is renamed \mathcal{W}_p . It adds an additional overhead of $o(n) \log(|P|)$ bits to the space used in the original \mathcal{S}_p , and allows each predicate occurrence to be located in time $O(\log |P|)$ through the `select` operation. This is an acceptable cost for our retrieval purposes because of the small number of predicates used, in practice, for RDF modeling. In the same way, the `access` operation also has a logarithmic cost, so any predicate within \mathcal{W}_p is now retrieved in time $O(\log(|P|))$. Finally, note that `rank` allows the occurrences of a predicate to be counted up to a certain position of \mathcal{W}_p .

The wavelet tree structure allows access by predicate to be supported on two new primitives traversing adjacency lists:

- `findSubj(i)`: returns the list of subjects related to the predicate *i* and the positions in which they occur in \mathcal{W}_p . This operation is described in Algorithm 1. It iterates over all occurrences of the predicate *i* and processes one of them for each step. It locates the occurrence position in \mathcal{W}_p (line 3) and uses it for retrieving the subject (line 4) which is added to the result set.
- `filterSubj(i, j)`: checks whether the predicate *i* and the subject *j* are related. It is described in Algorithm 2. It delimits the predicate list for the *j*-th subject, and counts the occurrences of the predicate *i* to *pos_j* (*o_j*) and *pos_{j+1}* (*o_{j+1}*). *Iff* $o_{j+1} > o_j$, the subject and the predicate are related.

Hence, the wavelet tree contributes with a PS-O index which allows two additional patterns to be efficiently resolved (row BT+ \mathcal{W}_p in Table 2). Both $(?S, P, ?O)$ and $(?S, P, 0)$ first perform `findSubj(P)` to retrieve the list of subjects related to the predicate P. Then, $(?S, P, ?O)$ executes `findObj` for each retrieved subject

and obtains all objects related to it. In turn, $(?S,P,0)$ performs a `filterObj` for each subject to test if it is related to the object given in the pattern.

Let us suppose that, having the triples in Figure 2, we ask for all subjects and objects related through the predicate 7: $(?S,7,?O)$. `findSubj(7)` obtains the list of two subjects related to the predicate ($\mathcal{S} = \{1,2\}$) and their positions in \mathcal{W}_p ($\text{pos}=\{1,5\}$). The subsequent `findObj(1)` and `findObj(5)` return the list of objects $\{2\}$ and $\{4\}$ respectively representing the triples $(1,7,2)$ and $(2,7,4)$.

4.3 An Additional Adjacency List for OP-S Indexing

The wavelet-tree based enhancement leaves object-based access as the only non-efficient retrieval in our approach. As we illustrated in Figure 2, objects are represented as leaves of the tree drawn for each adjacency list, so the sequence \mathcal{S}_o stores all object occurrences within the dataset (each one related to the corresponding predicate-subject pair). In this case, we require an additional index OP-S which allows adjacency lists to be traversed from the leaves.

The index OP-S is represented with an integer sequence: \mathcal{S}_{oP} , which stores, for each object, a sorted list of references to the predicate-subject pairs (sorted by predicate) related to it. It is worth noting that the i -th predicate-subject pair is identified through the i -th 1-bit in \mathcal{B}_o . A bitsequence \mathcal{B}_{oP} is also used for representing cardinalities as in the upper levels. This is illustrated in Figure 3; for instance, the fourth list in \mathcal{S}_{oP} (pointed to by the fourth 1-bit in \mathcal{B}_{oP}) stores the reference $\{3,5,2\}$: 3 points to the third 1-bit in \mathcal{B}_o representing the relation between the object 4 and the predicate 5; the reference 5 points to the fifth 1-bit (predicate 7), and the third reference: 8 points to the predicate 8.

This index OP-S enables efficient object-based retrieval through two new primitives which traverse adjacency lists from the leaves:

- `findPredSubj(i)`: returns the list of predicate-subject pairs related to the object i . This operation is described in Algorithm 3. It firstly delimits the list for the object i and then iterates over each reference. Each step locates the position ptr which points to the position which represents the reference in \mathcal{B}_o (line 4). This value is then used for retrieving the corresponding predicate from \mathcal{W}_p (line 5), and the subject from \mathcal{B}_p (line 6).
- `filterPredSubj(i,j)`: checks whether the object i and the predicate j are related, and narrows their occurrences in the list. It firstly delimits the list for the object i and then binary searches it to narrow the occurrences of j .

This enhancement contributes to our solution with the index OP-S and allows triple patterns $(?S,?P,0)$ and $(?S,P,0)$ to be efficiently resolved (see row **HDT-FoQ** in Table 2). The first one is resolved by performing `findPredSubj` for the object provided in the pattern. $(?S,P,0)$ was resolved through the wavelet tree, but its resolution is now speeded up. In this case, `filterPredSubj(0,P)` narrows the references from 0 to P, and then retrieves the corresponding subject (as in line 6 of Algorithm 3).

The functionality of the index OP-S can be seen through the $(?S,?P,1)$ pattern. It asks for all subject-predicate pairs related to the object 1 in the

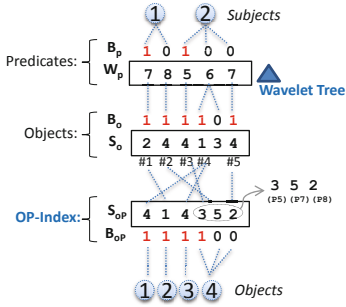


Fig. 3. Final HDT-FoQ configuration

Algorithm 3. `findPredSubj(i)`

```

1:  $pos_i \leftarrow select_1(\mathcal{B}_{oP}, i)$ ;
2:  $pos_{i+1} \leftarrow select_1(\mathcal{B}_{oP}, i + 1) - 1$ ;
3: for ( $x = pos_i$  to  $pos_{i+1}$ ) do
4:    $ptr \leftarrow select_1(\mathcal{B}_o, \mathcal{B}_{oP}[x])$ ;
5:    $\mathcal{P}[] \leftarrow access(\mathcal{W}_p, ptr)$ ;
6:    $\mathcal{S}[] \leftarrow rank_1(\mathcal{B}_p, ptr)$ 
7: end for
8: return  $\mathcal{P}; \mathcal{S}$ 

```

triples represented in Figure 2. The operation `findPredSubj(i)` firstly narrows the range of references from the object 1 to $\mathcal{S}_{oP}[1, 1]$. It only comprises the value 4 which points to the fourth 1-bit in \mathcal{B}_p . It represents the predicate $access(\mathcal{W}_p, 4) = 6$ and the subject $rank_1(\mathcal{B}_p, 4) = 2$, so this pattern matches the triple (2, 6, 1).

4.4 Joining Triple Patterns

HDT-FoQ is the result of post-processing HDT for RDF consumption. It is built, at consumption, on top of the exchanged HDT, which includes the functional dictionary and the Bitmap Triples. First, the wavelet tree is constructed using the implementation provided in the *libcds* library [1]. Then, the object structure in Bitmap Triples is scanned to build the OP-S index. The result is a compact RDF representation optimized to be managed and queried in main memory. As summarized in Table 2, HDT-FoQ only requires three indexes (SP-0, PS-0 and OP-S) to perform efficient RDF retrieval, in contrast to the six combinations used in solutions within the state-of-the-art [18, 17].

HDT-FoQ efficiently performs triple patterns, setting the basis for SPARQL resolution. We rely on the fact that the SPARQL core is built around the concept of Basic Graph Pattern (BGP) and its semantics in order to build conjunctive expressions joining triple patterns through shared variables.

Merge and **Index** joins can be directly resolved on top of HDT-FoQ. **Merge join** is used when the results of both triple patterns are sorted by the join variable. It is worth noting that triple pattern results are given in the order provided by the index used (see Table 2). If the results of one triple pattern are not sorted by the join variable, **index join** is performed. It first retrieves all results for the join variable in one triple pattern and replaces them in the other one. Ideally, the first evaluation should be on the less expensive pattern, in terms of its expected number of results.

5 Experimental Evaluation

This section studies the Publication-Exchange-Consumption workflow on a real-world setup in which the three main agents are involved:

- The **data provider** is implemented on a powerful computational configuration. It simulates an efficient data provider within the Web of Data. We use an Intel Xeon E5645@2.4GHz, 96GB DDR3@1066Mhz.
- The **network** is regarded as an ideal communication channel for a fair comparison. It is considered free of errors and any other external interference. We assume a transmission speed of 2Mbyte/s.
- The **consumer** is designed on a conventional configuration because it plays the role of any agent consuming RDF within the Web of Data. It is implemented on an AMD-PhenomTM-II X4 955@3.2GHz, 8GB DDR2@800MHz.

We first analyze the impact of using HDT as a basis for publication, exchange and consumption within the studied workflow, and compare its performance with respect to those obtained for the methods currently used in each process. Then, we focus on studying the performance of HDT-**FoQ** as the querying infrastructure for SPARQL: we measure response times for triple pattern and join resolution.

All experiments are carried out on a heterogeneous configuration of real-world datasets of different sizes and from different application domains (Table 3). We report “user” times in all experiments. The HDT prototype is developed in C++ and compiled using `g++-4.6.1 -O3 -m64`. Both the HDT library and a visual tool to generate/browse/query HDT files are publicly available⁴.

Table 3. Description of the real-world datasets used in the experimentation

Dataset	Plain Ntriples	Size (MB)	Available at
linkedMDB	6,148,121	850.31	http://queens.db.toronto.edu/~oktie/linkedmdb
dblp	73,226,756	11,164.41	http://dblp.l3s.de/dblp++.php
geonames	112,335,008	12,358.98	http://download.geonames.org/all-geonames-rdf.zip
dbpedia (en)	257,869,814	37,389.90	http://wiki.dbpedia.org/Downloads351

5.1 Analyzing the Publication-Exchange-Consumption Workflow

The overall workflow analysis considers that the publication process is performed once, whereas exchange and preprocessing costs are paid each time that any consumer retrieves the published dataset. The publication policy affects the performance of exchange because it depends on the dataset size, but also the decompression time (as initial consumption step) which is directly related to the compressor used for publication. We use two Lempel-Ziv based compressors⁵ for publication: the widely-used `gzip` and the `lzma` algorithm in the suite `p7zip`.

We assume that the publication process begins with the dataset already serialized. Thus, `gzip/lzma` based publication only considers the compression time, whereas processes based on HDT comprise the times required for generating the HDT representation and its subsequent compression.

Table 5 shows the time used for publication in the data provider: `gzip` is the faster choice and largely outperforms `lzma` and the HDT-based publication.

⁴ <http://www.rdfhdt.org>

⁵ <http://www.gzip.org/> (`gzip`), and <http://www.7-zip.org/> (`lzma`)

Table 4. Compressed sizes (MB)

Dataset	gzip	lzma	HDT+	
			gzip	lzma
linkedMDB	38.86	19.21	15.77	12.49
dblp	468.47	328.17	229.23	178.71
geonames	701.98	348.93	305.30	236.59
dbpedia	3,872.29	2,653.79	1,660.73	1,265.43

Table 5. Publication times (seconds)

Dataset	gzip	lzma	HDT+	
			gzip	lzma
linkedMDB	11.36	882.80	65.57	91.01
dblp	162.91	6,214.32	808.93	1,319.60
geonames	196.90	14,683.90	1,586.15	2,337.82
dbpedia	956.71	27,959.85	3,648.75	7,306.17

Table 6. Exchange times (seconds)

Dataset	gzip	lzma	HDT+	
			gzip	lzma
linkedMDB	19.43	9.61	7.88	6.25
dblp	234.23	164.08	114.62	89.35
geonames	350.99	174.46	152.65	118.29
dbpedia	1,936.14	1,326.89	830.36	632.71

Table 7. Decompression times (seconds)

Dataset	gzip	lzma	HDT+	
			gzip	lzma
linkedMDB	3.04	5.11	0.33	1.05
dblp	37.08	70.86	4.63	14.82
geonames	45.49	87.51	8.81	19.91
dbpedia	176.14	357.86	46.51	103.03

Table 8. Indexing times (seconds)

Dataset	Virtuoso	Hexastore	RDF3X	HDT-FoQ
linkedMDB	369.05	1,810.67	111.08	1.91
dblp	5,543.99	×	1,387.29	16.79
geonames	17,902.43	×	2,691.66	43.98
dbpedia	×	×	7,904.73	124.44

Table 9. Overall times (seconds)

Dataset	Comp. RDF+ Indexing	Comp. HDT+ HDT-FoQ
linkedMDB	125.80	9.21
dblp	1,622.23	120.96
geonames	2,953.63	182.18
dbpedia	9,589.48	860.18

However, size is the most important factor due to its influence on the subsequent processes (Table 4). HDT+*lzma* is the best choice. It achieves highly-compressed representations: for instance, it takes 2 and 3 times less space than *lzma* and *gzip* for *dbpedia*. This spatial improvement determines exchange and decompression (for consumption) times as shown in Tables 6 and 7.

On the one hand, the combination of HDT and *lzma* is the clear winner for exchange because of its high-compressibility. Its transmission costs are smaller than the other alternatives: it improves them between 10 – 20 minutes for the largest dataset. On the other hand, HDT+*gzip* is the most efficient at decompression, but its improvement is not enough to make up for the time lost in exchange with respect to HDT+*lzma*. However, its performance is much better than the one achieved by universal compression over plain RDF. Thus, HDT-based publication and its subsequent compression (especially with *lzma*) arises as the most efficient choice for exchanging RDF within the Web of Data.

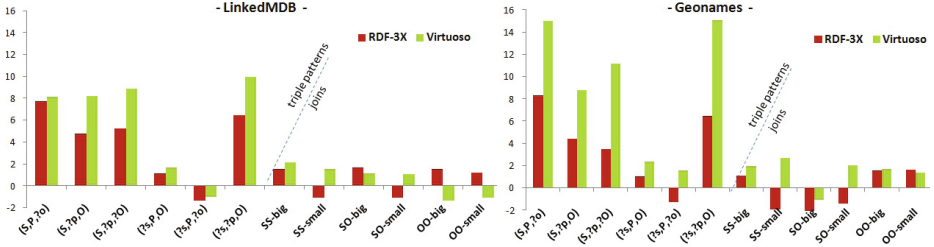
The next step focuses on making the exchanged datasets queryable for consumption. We implement the traditional process, which relies on the indexing of plain RDF through any RDF store. We choose three systems⁶: *Virtuoso* (relational solution), *RDF3X* (multi-indexing solution), and *Hexastore* (in-memory solution). We compare their performance against HDT-FoQ, which builds additional structures on the HDT-serialized datasets previously exchanged.

Table 8 compares these times. As can be seen, HDT-FoQ excels for all datasets: its time is between one and two orders of magnitude lower than that obtained

⁶ Hexastore has been kindly provided by the authors. <http://www.openlinksw.com/> (Virtuoso), <http://ht tp://www.mpi-inf.mpg.de/~neumann/rdf3x/> (RDF3X).

Table 10. Indexing sizes (MB)

Dataset	Virtuoso	Hexastore	RDF3X	HDT	HDT-FoQ
linkedMDB	518.01	6,976.07	377.34	48.70	68.03
dblp	3,982.01	×	3,252.27	695.73	850.62
geonames	9,216.02	×	6,678.42	1,028.85	1,435.05
dbpedia	×	×	15,802.03	4,433.28	5,260.33

**Fig. 4.** Comparison on querying performance

for the other techniques. For instance, HDT-FoQ takes 43.98 seconds to index *geonames*, whereas RDF3X and Virtuoso use respectively 45 minutes and 5 hours. It demonstrates how HDT-FoQ leverages the binary HDT representation to make RDF quickly queryable through its retrieval functionality. Finally, it is worth noting that Virtuoso does not finish the indexing for *dbpedia* after more than 1 day, and Hexastore requires a more powerful computational configuration for indexing datasets larger than *linkedMDB*. This fact shows that we successfully achieve our goal of reducing the amount of computation required by the consumer to make queryable RDF obtained within the Web of Data.

Overall Performance. This section comprises an overall analysis of the processes above. Note that publication is decoupled from this analysis because it is performed only once, and its cost is attributed to the data provider. Thus, we comprise times for exchange and consumption. These times are shown in Table 9. It compares the time needed for a conventional implementation against that of the HDT driven approach. We choose the most efficient configurations in each case: i) *Comp.RDF+Indexing* comprises *lzma* compression over the plain RDF representation and indexing in RDF3X, and ii) *Comp.HDT+HDT-FoQ* compresses the obtained HDT with *lzma* and then obtains HDT-FoQ.

The workflow is completed between 10 and 15 times faster using the HDT driven approach. Thus, the consumer can start using the data in a shorter time, but also with a more limited computational configuration as reported above.

5.2 HDT-FoQ in Consumption: Performance for SPARQL Querying

This section complements the previous analysis focusing on the performance of HDT-FoQ as the basis for SPARQL querying. As explained, HDT-FoQ is the final

result, in the consumer, when the workflow Publication-Exchange-Consumption is driven through our HDT-based approach, and it is used as an in-memory index for SPARQL querying. We firstly show the spatial needs of HDT-FoQ to be efficiently loaded in the consumer configuration and then study its performance for triple pattern and join query resolution. Our main aim is to show the HDT-FoQ efficiency for RDF retrieval and also for joining in order to demonstrate its capabilities for SPARQL resolution on top of that. We compare our results with respect to the indexing systems presented above.

Table 10 summarizes the sizes of the indexes built for each dataset within each studied solution. The columns HDT and HDT-FoQ, respectively, show the size of the original HDT representation (after decompression) and the resultant indexed one built on top of it. It is worth noting that the sizes reported for HDT-FoQ also include the overhead required for managing it in main memory. As can be seen, HDT-FoQ takes between 15% and 40% of extra space on top of HDT representations. These results place HDT-FoQ as the more compact index, largely doing better than the other solutions. Finally, we emphasize the comparison between *Hexastore* and HDT-FoQ because both are in-memory solutions. Whereas *Hexastore* requires ≈ 7 GB of main memory for managing just over 6 million triples (*linkedMDB*), our approach just uses 68.03 MB for fitting this dataset in memory. This achievement is analyzed from a complementary perspective: the consumer can manage just over 258 million triples (*dbpedia*) using HDT-FoQ (and 3GB of memory are still free in the system), whereas only 6 million triples can be managed using *Hexastore* (and only 1GB would remain free).

Query performance is evaluated over *linkedMDB* and *geonames*. For each one, we design a testbed of randomly generated queries which covers the entire spectrum of triple patterns and joins. We consider 5000 random triple patterns of each type ($(?S,P,?O)$ is limited by the number of different predicates). We split join tests into Subject-Subject (SS), Object-Object (OO) and Subject-Object (SO) categories. For each one, we generate 15 queries with a high number of intermediate results (*big* subsets) and another 350 queries with fewer results (*small* subsets). The full testbed is available at <http://dataweb.infor.uva.es/queries-eswc12.tgz>

Querying times are obtained by running 5 independent executions of the testbed and average total user times. We compare HDT-FoQ against RDF-3X and Virtuoso (*Hexastore* could not run most queries because of the aforementioned limited free memory). We query these systems within a “warm” scenario: we run 5 previous executions before measuring time for these disk-based systems to have the required data available in main memory. Figure 4 shows the performance comparison as $\{time_in_compared_system\}/\{time_in_FoQ\}$. For instance a value of 6 means that it performs 6 times faster than the compared system. For visualization purposes, we invert this ratio whenever we run slower, hence a value of -2 means that we perform 2 times slower.

It is worth noting that HDT-FoQ excels for almost every individual triple pattern. It speeds-up their resolution, only losing performance for $(?S,P,?O)$, in which a logarithmic cost is paid for accessing predicates in the wavelet tree. The analysis of join performance shows that i) HDT-FoQ is the most efficient choice

for most of the joins in medium-sized datasets such as `linkedMDB`, thanks to efficient triple pattern resolution, but ii) these stores leverage their optimized join implementations in larger datasets (`geonames`). Optimized join algorithms implemented on top of HDT-FoQ would allow it to compete fairly in this latter case by leveraging HDT-FoQ performance for triple pattern resolution.

6 Conclusions and Future Work

Inherent scalability drawbacks of huge RDF graphs discourage their consumption due to the space they take up, the powerful resources and the large time required to process them. In this paper, we focus on a novel direction for speeding up consumption. We firstly rely on an existing binary format, called HDT (Header-Dictionary-Triples), which provides efficient exchange. Then, we propose HDT-FoQ, a compact full-index created over HDT, at consumption. Thus, the exchanged RDF data become direct and easily queryable.

Our experiments show that huge RDF data are exchanged and post-processed (ready to be queried) 10 – 15 times faster than traditional solutions. Then, the proposed in-memory system for consumption (HDT-FoQ) excels in triple pattern resolution, remains competitive in joins of middle-sized datasets and shows potential improvement for larger datasets.

These results open up interesting issues for future work. We should work on improving predicate-based retrieval because it reports the less-competitive performance. Our on-going work relies on the optimization of the predicate index by tuning the trade-off between access time and spatial needs. In addition, we plan to optimize our join algorithms with *Sideways Information Passing* mechanisms, leveraging efficient resolution of triple patterns. Finally, although the use of succinct data structures allows more data to be managed in the main memory, it could remain excessive for consumers with limited memory. Under this scenario, we devise an evolution of HDT-FoQ to perform as an in-memory/on-disk system providing dynamic data management, *i.e.*, efficient insertion, updating and deletion of triples at consumption.

Acknowledgments. This research has been supported by MICINN (TIN2009-14009-C02-02) and Science Foundation Ireland under Grant No.~SFI/08/CE/I1380(Lion-II). The third author receives a grant from the JCyL and the ESF. We particularly wish to thank Claudio Gutierrez, for his continued motivation and selfless help, and the Database Lab (Univ. of A Coruña) for lending us the servers for our experiments.

References

1. Compact Data Structures Library (libcds), <http://libcds.recoded.cl/>
2. SPARQL Query Language for RDF. W3C Recomm. (2008), <http://www.w3.org/TR/rdf-sparql-query/>

3. Turtle-Teerse RDF Triple Language. W3C Team Subm. (2008), <http://www.w3.org/TeamSubmission/turtle/>
4. Notation3. W3C Design Issues (1998), <http://www.w3.org/DesignIssues/Notation3>
5. RDF/XML Syntax. W3C Recomm. (2004), <http://www.w3.org/TR/REC-rdf-syntax/>
6. Binary RDF Representation for Publication and Exchange (HDT). W3C Member Subm. (2011), <http://www.w3.org/Submission/2011/03/>
7. Bizer, C., Heath, T., Idehen, K., Berners-Lee, T.: Linked Data On the Web (LDOW 2008). In: Proc. of WWW, pp. 1265–1266 (2008)
8. Brisaboa, N.R., Cánovas, R., Claude, F., Martínez-Prieto, M.A., Navarro, G.: Compressed String Dictionaries. In: Pardalos, P.M., Rebennack, S. (eds.) SEA 2011. LNCS, vol. 6630, pp. 136–147. Springer, Heidelberg (2011)
9. Ding, L., Finin, T.: Characterizing the Semantic Web on the Web. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 242–257. Springer, Heidelberg (2006)
10. Erling, O., Mikhailov, I.: RDF Support in the Virtuoso DBMS. In: Proc. of CSSW, pp. 59–68 (2007)
11. Fernández, J.D., Martínez-Prieto, M.A., Gutierrez, C.: Compact Representation of Large RDF Data Sets for Publishing and Exchange. 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. 193–208. Springer, Heidelberg (2010)
12. González, R., Grabowski, S., Mäkinen, V., Navarro, G.: Practical Implementation of Rank and Select Queries. In: Proc. of WEA, pp. 27–38 (2005)
13. Grossi, R., Gupta, A., Vitter, J.: High-order entropy-compressed text indexes. In: Proc. of SODA, pp. 841–850 (2003)
14. Le-Phuoc, D., Parreira, J.X., Reynolds, V., Hauswirth, M.: RDF On the Go: An RDF Storage and Query Processor for Mobile Devices. In: Proc. of ISWC (2010), <http://iswc2010.semanticweb.org/pdf/503.pdf>
15. Martínez-Prieto, M., Fernández, J., Cánovas, R.: Compression of RDF Dictionaries. In: Proc. of SAC (2012), <http://dataweb.infor.uva.es/sac2012.pdf>
16. Navarro, G., Mäkinen, V.: Compressed Full-Text Indexes. ACM Comput. Surv. 39(1), art. 2 (2007)
17. Neumann, T., Weikum, G.: The RDF-3X Engine for Scalable Management of RDF data. The VLDB Journal 19(1), 91–113 (2010)
18. Weiss, C., Karras, P., Bernstein, A.: Hexastore: Sextuple Indexing for Semantic Web Data Management. Proc. of the VLDB Endowment 1(1), 1008–1019 (2008)
19. Witten, I.H., Moffat, A., Bell, T.C.: Managing Gigabytes: Compressing and Indexing Documents and Images. Morgan Kaufmann (1999)

Impact of Using Relationships between Ontologies to Enhance the Ontology Search Results^{*}

Carlo Allocca, Mathieu d'Aquin, and Enrico Motta

Knowledge Media Institute (KMi), The Open University, Walton Hall,
Milton Keynes, MK7 6AA, United Kingdom
{c.allocca,m.daquin,e.motta}@open.ac.uk

Abstract. Using semantic web search engines, such as Watson, Swoogle or Sindice, to find ontologies is a complex exploratory activity. It generally requires formulating multiple queries, browsing pages of results, and assessing the returned ontologies against each other to obtain a relevant and adequate subset of ontologies for the intended use. Our hypothesis is that at least some of the difficulties related to searching ontologies stem from the lack of structure in the search results, where ontologies that are implicitly related to each other are presented as disconnected and shown on different result pages. In earlier publications we presented a software framework, KANNEL, which is able to automatically detect and make explicit relationships between ontologies in large ontology repositories. In this paper, we present a study that compares the use of the Watson ontology search engine with an extension, WATSON+KANNEL, which provides information regarding the various relationships occurring between the result ontologies. We evaluate WATSON+KANNEL by demonstrating through various indicators that explicit relationships between ontologies improve users' efficiency in ontology search, thus validating our hypothesis.

1 Introduction

From the users' perspective, the most important aspect of Semantic Web Search Engines (SWSEs) [7] is the ability to support the search for ontologies which match their requirements. Indeed, finding ontologies is a complex and creative process which requires a lot of intuition. In addition, it also requires manual analyses of the content of candidate ontologies to choose the ones that are adequate to their intended use. For these reasons, the automatic *Ontology Selection* process has been studied in several different contexts in the recent years [4,5,14,18,22,26,27,29,31] with the aim of improving the methods used to collect, assess and rank candidate ontologies. However, the more user-centric *Ontology Search* process, here defined as the activity of browsing the results from a SWSE to identify the ontologies adequate to the search goal, has not been researched

^{*} This work was funded by the EC IST-FF6-027595 NeOn Project.

extensively until now. Such an activity is becoming crucial for the rapidly growing set of scenarios and applications relying on the reuse of existing ontologies.

Our view is that one of the issues hampering efficient ontology search is that the results generated by SWSEs, such as Watson (<http://watson.kmi.open.ac.uk>), Swoogle (<http://swoogle.umbc.edu>) or Sindice (<http://sindice.com>), are not structured appropriately. These systems return flat lists of ontologies where ontologies are treated as if they were independent from each other while, in reality, they are implicitly related. For example, the query “*Conference Publication*” currently¹ gives 218 ontologies as a result in Watson. The first two pages of results list several items, including http://lstdis.cs.uga.edu/projects/semdis/sweto/testbed_v1_1.owl, http://lstdis.cs.uga.edu/projects/semdis/sweto/testbed_v1_3.owl and http://lstdis.cs.uga.edu/projects/semdis/sweto/testbed_v1_4.owl that represent different versions of the same ontology (*isPrevVersionOf*). Another common situation is when an ontology has been translated in different ontology languages. This is the case in the first (<http://reliant.teknowledge.com/DAML/Mid-level-ontology.owl>) and second (<http://reliant.teknowledge.com/DAML/Mid-level-ontology.daml>) results of the query “*student university researcher*” in Watson or the second (<http://annotation.semanticweb.org/iswc/iswc.daml>) and third (<http://annotation.semanticweb.org/iswc/iswc.owl>) results of the same query in Swoogle. These ontologies are obviously two different encodings of the same model. Analogously, it is not hard to find ontologies connected through other, more sophisticated relations such as different levels of similarity (*isLexicallySimilarTo*, regarding the vocabulary, and *isSyntacticallySimilarTo*, regarding the axioms), as well as the relationship between ontologies that originate from the same provenance, as expressed through their URIs having the same second level domain² name (*ComesFromTheSameDomain*)³.

It is our view that the failure of these systems to provide structured views of the result space hamper the ontology search process as the result space becomes unnecessarily large and full of redundancies. Hence, we have been investigating the hypothesis that *making explicit the relations between ontologies and using them to structure the results of a SWSE system would support a more efficient ontology search process*. In previous publications [13], we presented a software framework, KANNEL, which is able to detect and make explicit relationships between ontologies in large ontology repositories. In this paper we present a comparative study evaluating the improvement brought by extending a SWSE (Watson) by making explicit the relationships between ontologies and presenting them in the results of the ontology search task (the WATSON+KANNEL system). To this purpose, we have used a task-oriented and user-centred approach [20].

¹ That is, on 16/12/2011.

² A second-level domain (SLD) is a domain that is directly below a top-level domain such as com, net and org, see

http://en.wikipedia.org/wiki/Second-level_domain

³ In [13] are reported the formal definitions of the above ontology relations.

Based on a sample of users with a suitable profile, we randomly allocated them into two groups and asked them to perform three ontology search tasks to the best of their ability using either the Watson system alone or WATSON+KANNEL. We then evaluated the differences in the way the two groups performed in these tasks, showing through concrete measures as well as responses to questionnaires that the inclusion of relationship between ontologies in the results of a SWSE system improves the efficiency (and to a smaller extent, the satisfaction) of users involved in ontology search tasks.

In the next section, we briefly describe the systems we used in this study. In Section 4 we describe the methodology adopted to evaluate our hypothesis. In Section 5, we discuss the results of our study. In Section 2, we discuss the relevant related work and in Section 6 we summarise the key contributions of this work and outline our plans for future work.

2 Related Work

The discussion of the related work follows two main directions. The first is related to the process of developing Semantic Web Search Engine systems, while the second concerns different types of relationships between ontologies that have been studied in the literature. Most of the research work related to the ontology search task concerns the development of SWSE systems [7], including: Watson [8], Sindice [28], Swoogle [11], OntoSelect [4], ontokhoj [5] and OntoSearch [32]. All these systems have the aim of collecting and indexing ontologies from the web and providing, based on keywords or other inputs, efficient mechanisms to retrieve ontologies and semantic data. To the best of our knowledge, there is no study regarding the comparison of the above ontology search engines. The most common issues addressed by these systems are *Ontology Selection* - how to identify/select automatically the set of relevant ontologies from a given collection [26,27] - and *Ontology Evaluation* -how to assess the quality and relevance of an ontology [14,27]. Several studies have contributed to the solution of both the above problems, including approaches to ranking ontologies [18] and to select appropriate ontologies [22,29,31]. These works focus on the mechanisms required to support SWSE systems in automatically identifying ontologies from their collections and presenting them in a ranked list to the users. However, *Ontology Search*, as the activity of using a SWSE to find appropriate ontologies, has not been considered before from an user-centric point of view. Furthermore, we can find in literature many works related to the field of Search Engine Usability and how humans interact with search engines [17], but such studies have not yet been applied to SWSEs.

Ontologies are not isolated artefacts: they are, explicitly or implicitly, related to each other. Kleshchev [21] characterised, at a very abstract level, a number of relations between ontologies such as *sameConceptualisation*, *Resemblance*, *Simplification* and *Composition*, without providing formal definitions for them, and without providing mechanisms to detect them. Hefin [19] was the first

to study formally some of the different types of links between ontologies, focusing on the crucial problems of versioning and evolution. Although, these links are available with one of the most used web ontology language (OWL), they are rarely used [2]. Several approaches have focused on the comparison of different versions of ontologies in order to find the differences [16]. In particular, PROMTDIF [25] compares the structure of ontologies and OWLDiff (<http://semanticweb.org/wiki/OWLDiff>) computes the differences by entailment, checking the two set of axioms. SemVersion [30] compares two ontologies and computes the differences at both the structural and the semantic levels. Gangemi in [15] defined the ontology integration as the construction of an ontology C that formally specifies the union of the vocabularies of two other ontologies A and B. The most interesting case is when A and B commit to the conceptualisation of the same domain of interest or of two overlapping domains. In particular, A and B may be related by being *alternative ontologies*, *truly overlapping ontologies*, *equivalent ontologies with vocabulary mismatches*, *overlapping ontologies with disjoint domain*, or *homonymically overlapping ontologies*. There also exists an extensive collection of works, including [10,12,13,33], that propose formal definitions of the ontology mapping concept. Most of them formalise mappings between concepts, relations and instances of two ontologies, to establish an alignment between them, while we focus on relationships between whole ontologies. Finally, studies have targeted ontology comparison in order to identify overlaps between ontologies [23] and many measures exist to compute particular forms of similarity between ontologies [9].

All these studies discuss particular relations separately. While they contribute interesting elements for us to build on, we focus here on assessing the impact of providing various ontology relations to users of SWSE systems.

3 Systems Used

KANNEL is an ontology-based framework for detecting and managing relationships between ontologies for large ontology repositories. Watson is a gateway to the Semantic Web that collects, analyses and gives access to ontologies and semantic data available online. These two systems have already been detailed in [3,8] respectively. Therefore, in this section we only describe the integration of Watson with KANNEL's features to explain in more details how KANNEL is used on top of Watson's repository and integrated into its interface.

WATSON+KANNEL⁴ is an extension of Watson where Watson's ontology space has been processed by KANNEL to detect implicit relationships between ontologies (similarity, inclusion, versioning, common provenance). In addition, two relationship-based mechanisms were added to the Watson ontology search interface (see Fig. 1). They are:

⁴ The WATSON+KANNEL integration can be tested at <http://smartproducts1.kmi.open.ac.uk:8080/WatsonWUI-K>.

What is it? - [Submit URI](#) - [Website](#) - [Blog](#) - [APIs](#)

student versions [Search Watson](#)

Found 1080 semantic documents

versions

- group by
-
- similarity
- lexical similarity
- syntactic similarity
- ✓ versions
- same internet domain

GroupBy Mechanism

Link → [3 similar results](#) 3 other versions 57 results from the same domain

Mechanism

- <http://www.vistology.com/ont/tests/st>

<http://www.vistology.com/ont/>

<http://www.vistology.com/ont/>

<http://www.vistology.com/ont/>

3 similar results 3 other versions
- <http://www.vistology.com/ont/tests/stu>

<http://www.vistology.com/ont/>

<http://www.vistology.com/ont/tests/student2.owl>

<http://www.vistology.com/ont/tests/student3.owl>

3 similar results 3 other versions
- <http://www.cs.vu.nl/~kubbe/webkr/model.daml>

<http://www.cs.vu.nl/~kubbe/webkr/model.daml#Student>

http://www.cs.vu.nl/~kubbe/webkr/model.daml#has_student

http://www.cs.vu.nl/~kubbe/webkr/model.daml#has_audience

http://www.cs.vu.nl/~kubbe/webkr/model.daml#follows_study

http://www.cs.vu.nl/~kubbe/webkr/model.daml#Examination_m_

http://www.cs.vu.nl/~kubbe/webkr/model.daml#has_course_code

http://www.cs.vu.nl/~kubbe/webkr/model.daml#has_to_complete

<http://www.cs.vu.nl/~kubbe/webkr/model.daml#Studies>

<http://www.cs.vu.nl/~kubbe/webkr/model.daml#Coursecode>

http://www.cs.vu.nl/~kubbe/webkr/model.daml#has_completed

[More...](#)

47 results from the same domain
- <http://www.cs.vu.nl/~jkieviet/web/model.daml>

<file://H:/www/web/ideaal.daml#Student>

file://H:/www/web/ideaal.daml#krijgt_les_in

file://H:/www/web/ideaal.daml#krijgt_les_van

file://H:/www/web/ideaal.daml#geeft_les_aan

file://H:/www/web/ideaal.daml#info_uitgever

<file://H:/www/web/proefje.daml#nummer>

<file://H:/www/web/ideaal.daml#letter>

<file://H:/www/web/ideaal.daml#literatuur>

file://H:/www/web/ideaal.daml#is_schrijver_van

<file://H:/www/web/ideaal.daml#bepaalt>

[More...](#)

1 similar results 47 results from the same domain
- <http://www.ontoweb.org/ontology/1>

<http://www.ontoweb.org/ontology/1#student>

<http://www.ontoweb.org/ontology/1#Student>

<http://www.ontoweb.org/ontology/1#PhDStudent>

9 similar results

Fig. 1. Watson's interface with KANNEL's features (i.e., WATSON+KANNEL)

Link mechanism which provides additional ontology links⁵ to the ontologies in the search result space, depending on the relationships they share with others. For example, three ontology links (*3 similar results, 3 other versions and 57 results from the same domain*) have been added to the second result in Fig. 11, meaning that there are three ontologies similar to this one in Watson, three other versions and fifty seven that come from the same second level domain.

GroupBy mechanism which provides a feature to support the re-grouping of ontologies in the search result according to a selected ontology relation. For example, in Fig. 11 the search results have been grouped according to the versioning relationship (as it is the one marked as selected), meaning that different versions of the same ontology appear as a single item, represented by the latest version and links to the others. Different levels of similarity (lexical, regarding the vocabulary, and syntactic, regarding the axioms), as well as common provenance can also be used to group results.

4 Methodology

The general aim of this study is to evaluate whether the inclusion of ontology relationships to structure the results of a SWSE system, as described in the previous section, improves ontology search. In particular, we consider the following two major aspects to be evaluated:

User efficiency: It is envisaged that ontology relations could benefit users by making the ontology search process more efficient, i.e., requiring less time and effort (as measured by indicators such as the number of pages and ontologies visited).

User satisfaction: It is also hypothesised that, by showing better structured and connected search results, users of the WATSON+KANNEL system would be more satisfied and more confident with the results of their ontology search activity.

The comparative study of Watson and WATSON+KANNEL is based on a task-oriented and user-centred approach [20] that involved participants, tasks and data collections as detailed below.

4.1 Participants

Sixteen members of the Open University, from PhD students to senior researchers, participated to the evaluation. They were randomly divided into two groups and asked to perform three ontology search tasks to the best of their ability⁶ using

⁵ The available links are based on the similarity, versioning, common provenance and inclusion relationships as they are described in [13].

⁶ In this work, we considered the tasks to be successfully achieved when the users were satisfied with the ontologies they found.

either Watson or WATSON+KANNEL. We call **W** the group of 7 participants who used Watson only and **W+K** the group of 9 participants who used WATSON+KANNEL. To make the evaluation unbiased, several requirements had to be fulfilled regarding the ability of users to use the specific systems. The following criteria were used to select participants:

1. Experience with computers and web search engines was necessary.
2. A reasonable degree of understanding of “what an ontology is” and how ontologies might be used in concrete scenarios was needed.
3. A general understanding of what a SWSE is and how it could be used to search ontologies (without necessarily having a direct experience of any of them) was required.

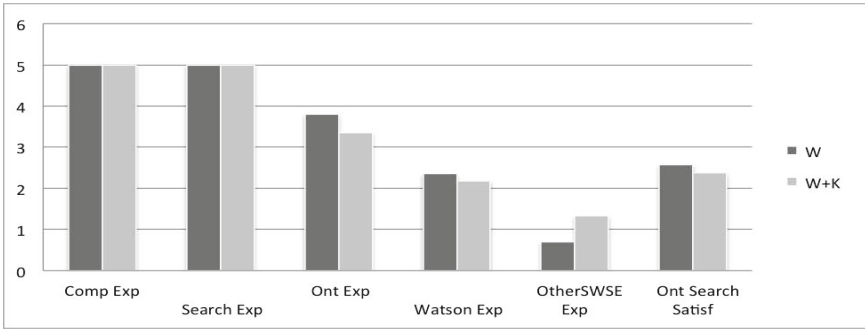


Fig. 2. Profiles of the participants in the two groups **W** and **W+K**. Answers to the corresponding questions could range from 0 to 5. The average is shown.

Fig. 2 gives an overview of the profiles of the participants with respect to their experience of different aspects of the evaluation and our three requirements expressed above. It can be seen from the chart that both groups included participants who declared extensive experience of computers in general (*Comp Exp*) and of Web search engines in particular (*Search Exp*). In both cases participants had on average a medium to good experience with ontologies (*Ont Exp*). They generally knew Watson, even if they had not extensively used it before (if at all, see *Watson Exp*) and some of them had some knowledge of other SWSEs (such as Swoogle and Sindice, see *OtherSWSE*). When asked about their level of satisfaction with SWSEs, especially in supporting ontology search tasks, participants in both groups gave on average a rather neutral answer (*Ont Search Satisf*). Moreover, it is worth mentioning that the participants were not aware of the overall goal of the study and were not part of the Watson and WATSON+KANNEL development teams. Thus, regardless of what system they used, they simply performed the ontology search tasks to their best ability, which was actually the only condition needed to ensure the validity of the evaluation. From these results, it appears that participants in our evaluation matched the target audience, as expressed through the requirements above, and formed two homogeneous groups with respect to their experience of the systems involved.

4.2 Tasks

Each individual participant was asked to realise three different ontology search tasks that are described (as presented to the participants) below:

Task 1 - Modelling. You want to develop an ontology about recipes, where you are going to represent the cooking process reusing existing ontologies from the web. Your task is to use Watson (or WATSON+KANNEL) to find ontologies to cover the topic of recipes and processes dealing with cooking.

Task 2 - Annotation. Consider the two links provided⁷. They are both about the same domain, which is books. Consider them as webpages to which you want to add semantic annotations based on ontologies. To achieve this goal, you need to find ontologies using Watson (or WATSON+KANNEL) that can be used to annotate the above web pages.

Task 3 - Extension. Consider the KMi web page, <http://kmi.open.ac.uk/people/>. As you can see, for each person, there is a RDF description represented using the FOAF ontology (<http://xmlns.com/foaf/spec/>). Imagine that we want to extend the KMi people semantic description including information about the books they have published, the events (conferences, workshops, etc.) they attend and the projects they work on. Your task is to use Watson (or WATSON+KANNEL) to find ontologies that you think would be suitable to be used for such an extension of the existing representation.

4.3 Data Collection

The data for evaluation is collected from two main sources: **questionnaires** and **videos**. Regarding the first: two main questionnaires were designed for this evaluation. One, regarding the background of users, was filled in by the participants before realising the tasks (cf. participant profiles in Fig. 2). The other one, filled in after realising the tasks, included questions regarding the user's satisfaction and confidence in the results obtained for the three ontology search tasks. Questions in this second questionnaire asked how users felt they succeeded with the tasks, how confident they were about having explored a significant part of the relevant ontology space and their overall opinion about the ability of the tool to support them in the tasks. Videos capturing the screen of participants as they realised the given ontology tasks were used to collect concrete information regarding the performance of users in these tasks. Analysing the videos, we were able to measure the average time taken for each task, the number of pages visited and the number of ontologies inspected.

⁷ http://www.amazon.co.uk/Shockwave-Rider-John-Brunner/dp/0345467175/ref=sr_1_1?ie=UTF8&qid=1284638728&sr=1-1-spell and <http://www.booksprice.com/compare.do?inputData=top+gear&Submit2.x=0&Submit2.y=0&Submit2=Search&searchType=theBookName>.

5 Results

The results of our evaluation are presented from both the users' efficiency and satisfaction perspectives. We also discuss how the different ontology relationships included in WATSON+KANNEL were used to support ontology search in the **W+K** group.

5.1 User Efficiency

The diagrams in Fig. 3 show the main results of the typical efficiency of the two groups (**W** and **W+K**) with respect to the three following measures:

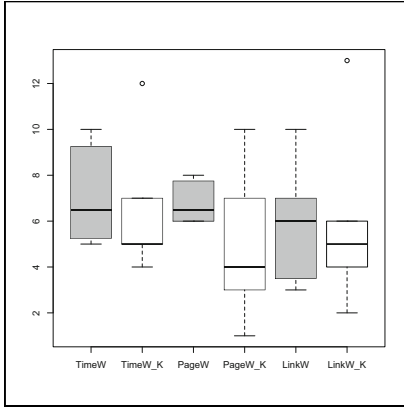
Time is the time in minutes taken to realise the task, i.e., between the beginning of the session, until the user was satisfied with the results obtained. This is the most obvious way to assess the performance of users in ontology search.

Page is the number of pages of results an user would have viewed in order to realise a task. This gives an indication of the effort required in browsing the results of the SWSE to identify relevant ontologies.

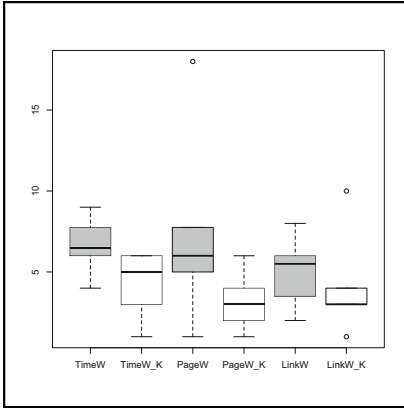
Link is the number of links followed to realise the task, which corresponds to the number of ontologies being inspected.

It clearly appears in Fig. 3 that our hypothesis (that including ontology relationships in the results of an ontology search system would make the ontology search task more efficient) has been confirmed. Indeed, for the indicator Time and Page, the differences between the **W** and **W+K** groups show a significant improvement (taking into account the three tasks, the T-test result for **Time** was 0.017 and for **Page** was 0.013, at significance level $\alpha < 0.05$). While we can observe a difference for the **Link** indicator, this difference was not shown to be statistically significant (T-test result was 0.27). For example, in Task 3, it typically took 2.5 minutes less to achieve the task when using WATSON+KANNEL, and required inspecting only half as many result pages and two third of the links compared to when using Watson only.

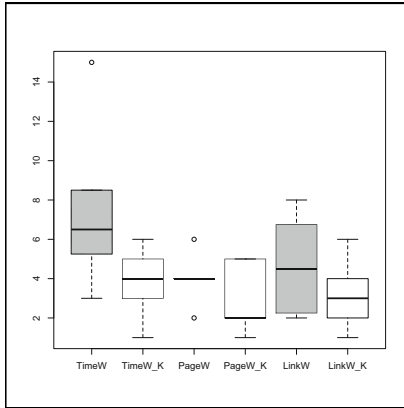
It is worth noticing however that there are significant discrepancies in the results obtained for the three different tasks as shown in Fig. 3. In particular, it appears that the differences between **W** and **W+K** are less significant in Task 1. One of the possible explanations for this phenomenon is that it took some time for participants in group **W+K** to explore and learn the features provided by WATSON+KANNEL that were not present in Watson. To support this interpretation, we analysed the videos corresponding to the group **W+K** to determine to what extent the features provided by KANNEL were used in the three ontology search tasks. As shown in Fig. 4, the features of KANNEL (especially, the ontology relation links) were used significantly less for Task 1 than they were for Tasks 2 and 3 (see charts A and B). It appears that, after Task 1, users learnt to use the ontology relation mechanisms provided by WATSON+KANNEL more efficiently (see charts C – regarding the **Link** mechanism – and D – regarding the **GroupBy** mechanism).



(a) Performance profiles for Task 1.



(b) Performance profiles for Task 2.



(c) Performance profiles for Task 3.

Fig. 3. Performance profiles for the three ontology search tasks, regarding the **Time**, **Page** and **Link** indicators. Each 'box' represent the median (black line), the quartiles (top and bottom of the box), min and max values for each indicator, in each group for each task. Grey boxes correspond to the profiles of the **W** group, white boxes correspond to the **W+K** group.

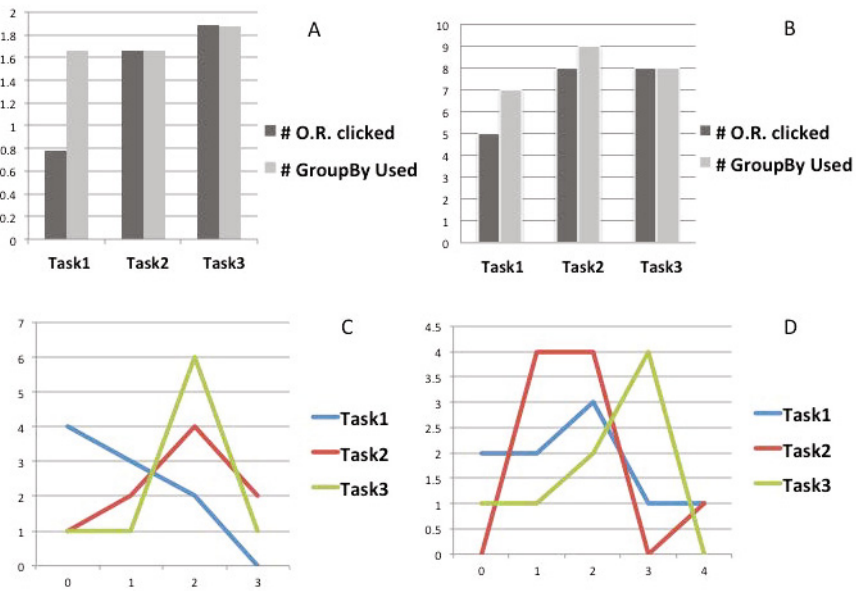


Fig. 4. Use of the ontology relation features in WATSON+KANNEL by group **W+K**. (A) shows the average number of ontology relation links followed and the number of times the “group by” mechanism was applied in each task; (B) shows how many participants used these mechanisms in each task. The diagrams (C) and (D) show the distributions of the number of uses (number of times a feature is used – x axis – by number of users – y axis) for **Link** and **GroupBy** respectively.

5.2 User Satisfaction

Fig. 5 summarises the answers to the five questions asked after the users realised the tasks:

1. Are you satisfied with your success with the tasks?
2. Are you satisfied with the ontologies retrieved through Watson?
3. Are you confident that you have seen most of the relevant, retrieved ontologies?
4. Are you satisfied with the search results?
5. Give an indication of your overall opinion regarding your experience of the tested system.

Each question was given an answer from 0 to 5, 0 being the most negative and 5 the most positive. Surprisingly, considering that users of the Group **W+K** performed significantly better than the ones of the Group **W**, there are only very small differences between their answers to most of the user satisfaction questions. Generally, users ranked both systems highly, with the exception of

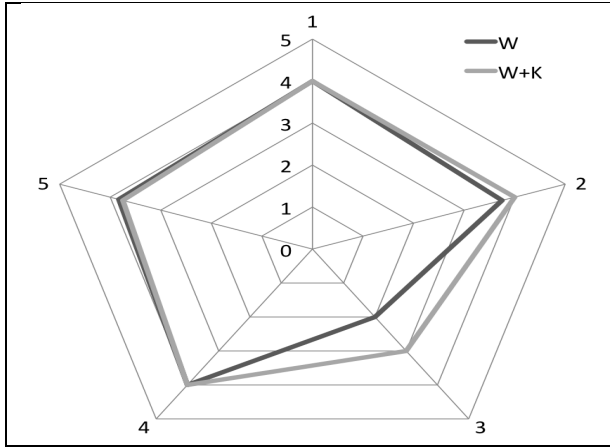


Fig. 5. Median answers to the 5 user satisfaction questions in the two evaluation groups

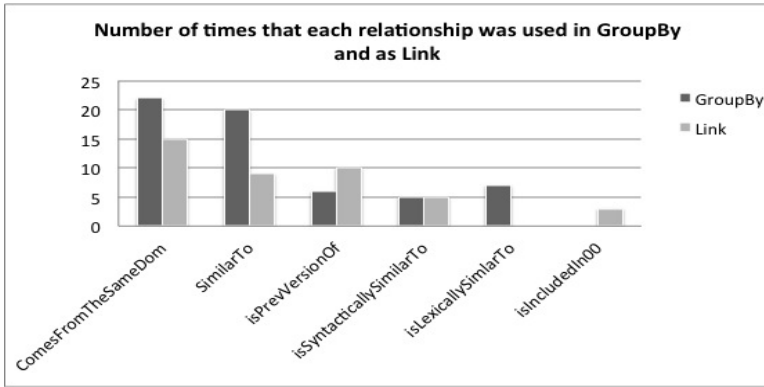


Fig. 6. Use of the 6 relations by the W+K group over the three ontology search tasks

Question 3. Question 3 relates to a clear shortcoming of SWSE systems, i.e., the ability to understand and check the whole set of results. This result is consistent with what reported in [24], which shows that expert users tend to rank usability high in systems, regardless of the task performance.

5.3 Analysis of the Ontology Relationships

In addition to the users' efficiency and satisfaction, we briefly discuss the extent to which different relations were used to support ontology search. Measures of the use of the **Link** and **GroupBy** functions in the WATSON+KANNEL system over the three tasks in group W+K are summarised in the charts of Fig. 6.

There, it is shown that *ComesFromTheSameDomain* and *SimilarTo* (especially in groupBy) are the two most used ontology relationships. One could argue that this is due to the fact that these relations are significantly more present in the Watson repository than any of the others. However, we cannot actually establish a clear correlation between the number of instances of a relationship and its use for the three tasks. Indeed, *ComesFromTheSameDomain* is only second in number of relationships (with about four million instances), while *SimilarTo* is two orders of magnitude more present (more than 100 million instances). More importantly, the third most popular relationship, *isPrevVersion*, is the least present in the repository with only about 2,800 instances, compared to 14,000 for *isIncludedIn* (*isSyntacticallySimilarTo* and *isLexicallySimilarTo* overlap to a large extent with *SimilarTo* and with each other, therefore both have more than 100 million instances). A possible explanation for these results is therefore that the three popular relationships (*ComesFromTheSameDomain*, *SimilarTo* and *isPrevVersionOf*) represent notions that are useful in supporting ontology search. In other terms, being able to discover ontologies that have the same provenance, have a high degree of domain overlap, or represent evolutions of other ontologies appears more natural to users than the use of the *isIncludedIn* relation. We can also envisage that these three relations are more directly understandable and verifiable than the three others which were less used. Another observation is that relationships with very large numbers of instances (and therefore appearing for many ontologies in the search results) are generally used more with the **GroupBy** mechanism than with the **Link** mechanism. This appears natural as ‘rare’ relationships have less impact than more common ones when used to structure the search results through the **GroupBy** mechanism.

6 Conclusion and Future Work

In this paper, we have presented an evaluation of our hypothesis that, in the context of a Semantic Web Search Engine, making explicit the relationships between ontologies and using them to structure the results of a SWSE system leads to a more efficient ontology search process. This evaluation was based on the comparative study of the Watson search engine and its extension WATSON+KANNEL through a task-oriented and user-centred approach. Both the feedback obtained from questionnaires and concrete performance measures show an improvement in efficiency when performing the ontology search task with WATSON+KANNEL. The current study also provides the basis for several promising research directions. Firstly, we are interested in extending the evaluation of our hypothesis by conducting a comparative study of other ontology search engines such as Swoogle and Sindice with their KANNEL’s extension (SWOOGLE+KANNEL and SINDICE+KANNEL respectively). Secondly, we are interested in building on this work to provide novel ontology ranking solutions. Thirdly, this work also provides the basis for novel empirical studies. In particular, we plan to analyse relationships between ontologies at scale to understand better the ontology engineering practices behind the modelling process. For example, different ontology versions provide data about the development process of ontologies, showing how

they reach stability or adapt to changes in the domain. Thus, once we have detected the links between different versions of ontologies, it becomes possible to explore how such ontologies evolve on the Semantic Web, in particular with the aim of discovering relevant high level *ontology evolution patterns*, which can be used to focus ontology search around notions such as ‘stability’ and ‘activity’. Finally, from a practical point of view and as part of our broader work on building a framework for the management of relationships between ontologies (see e.g., [3]), one of our future directions of research is to extend the set of relationships between ontologies that can be considered by our system. One of the most interesting aspects here concerns providing mechanisms to explore not only single, atomic relations between ontologies, but also the relations derived from the combination of others (e.g., *compatibility* and *disagreement* [6]).

References

1. Allocca, C.: Making explic semantic relations between ontologies in large ontology repositories. PhD Symposium at the European Semantic Web Conference (2009)
2. Allocca, C.: Automatic Identification of Ontology Versions Using Machine Learning Techniques. 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. 352–366. Springer, Heidelberg (2011)
3. Allocca, C., d'Aquin, M., Motta, E.: Door - towards a formalization of ontology relations. In: Proc. of the Inter. Conf. on Knowledge Engineering and Ontology Development, KEOD, pp. 13–20 (2009)
4. Buitelaar, P., Eigner, T., Declerck, T.: Ontoselect: A dynamic ontology library with support for ontology selection. In: Proceedings of the Demo Session at the International Semantic Web Conference (2004)
5. Chintan, P., Kaustubh, S., Yugyung, L., Park, E.K.: Ontokhoj a semantic web portal for ontology searching, ranking, and classification. In: Proc. 5th ACM Int. Workshop on Web Information and Data Management, New Orleans, Louisiana, USA, pp. 58–61 (2003)
6. d'Aquin, M.: Formally measuring agreement and disagreement in ontologies. In: K-CAP, pp. 145–152. ACM (2009)
7. d'Aquin, M., Ding, L., Motta, E.: Semantic web search engines. In: Domingue, J., Fensel, D., Hendler, J.A. (eds.) Handbook of Semantic Web Technologies, pp. 659–700. Springer, Heidelberg (2011)
8. d'Aquin, M., Motta, E.: Watson, more than a semantic web search engine. *Semantic Web Journal* 2(1), 55–63 (2011)
9. David, J., Euzenat, J.: Comparison between Ontology Distances (Preliminary Results). In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 245–260. Springer, Heidelberg (2008)
10. David, J., Euzenat, J., Šváb-Zamazal, O.: Ontology Similarity in the Alignment Space. 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. 129–144. Springer, Heidelberg (2010)

11. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V.C., Sachs, J.: Swoogle: A Search and Metadata Engine for the Semantic Web. In: Proc. of the 13th ACM Conf. on Information and Knowledge Management. ACM Press (November 2004)
12. Ehrig, M.: Ontology Alignment: Bridging the Semantic Gap. *Semantic Web and Beyond*, vol. 4. Springer, New York (2007)
13. Euzenat, J.: Algebras of Ontology Alignment Relations. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 387–402. Springer, Heidelberg (2008)
14. Gangemi, A., Catenacci, C., Ciaramita, M., Lehmann, J.: A theoretical framework for ontology evaluation and validation. In: Proceedings of SWAP 2005, the 2nd Italian Semantic Web Workshop, Trento, Italy, December 14-16. CEUR Workshop Proceedings (2005)
15. Gangemi, A., Pisanelli, D.M., Steve, G.: An overview of the onions project: Applying ontologies to the integration of medical terminologies. Technical report. ITBM-CNR, V. Marx 15, 00137, Roma, Italy (1999)
16. Gonçalves, R.S., Parsia, B., Sattler, U.: Analysing multiple versions of an ontology: A study of the nci thesaurus. In: *Description Logics* (2011)
17. Gordon, M., Pathak, P.: Finding information on the World Wide Web: the retrieval effectiveness of search engines. *Information Processing & Management* 35(2), 141–180 (1999)
18. Alani, H., Brewster, C., Shadbolt, N.: Ranking Ontologies with AKTiveRank. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 1–15. Springer, Heidelberg (2006), <http://eprints.ecs.soton.ac.uk/12921/01/iswc06-camera-ready.pdf>
19. Heflin, J., Pan, Z.: A Model Theoretic Semantics for Ontology Versioning. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 62–76. Springer, Heidelberg (2004)
20. Ingwersen, P.: *Information Retrieval Interaction*. Taylor Graham (1992)
21. Kleshchev, A., Artemjeva, I.: An analysis of some relations among domain ontologies. *Int. Journal on Inf. Theories and Appl.* 12, 85–93 (2005)
22. Lozano-Tello, A., Gómez-Pérez, A.: ONTOMETRIC: A Method to Choose the Appropriate Ontology. *Journal of Database Management* 15(2) (April-June 2004)
23. Maedche, A., Staab, S.: Comparing ontologies-similarity measures and a comparison study. In: Proc. of EKAW 2002 (2002)
24. Motta, E., Mulholland, P., Peroni, S., d’Aquin, M., Gomez-Perez, J.M., Mendez, V., Zablith, F.: A Novel Approach to Visualizing and Navigating 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. 470–486. Springer, Heidelberg (2011)
25. Noy, N.F., Musen, M.A.: Promptdiff: A fixed-point algorithm for comparing ontology versions. In: 18th National Conf. on Artificial Intelligence, AAAI (2002)
26. Sabou, M., Lopez, V., Motta, E.: Ontology Selection for the Real Semantic Web: How to Cover the Queen’s Birthday Dinner? In: Staab, S., Svátek, V. (eds.) EKAW 2006. LNCS (LNAI), vol. 4248, pp. 96–111. Springer, Heidelberg (2006)
27. Sabou, M., Lopez, V., Motta, E., Uren, V.: Ontology selection: Ontology evaluation on the real semantic web. In: 15th International World Wide Web Conference (WWW 2006), Edinburgh, Scotland, May 23-26 (2006)

28. 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), <http://dblp.uni-trier.de/db/conf/semweb/iswc2007.html#TummarelloD007>
29. Hong, T.-P., Chang, W.-C., Lin, J.-H.: A Two-Phased Ontology Selection Approach for Semantic Web. In: Khosla, R., Howlett, R.J., Jain, L.C. (eds.) KES 2005. LNCS (LNAI), vol. 3684, pp. 403–409. Springer, Heidelberg (2005)
30. Volkel, M.: D2.3.3.v2 SemVersion Versioning RDF and Ontologies. EU-IST Network of Excellence (NoE) IST-2004-507482 KWEB
31. Xiaodong, W., Guo, L., Fang, J.: Automated ontology selection based on description logic. In: CSCWD, pp. 482–487 (2008)
32. Zhang, Y., Vasconcelos, W., Sleeman, D.H.: Ontosearch: An ontology search engine. In: Bramer, M., Coenen, F., Allen, T. (eds.) SGAI Conf., pp. 58–69. Springer, Heidelberg (2004)
33. Zimmermann, A., Krötzsch, M., Euzenat, J., Hitzler, P.: Formalizing ontology alignment and its operations with category theory. In: Proceeding of the 4th Inter. Conf. on Formal Ontology in Information Systems, FOIS, pp. 277–288. IOS Press (2006)

Enhancing OLAP Analysis with Web Cubes

Lorena Etcheverry¹ and Alejandro A. Vaisman²

¹ Instituto de Computación, Universidad de la República, Uruguay

lorenae@fing.edu.uy

² Université Libre de Bruxelles

avaisman@ulb.ac.be

Abstract. Traditional OLAP tools have proven to be successful in analyzing large sets of enterprise data. For today's business dynamics, sometimes these highly curated data is not enough. External data (particularly web data), may be useful to enhance local analysis. In this paper we discuss the extraction of multidimensional data from web sources, and their representation in RDFS. We introduce Open Cubes, an RDFS vocabulary for the specification and publication of multidimensional cubes on the Semantic Web, and show how classical OLAP operations can be implemented over Open Cubes using SPARQL 1.1, without the need of mapping the multidimensional information to the local database (the usual approach to multidimensional analysis of Semantic Web data). We show that our approach is plausible for the data sizes that can usually be retrieved to enhance local data repositories.

1 Introduction

Business intelligence (BI) comprises a collection of techniques used for extracting and analyzing business data, to support decision-making. Decision-support systems (DSS) include a broad spectrum of analysis capabilities, from simple reports to sophisticated analytics. These applications include On-Line Analytical Processing (OLAP) [9], a set of tools and algorithms for querying large multidimensional databases usually called data warehouses (DW). Data in a DW come from heterogeneous and distributed operational sources, and go through a process, denoted ETL (standing for Extraction, Transformation, and Loading). In OLAP, data are usually perceived as a *cube*. Each cell in this data cube contains a measure or set of measures representing facts and contextual information (the latter called *dimensions*). For some data-analysis tasks (e.g., worldwide price evolution of a certain product), the data contained the DSS do not suffice. External data sources (like the web) can provide useful multidimensional information, although usually too volatile to be permanently stored in the DW. We now present, through a use case, the research problems that appear in this scenario, and our approach for a solution to some of them.

1.1 Motivation and Problem Statement

A new electronics retail store is running a promotional campaign to improve sales on an specific segment of the digital cameras market: amateur digital SLR

Product		Locat.		Time							
				November 2011				December 2011			
				profit	#sales	unitPrice	unitCost	profit	#sales	unitPrice	unitCost
Canon	T3i	Kit 18-55	NJ	870	10	870	783	736	8	875	783
			NY	1044	12	870	783	609	7	870	783
		Body only	NJ	340	4	850	765	375	5	840	765
			NY	425	5	850	765	300	4	840	765
	T3	Kit 18-55	CA	780	13	460	400	480	8	460	400
			NJ	1200	15	480	400	560	7	480	400
		Body only	CA	400	8	500	450	250	5	500	450
			NY	385	7	505	450	330	6	505	450
Nikon	D3100	Kit 18-55	CA	630	10	610	547	945	15	610	547
			NY	732	12	608	547	366	6	608	547
			WA	340	5	615	547	189	3	610	547
		Kit 55-200	NY	750	6	725	600	1500	12	725	600
			CA	400	8	500	450	250	5	500	450
			NY	385	7	505	450	330	6	505	450
	D5100	Kit 18-55	NJ	1215	15	810	729	688	8	815	729
			CA	456	6	746	670	456	6	746	670
		Body only	CA	456	6	746	670	456	6	746	670
			CA	456	6	746	670	456	6	746	670

Fig. 1. A Sales Data Cube

cameras. The company sales products from several manufacturers and wants to find out candidates for “best deals” kind of offer. In today’s business, web information is crucial for this. Price policies must take into account current deals found on the Internet (e.g., number of available offers, shipping policies, expected delivery time), as well as user opinions and product features. Jane, the data analyst of the company manually searches the web, querying different sites, and building spreadsheets with the collected data. Then she analyzes local data at the DSS, together with data from web sources. This procedure is not only inefficient but also imprecise. Jane needs flexible and intelligent tools to get an idea of what is being offered on the web. We propose to make Jane’s work more productive, by semi-automatically extracting multidimensional information from web data sources. This process produces what we denote **web cubes**, which can then be related to local OLAP cubes through a set of operators that we study in this paper. A key assumption is that web cubes only *add* knowledge for decision-making, and are not aimed at replacing precise information obtained from traditional DSS. Thus, we do not need these data to be complete, not even perfectly sound: Jane only needs a “few good answers” [17] to enhance her analysis, and our approach takes this into account. In a nutshell, web cubes are data cubes (obtained from web data sources) expressed using an RDF [10] vocabulary. We show through an use case how web cubes could be used to enhance existing DSS.

The case starts with Jane using her **local** DSS to analyze sales of digital cameras. From local cubes she produces a multidimensional report (Figure 1), actually a data cube with dimensions **Product**, **Geography**, and **Time**, and measures **profit**, **#sales**, **unitPrice**, and **unitCost**. From the report, Jane identifies that the sales of Canon T3 and T3i cameras have dropped in December. She conjectures that probably these products are being offered on the web at better prices, thus she decides to build a web cube to retrieve information about offers of these camera models. To start the process of building web cubes, Jane specifies her information requirements, which in this particular case are: price, delivery time

and shipping costs of new Canon T3 and T3i DSLR cameras. She will try to obtain sales facts with these three measures, if possible with the same dimensions than the local cube. We assume that there is a catalogue of web data sources, with metadata that allows deciding which sources are going to be queried, the query mechanisms available for each source, and the format of the results.

Web data are available in many formats. Each one of these formats can be accessed using different mechanisms. For example RDF data can be published via SPARQL [16] endpoints, or extracted from HTML pages that publish RDFa [1] (among other formats), while XML data may be the result of querying RESTful web services (also known as web APIs). Tabular data may be extracted from HTML tables or retrieved from data sharing platforms, such as Google Fusion Tables¹. In this paper we do not deal with the problem of retrieving web cubes. Well-known Information Retrieval and Natural Language Processing techniques can be used for this. For integrating the information retrieved from the data sources, and for representing the web cubes that are built after data extraction, we propose to use RDF as the data model. For the latter task, we devised a vocabulary called *Open Cubes* that we present in Section 3. It is highly possible that not all of Jane's requirements can be satisfied. For example, data could be obtained at an aggregation level different from the requested one. Or may be incomplete. We do not deal with these issues in this paper. Continuing with our use case, let us suppose that from www.overstock.com, Jane obtains the following following RDF triples.

```
@prefix dc: <http://purl.org/dc/>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix schema: <http://schema.org/>

<http://www.overstock.com/.../5700610/product.html> dc:title
  "Canon EOS Rebel T3i 18 D55mm IS II Digital SLR Camera Kit Overstock" .
_:node16bt8fdb2x1 rdf:type schema:Product .
_:node16bt8fdb2x1 schema:name
  "Canon EOS Rebel T3i 18 D55mm IS II Digital SLR Camera Kit" .
_:node16bt8fdb2x1 schema:offers _:node16bt8fdb2x2 .
_:node16bt8fdb2x2 rdf:type schema:Offer .
_:node16bt8fdb2x2 schema:price "USD 850.82" .
```

From these triples, a web cube is built (represented using the Open Cube vocabulary). Figure 2 shows this cube, in report format. Note that in this example, the new cube has the same dimensions than the cube in Figure 1, but different measures. Also notice that we have maximized the number of returned results, leading to the presence of null values (denoted by '-' in Figure 2), which should be replaced by appropriate values (e.g., 'unknown state') to guarantee the correctness of the results when performing OLAP operations. Jane now wants to compare the price of each product in the local cube, with the price for the same product in the web cube. This requires using OLAP operators like roll-up, slice, dice, or drill-across, *over web Cubes*. Jane then realizes that in the Geography dimension of the web cube, data are presented per city instead of per state (which is the case in the local cube). Thus, she needs to transform the web cube to the same level of detail as the local cube, taking both cubes to the country level, using a roll-up operation in the Geography dimension. After this, both cubes can be merged, and Jane can compare the prices in the local cube with those found

¹ <http://www.google.com/fusiontables/Home/>

Product					Geography		Time			
							Q3-2011			
							unitPrice	deliveryTime	shippingCost	
SLR Camera	Canon	T3i	Kit 18-55	USA	-	-	850.82	10	0	(1)
				NY	Amityville	799.95	-	19.95	(2)	
		-	-	760.00	5	0	(3)			
	T3	Kit 18-55	Body only	USA	-	-	672.99	-	0	(4)
			-	NJ	Somerset	466.82	-	-	(5)	
			-	-	476.99	7	0	(6)		

Fig. 2. A web cube in report format

Product				Geography		Time					
						Q3-2011					
						profit	#sales	unitPrice	unitCost	deliveryTime	shippingCost
Canon	T3i	Kit 18-55	USA	672.5	15	872.5	783.0	-	-	-	-
				-	-	803.6	-	7.5	19.95		
		Body only	USA	395.0	15	843.3	765.0	-	-		
	T3	Kit 18-55	USA	520.0	15	470.0	400.0	-	-		
			-	-	471.9	-	7.0	0.0			
			Nikon	D3100	Kit 18-55	USA	500.0	24	609.3	547.0	-
D5100	Kit 18-55	Kit 55-200	USA	1500	12	725	600	-	-		
		Body only	USA	290.0	11	502.5	450.0	-	-		
		Body only	USA	688	8	815	729	-	-		
		Body only	USA	456	6	746	670	-	-		

Fig. 3. Results of merging local and web cubes

on the Internet (grey rows). Figure 3 shows the result (Section 4 shows how web cubes can be mapped to the multidimensional model of the local cube).

Contributions. The following research questions arise in the scenario described above: Is it possible to use web data to enhance local OLAP analysis, without the burden of incorporating data sources and data requirements into the existent DSS life-cycle? What definitions, data-models, and query mechanisms are needed to accomplish these tasks? Our main goal is to start giving answers to the some of these questions. Central to this goal is the representation and querying of multidimensional data over the Semantic Web. Therefore, our main contributions are: (a) We introduce Open Cubes, a vocabulary specified using RDFS that allows representing the schema and instances of OLAP cubes, which extends and makes workable other similar proposals, since it is not only devised for data publishing, but for operating over RDF representation of multidimensional data as well (Section 3); (b) We show how typical OLAP operators can be expressed in SPARQL 1.1 using the vocabulary introduced in (a). We give an algorithm for generating SPARQL 1.1. CONSTRUCT queries for the OLAP operators, and show that implementing these operators is feasible. The basic assumption here is that web cubes are composed of a limited number of instances (triples) of interest (Section 4); (c) We sketch a mapping from a web cube to the multidimensional (in what follows, MD) model, in order to be able to operate with the local cubes. We do this through an example that shows how web cubes can be exported to the local system, using the Mondrian OLAP server² (Section 4).

² <http://mondrian.pentaho.com/documentation/schema.php>

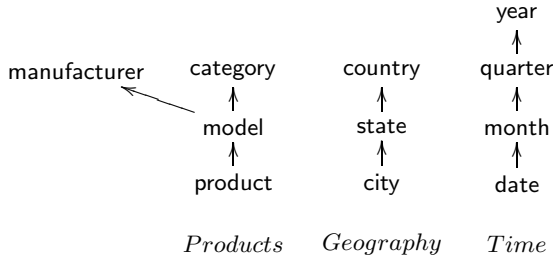
2 Preliminaries

RDF and SPARQL. The Resource Description Framework (RDF) [10] is a data model for expressing assertions over resources identified by an universal resource identifier (URI). Assertions are expressed as triples *subject - predicate - object*, where *subject* are always resources, and *predicate* and *object* could be resources or strings. *Blank nodes* are used to represent anonymous resources or resources without an URI, typically with a structural function, e.g., to group a set of statements. Data values in RDF are called *literals* and can only be *objects*. A set of RDF triples or *RDF dataset* can be seen as a directed graph where *subject* and *object* are nodes, and *predicates* are arcs. Many formats for RDF serialization exist. The examples presented in this paper use Turtle [2]. RDF Schema (RDFS) [3] is a particular RDF vocabulary where a set of reserved words can be used to describe properties like attributes of resources, and to represent relationships between resources. Some of these reserved words are `rdfs:range` [`range`], `rdfs:domain` [`dom`], `rdf:type` [`type`], `rdfs:subClassOf` [`sc`], and `rdfs:subPropertyOf` [`sp`].

SPARQL is the W3C standard query language for RDF [16]. The query evaluation mechanism of SPARQL is based on subgraph matching: RDF triples are interpreted as nodes and edges of directed graphs, and the query graph is matched to the data graph, instantiating the variables in the query graph definition. The selection criteria is expressed as a graph pattern in the **WHERE** clause, consisting basically in a set of triple patterns connected by the ‘.’ operator. The SPARQL 1.1 specification [5], with status of working draft at the moment of writing this paper, extends the power of SPARQL in many ways. Particularly relevant to our work is the support of aggregate functions and the inclusion of the **GROUP BY** clause.

OLAP. In OLAP, data are organized as hypercubes whose axes are *dimensions*. Each point in this multidimensional space is mapped through *facts* into one or more spaces of *measures*. Dimensions are structured in *hierarchies* of *levels* that allow analysis at different levels of aggregation. The values in a dimension level are called *members*, which can also have properties or *attributes*. Members in a dimension level must have a corresponding member in the upper level in the hierarchy, and this correspondence is defined through so-called rollup functions. In our running example we have a cube with sales data. For each sale we have four measures: quantity of products sold, profit, price and cost per product (see Figure 1, which shows a cube in the form of a report). We have also three dimensions: **Product**, **Geography** (geographical location of the point of sale), and **Time**. Figure 4 shows a possible schema for each dimension, and for the sales facts. We can see that in dimension **Geography**, cities aggregate over states, and states over countries. A well-known set of operations are defined over cubes. We present (for clarity, rather informally) some of these operations next, following [9] and [19].

Roll-Up. Summarizes data at a higher level in the hierarchies of a dimension. Given a cube \mathcal{C} , a dimension $D \in \mathcal{C}$, such that d_l is the lowest level of D in \mathcal{C} ; a dimension level $du \in D$ such that $d_l \preceq d_u$ (meaning that d_l is lower than du



SALES[product, city, date] → [profit, #sales, unitPrice, unitCost]

Fig. 4. Examples of dimensions (above) and fact schemas (below)

in the hierarchy of D), and a set of aggregate functions f , $\text{Roll-Up}(C, D, d_u, f)$ returns a new cube C' whose dimensions and levels are the same than in C , except for the lowest level of D in C' , which becomes d_u . Measures in C' are the result of applying each function in f to the corresponding measures in C . Aggregation is performed according with the rollup function associated with the relation $d_l \preceq d_u$. As an example, consider the cube C in Figure 5a with dimensions Product and Time and measure qtySold. Figure 5b shows the result of $\text{Roll-Up}(C, \text{Time}, \text{month}, \text{SUM})$. (If the cube has more than one measure, and different aggregate functions f_i are applied, pairs (measure, f_i) must be specified.

Slice. Removes one dimension from a cube C . Intuitively, given a cube C , a dimension $D \in C$, and an aggregation function f , the result of applying the Slice operation is a new cube $C' = \text{Slice}(C, D, f)$ whose dimensions are the ones of C , except for D . Measures in C' are the result of applying the aggregation function f to the measure in C . Figure 5c shows the result of $\text{Slice}(C, \text{Product}, \text{SUM})$. (The same as above holds if there is more than one measure in C).

Dice. Selects a subset of the instances of a cube. Intuitively, given a cube C , a dimension $D \in C$, and a formula σ over the levels in D , the operation Dice is a new cube $C' = \text{Dice}(C, D, \sigma)$ whose dimensions are same as in C , and such that the elements in C are the ones that satisfy σ . Assuming that $date1 \leq date2 \leq date3$, Figure 5d shows the result of $\text{Dice}(C, \text{Time}, date > date1)$.

3 OLAP Cubes Specification in RDF

We now introduce Open Cubes, a vocabulary specified using RDFS that allows representing the schema and instances of OLAP cubes. Recently, an RDF vocabulary called Data Cube [4] has been proposed that supports the exchange of statistical data according to the ISO standard SDMX. [3] Although OLAP and statistical databases are very similar concepts, they differ in the problems

³ <http://sdmx.org/>

		prod1	prod2
month1	date1	1	3
	date2	3	2
month2	date3	4	3

(a) Cube C

	prod1	prod2
month1	4	5
month2	4	3

(b) $RollUp(C, Time, month, sum)$

month1	date1	4
	date2	5
month2	date3	7

(c) $Slice(C, Product, sum)$

		prod1	prod2
month1	date2	3	2
month2	date3	4	3

(d) $Dice(C, Time, date > date1)$

Fig. 5. Applying OLAP operations to a cube

they address [18]. In particular, the Data Cube vocabulary does not provide the means to perform OLAP operations over data. This and other issues will be further discussed in Section 5.

Open Cubes is based on the multidimensional data model presented in [6], whose main concepts are dimensions and facts [4].

Dimensions have a schema and a set of instances; the schema contains the name of the dimension, a set of *levels* and a *partial order* defined among them; a dimension level is described by **attributes**; a dimension instance contains a set of partial functions, called *roll-up functions*, that specify how *level members* are aggregated. **Facts** also have a schema and instances; the former contains the name of the fact, a set of levels and a set of *measures*; the latter is a partial function that maps points of the schema into measure values. Figure 6 graphically presents the most relevant concepts in the Open Cubes vocabulary, where bold nodes represent classes and regular nodes represent properties. Labelled directed arcs between nodes represent properties with a defined domain and range among concepts in the vocabulary. We omit properties whose range is a literal value.

In Open Cubes, the class `oc:Dimension` and a set of related levels, modelled by the `oc:Level` property, represent dimension schemas. A partial order among levels is defined using properties `oc:parentLevel` and `oc:childLevel`, while the attributes of each level member are modelled using the `oc:Attribute` property. A fact schema is represented by the class `oc:FactSchema` and a set of levels and measures, which are modelled using the properties `oc:Level` and `oc:Measure` respectively. For each measure the aggregation function that can be used to compute the aggregated value of the measure, can be stated using the `oc:hasAggFunction` property. As an example, Figure 8 shows RDF triples (in Turtle notation) that represent the schemas of the **Products** dimension and the **Sales** fact from the report in Figure 7, using the Open Cubes vocabulary. The prefixes `oc` and `eg` represent the Open Cube vocabulary and the URI of the RDF graph of this example, respectively.

Dimension instances are modelled using a set of level members, which are represented by the `oc:LevelMember` class. The properties `oc:parentLevelMember`

⁴ We could have used a more complex model, like [7]. However, the chosen model is expressive enough to capture the most usual OLAP features.

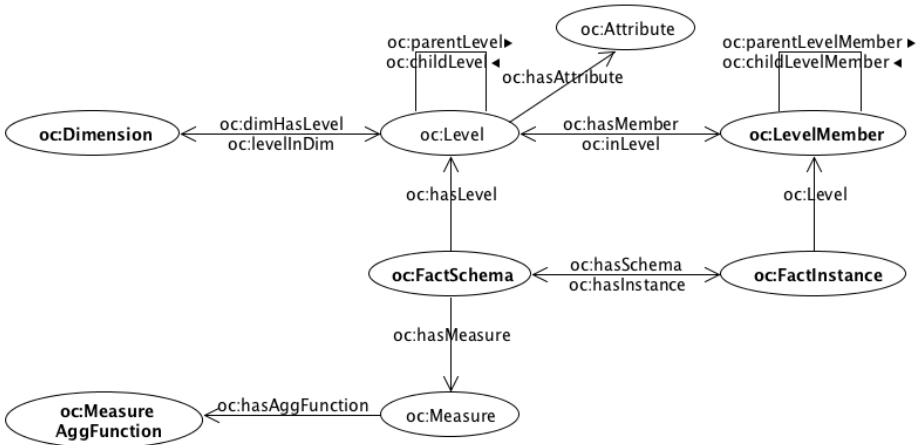


Fig. 6. Open Cubes vocabulary: classes and properties

				Time						
				Q3-2011						
				December 2011						
				date1		date2				
Product				Geography		price	qtySold	price	qtySold	
DSLR Camera	Canon	T3i	Kit 18-55	USA	OR	Portland	714.54	2	714.54	3
		T3	Kit 18-55	USA	NJ	Somerset	466.82	5	466.82	5
						Jersey City	480	4	480	3

Fig. 7. Sample sales fact instances

<pre> eg:products rdf:type oc:Dimension; oc:dimHasLevel eg:product; oc:dimHasLevel eg:model; oc:dimHasLevel eg:manufacturer; oc:dimHasLevel eg:category. eg:product rdf:type oc:Level. eg:model rdf:type oc:Level. eg:manufacturer rdf:type oc:Level. eg:category rdf:type oc:Level. eg:product oc:parentLevel eg:model. eg:model oc:parentLevel oc:manufacturer. eg:model oc:parentLevel eg:category. </pre>	<pre> eg:sales rdf:type oc:FactSchema; oc:hasLevel eg:product; oc:hasLevel eg:city; oc:hasLevel eg:date; oc:hasMeasure eg:price; oc:hasMeasure eg:qtySold; eg:price rdf:type oc:Measure; oc:hasAggFunction avg. eg:qtySold rdf:type oc:Measure; oc:hasAggFunction sum. </pre>
Products dimension schema	Sales fact schema

Fig. 8. Expressing schemas using Open Cubes vocabulary

and `oc:childLevelMember` represent the roll-up functions that specify the navigation among level members. Instances of `oc:FactInstance` class represent fact instances. The set of level members and the values of each measure are related to the fact instance using pre-defined properties of type `Level` and `Measure`

<pre>'t3i kit 18-55' oc:inLevel eg:product; oc:parentLevelMember 't3i'. 't3i' oc:inLevel eg:model; oc:parentLevelMember 'Canon'; oc:parentLevelMember 'DSLR camera'. 't3 kit 18-55' oc:inLevel eg:product; oc:parentLevelMember 't3'. 't3' oc:inLevel eg:model; oc:parentLevelMember 'Canon'; oc:parentLevelMember 'DSLR camera'. 'Canon' oc:inLevel eg:manufacturer. 'DSLR camera' oc:inLevel eg:category.</pre>	<pre>eg:sales_i1 rdf:type oc:FactInstance ; oc:hasSchema eg:sales ; eg:product 't3i kit 18-55' ; eg:date 'date1' ; eg:city 'Portland' ; eg:price '714.54'^^xsd:decimal; eg:qtySold '3'^^xsd:integer . eg:sales_i2 rdf:type oc:FactInstance ; eg:sales_i3 rdf:type oc:FactInstance ; </pre>
Products dimension instances	Sales fact instances

Fig. 9. Expressing instances using Open Cubes vocabulary

respectively. For example, Figure 9 shows how the instances in Figure 7 can be represented using the Open Cubes vocabulary. Is it worth noting that subjects in RDF triples should be either blank nodes or URIs. For clarity, we use constant values between quotation marks to represent the identifier of each level member. These constant values should be replaced by URIs that uniquely identify each of the level members. The `oc:hasFactId` property allows to provide a literal that uniquely identifies each fact instance within a collection of cubes or multi-dimensional database. Due to space reasons we omit the `oc:childLevelMember` relationships between the level members, and only show one complete fact instance RDF representation (the first tuple in Figure 7) .

4 Operating with Web Cubes

The operators introduced in Section 2 can be implemented in SPARQL 1.1 as we show next. Let us start with a couple of examples.

Roll-up: This operation encompasses two tasks: (i) creating the schema of the new cube; (ii) populating it. For instance, consider the *Sales* cube representation in Open Cubes (Figures 8 and 9), and a new cube *SalesByMonth* = *Roll-Up(Sales, Time, month, price, AVG, qty, SUM)*. Figure 10 shows the specification in Open Cubes of the *SalesByMonth* schema and a SPARQL 1.1 query that populates it. It is important to remark that new IRIs must be generated to identify each of the fact instances.

Slice: Slicing out the *Geography* dimension from the *Sales* cube of in Figures 8 and 9, returns a new cube *SalesWithoutGeo* that satisfies *SalesWithoutGeo* = *Slice(Sales, Geography, AVG, SUM)*. Figure 11 shows the Open Cubes specification of the *SalesWithoutGeo* schema, and a SPARQL 1.1 query that populates it.

4.1 A General Algorithm for Roll-Up over Open Cubes

We now show an algorithm for the Roll-up operation, probably the most used one in the OLAP setting. We define the following functions: (a) *newVar(i)*

generates unique SPARQL variable names; (b) *value(v)* returns the value stored in variable *v*; (c) *levels(s)* returns all the levels in a schema *s* (i.e., all the values of *?l* that satisfy *s oc:hasLevel ?l*); (d) *measures(s)* returns all the measures in a schema *s* (all the values of *?m* that satisfy *s oc:hasMeasure ?m*); and (e) *aggFunction(m)* returns the aggregation function of measure *m* (all the values of *?f* that satisfy *m oc:hasAggFunction ?f*). Also assume that there is a level dl_o in dimension d such that there exists a path between dl_o and dl_r which contains only arcs with type *oc:parentLevel*. The function *levelsPath(d₁, d₂)* retrieves the ordered list of levels in the path between d_1 and d_2 (including both levels). Also assume that it is possible to access and modify different parts of a SPARQL query via the properties: *resultFormat*, *graphPattern*, and *groupBy*, among others. The *add(s)* function appends *s* to a particular part of the query.

<pre>eg:salesMonth rdf:type oc:FactSchema; oc:hasLevel eg:product; oc:hasLevel eg:city; oc:hasLevel eg:month; oc:hasMeasure eg:price; oc:hasMeasure eg:qtySold .</pre>
SalesByMonth schema
<pre>CONSTRUCT { ?id oc:hasSchema eg:salesMonth . ?id eg:product ?prod . ?id eg:city ?city . ?id eg:month ?mon . ?id eg:price ?priceMonth . ?id eg:qtySold ?qtyMonth . } WHERE { SELECT ?prod ?city ?mon (AVG(?price) AS ?priceMonth) (SUM(?qtySold) AS ?qtyMonth) (iri(fn:concat(" http://example.org/salesInstances#", "sales","-"), fn:substring-after(?prod," http://example.org/salesInstances#"),"-", fn:substring-after(?city," http://example.org/salesInstances#"),"-", fn:substring-after(?mon," http://example.org/salesInstances#")) AS ?id) WHERE { ?i oc:hasSchema sl:sales . ?i eg:product ?prod . ?i eg:city ?city . ?i eg:date ?date . ?i eg:price ?price . ?i eg:qtySold ?qty . ?date oc:parentLevelMember ?mon . ?mon oc:inLevel eg:month }GROUP BY ?prod ?city ?mon}}</pre>
SalesByMonth instances

Fig. 10. RollUp implementation over Open Cubes

<pre>eg:salesWithoutGeo rdf:type oc:FactSchema; oc:hasLevel eg:date; oc:hasLevel eg:city; oc:hasMeasure eg:price; oc:hasMeasure eg:qtySold .</pre>
SalesWithoutGeo schema
<pre>CONSTRUCT { ?id oc:hasSchema eg:salesWithoutGeo . ?id eg:city ?city . ?id eg:date ?date . ?id eg:price ?avgPrice . ?id eg:qtySold ?sumQtySold . } WHERE {{ SELECT ?city ?date (AVG(?price) AS ?avgPrice) (SUM(?qty) AS ?sumQtySold) (iri(fn:concat(" http://example.org/salesInstances#", "salesSGeo","-"), fn:substring-after(?city," http://example.org/salesInstances#"),"-", fn:substring-after(?date," http://example.org/salesInstances#")) AS ?id) WHERE { ?i oc:hasSchema eg:sales . ?i eg:city ?city . ?i eg:date ?date . ?i eg:price ?price . ?i eg:qtySold ?qty . }GROUP BY ?city ?date }}</pre>
SalesWithoutGeo instances

Fig. 11. Slice implementation over Open Cubes

Algorithm 1 shows a general procedure to build the SPARQL query needed to populate a cube $\mathcal{C}' = \text{Roll-up}(\mathcal{C}, D, dl_r, F)$, according to the definitions of Section 2. Two SPARQL queries are needed: an inner query $qInner$ that performs the GROUP BY, and an outer query $qOuter$ that builds triples based on the retrieved values from the inner query. The algorithm incrementally builds both queries simultaneously, using the *add* function. Line 1 states that generated facts have s_r as schema. Line 2 adds a triple to the WHERE clause of the inner query, that allows selecting facts from schema s_o . Lines 3 through 11 project the members of each level in the schema into the result of both queries, also adding triples to the WHERE clause of the inner query and adding the variables that represent the level members to the GROUP BY clause in the inner query. Lines 12 through 18 do the same for measures. In lines 13 and 16 f represents the SPARQL function corresponding to the aggregate function for each measure and $f(\text{value}(m_i))$ is the string that should be included to calculate the aggregated value (e.g. $\text{SUM}(?m)$ if $\text{value}(m_i) = ?m$). Lines 19 to 30 add the triples needed to navigate the dimension hierarchy. Line 22 adds to the inner query a triple that associates the level member with the fact instance, line 25 adds a triple that allows to check to which level the level member belongs to, and line 27 retrieves the parent level member of the current level. Line 30 adds the target level to the GROUP BY clause. Finally, line 33 sets the inner query as the WHERE clause of the outer query, which is returned in line 34. For clarity, we omitted the clause that generates the expression that binds variable `?id` to a dynamically generated URI from the values in the fact.

Preliminary Results. We have implemented the roll-up operation. We generated over 24000 triples from synthetic data that represent instances of 3000 facts and dimension members. Those triples were loaded in Virtuoso Opensource 6.1.4⁵. SPARQL 1.1 queries that implement the roll-up operation were performed over the triples, retrieving results in less than 0.1 seconds. Our assumption is that web cubes will not be large, since they will contain very focused and current data from selected sources. Then, the operators over web cubes will be useful to avoid going back and forth from the local multidimensional representation.

4.2 Exporting Web Cubes to a Local DSS

We now sketch how web cubes can be exported into the Mondrian OLAP server. Multidimensional schemas in Mondrian are specified via an XML file, which also contains the directives that allow populating the schemas.

The general mechanism has two phases: structure definition and population. The definition phase takes a web cube schema as input and produces two outputs: (i) the SQL code that creates tables in a relational database⁶, and (ii) an XML file that contains the schema definition of the cube, and the mappings that allow to populate the multidimensional schema with data stored in the relational database. The population phase takes as input a web cube instance, expressed

⁵ <http://www.openlinksw.com/wiki/main/Main>

⁶ Mondrian represents the cube in the relational model.

using Open Cubes, and produces SQL code that loads data into the database created in the definition phase. Figure 12 shows a portion of the XML file that represents the cube shown in Figure 2, closing the cycle of our running example: Jane requested the web cubes, which were retrieved, and represented in RDF using Open Cubes vocabulary; then she operated over the RDF representation of the web cube, and finally imported it to the local DSS, for joint analysis with the local cube.

Algorithm 1. Generates the SPARQL query that builds the Roll-Up instances

Input: s_o original schema, s_r new schema, dl_o level of $D \in s_o$, dl_r level of $D \in s_r$

Output: $qOuter$ is a SPARQL CONSTRUCT query that creates the roll-up instances

```

1: qOuter.graphPattern.add(?id , oc:hasSchema,  $s_r$ )
2: qInner.graphPattern.add(?i , oc:hasSchema,  $s_o$ )
3: for all  $l \in L = levels(s_o)$  do
4:   if  $l \neq dl_o$  then
5:     newVar( $l_i$ )
6:     qOuter.resultFormat.add(?id , l, value( $l_i$ ))
7:     qInner.resultFormat.add(value( $l_i$ ))
8:     qInner.graphPattern.add(?i , l, value( $l_i$ ))
9:     qInner.groupBy.add(value( $l_i$ ))
10:  end if
11: end for
12: for all  $m \in M = measures(s_o)$  do
13:    $f = aggFunction(m)$ 
14:   newVar( $m_i$ ); newVar( $ag_i$ )
15:   qOuter.resultFormat.add(?id , m, value( $ag_i$ ))
16:   qInner.resultFormat.add(f(value( $m_i$ )) AS  $ag_i$ )
17:   qInner.graphPattern.add(?i , m, value( $m_i$ ))
18: end for
19: for all  $dl_i \in path = levelsPath(dl_o, dl_r)$  do
20:   newVar( $lm_i$ )
21:   if  $dl_i = dl_o$  then
22:     qInner.graphPattern.add(?i , value( $dl_i$ ), value( $lm_i$ ))
23:   else
24:     newVar( $plm_i$ )
25:     qInner.graphPattern.add(value( $plm_i$ ), oc:inLevel, value( $dl_i$ ))
26:     if  $dl_i \neq dl_r$  then
27:       qInner.graphPattern.add(value( $lm_i$ ), oc:parentLevelMember, value( $plm_i$ ))
28:     end if
29:   end if
30: end for
31: newVar( $lm_i$ )
32: qInner.groupBy.add( $plm_i$ )
33: qInner.resultFormat.add( $plm_i$ )
34: qOuter.resultFormat.add(?id ,  $dl_r$ , value( $plm_i$ ))
35: qOuter.graphPattern.set(qInner)
36: return qOuter

```

```

<Schema>
<Cube name="WCubeSales">
<Table name="wcube_sales_fact" />
<Dimension name="Products" foreignKey="product_id">
<Hierarchy hasAll="false" primaryKey="product_id">
<Table name="products" />
<Level name="Model" column="model_id" uniqueMembers="true" />
<Level name="Manufacturer" column="manuf_id" uniqueMembers="true" />
<Level name="Category" column="category_id" uniqueMembers="true" />
</Hierarchy>
</Dimension>
<Dimension name="Time" foreignKey="date">
...
</Dimension>
<Measure name="Unit Price" column="unit_price" aggregator="avg" />
<Measure name="Delivery Time" column="del_time" aggregator="avg" />
<Measure name="Shipping Cost" column="ship_cost" aggregator="avg" />
</Cube>
</Schema>

```

Sales schema representation using Mondrian

Fig. 12. Exporting Web Cubes to a local DSS

5 Related Work

Our work is highly related with the idea of *situational BI*, a term coined in [11]. Situational BI focuses on executing complex queries, mainly natural language queries, over unstructured and (semi-) structured data; in particular, unstructured text retrieved from documents is seen as a primary source of information. In the context of situational BI the process of augmenting local data with data retrieved from web sources is discussed in [12].

In [4] a vocabulary called RDF Data Cube(DC) is presented. This vocabulary is focused on representing statistical data, according to the SDMX data model. Although this underlying data model shares some terms with traditional multidimensional data models, the semantics of some of the concepts are different. An example of this is the concept of *slices*. Slices, as defined in the DC vocabulary, represent subsets of observations, fixing values for one or more dimensions. Slices are not defined in terms of an existing cube, they are defined as new structures and new instances (observations). An example can be found in [4], Section 7. The semantics of the slice operator in the MD model is quite different, as shown in Section 2. Besides, while dimensions and its hierarchical nature are first class citizens in MD models, DC dimensions are flat concepts that allow to identify observations at a single granularity. The DC vocabulary does not provide the constructs to explicitly represent hierarchies within dimensions, neither at the schema level (DataStructureDefinition) nor at the instance level. As the DC vocabulary adheres to Linked Data principles hierarchies within dimensions may be inferred from external hierarchies, whenever possible. For example, members of a dimension stated to be of type `foaf:Person` can be grouped according to their place of work using the `foaf:workplaceHomepage` property. This is clearly not enough to guarantee the capability of the model to support OLAP operations as roll-up, which need to represent hierarchical relationships within dimensions levels and level members. Some of the problems found when trying to map cubes expressed in the DC vocabulary into a multidimensional model are discussed in [8]. In light of the above, we decided to build a new vocabulary from scratch, instead of extending DC.

To our knowledge no previous work addresses the problem of expressing OLAP operators over multidimensional data structures expressed in RDF. Nevertheless the idea of using RDFS or OWL ontologies to represent multidimensional data structures has been already explored. There is a line of work that uses these kinds of ontologies as auxiliary artefacts in traditional DSS. In [15] the authors propose an OWL ontology that models OLAP cubes schemas to assist in the ETL process of a traditional DSS, while in [14] a variation of the former ontology is used to check the consistency of summarizations. Among the approaches to the problem of populating multidimensional schemas with semantic web data, in [13] a process that extracts facts from semantic web data sources is outlined, assuming that the multidimensional schema and the sources are annotated using a single ontology. This work, however, does not deal with the extraction of dimension hierarchies.

6 Conclusion and Open Problems

We have presented an scenario where DSS systems are enhanced with web cubes obtained from current data on the web. We defined a vocabulary to represent these multidimensional data in RDFS, and operations over this representation, to avoid exporting these cubes to a classic OLAP system. We also presented a general algorithm for creating the SPARQL 1.1 queries that implement the Roll-up operation. To the best of our knowledge, this is the first approach of this kind in the field of BI over the Semantic Web. Existing approaches are based on the idea of obtaining RDF data and exporting these data to traditional OLAP cubes. Future work includes developing the complete framework, which encompasses requirements specification, data acquisition, web cubes extraction and publication, and data analysis using web cubes.

Acknowledgement. This work has been partially funded by the LACCIR Project “Monitoring Protected Areas using an OLAP-enabled Spatio-Temporal Geographic Information System”. Alejandro Vaisman has been partially funded by the “Open Semantic Cloud for Brussels (OSCB)” Project, supported by the Brussels Capital Region, Belgium.

References

1. Adida, B., Birbeck, M.: RDFa Primer, Bridging the Human and Data Webs (2008), <http://www.w3.org/TR/xhtml1-rdfa-primer/>
2. Beckett, D., Berners-Lee, T.: Turtle - Terse RDF Triple Language (2011), <http://www.w3.org/TeamSubmission/turtle/>
3. Brickley, D., Guha, R., McBride, B.: RDF Vocabulary Description Language 1.0: RDF Schema (2004), <http://www.w3.org/TR/rdf-schema/>
4. Cyganiak, R., Field, S., Gregory, A., Halb, W., Tennison, J.: Semantic Statistics: Bringing Together SDMX and SCOVO. In: Proc. of the WWW2010 Workshop on Linked Data on the Web, pp. 2–6. CEUR-WS.org (2010)
5. Harris, S., Seaborne, A.: SPARQL 1.1 Query Language (2010), <http://www.w3.org/TR/sparql11-query/>

6. Hurtado, C.A., Mendelzon, A.O., Vaisman, A.A.: Maintaining Data Cubes under Dimension Updates. In: Proc. of the 15th International Conference on Data Engineering, ICDE 1999, pp. 346–355. IEEE Computer Society, Washington, DC (1999)
7. Hurtado, C.A., Gutiérrez, C., Mendelzon, A.O.: Capturing summarizability with integrity constraints in OLAP. *ACM Transactions on Database Systems* 30(3), 854–886 (2005)
8. Kämpgen, B., Harth, A.: Transforming statistical linked data for use in OLAP systems. In: Proc. of the 7th International Conference on Semantic Systems, I-Semantics 2011, New York, NY, USA, pp. 33–40 (2011)
9. Kimball, R.: *The Data Warehouse Toolkit*. J. Wiley and Sons (1996)
10. Klyne, G., Carroll, J.J., McBride, B.: Resource Description Framework (RDF): Concepts and Abstract Syntax (2004), <http://www.w3.org/TR/rdf-concepts/>
11. Löser, A., Hueske, F., Markl, V.: Situational Business Intelligence. In: Castellanos, M., Dayal, U., Sellis, T. (eds.) *BIRTE 2008*. LNBIP, vol. 27, pp. 1–11. Springer, Heidelberg (2009)
12. Löser, A., Nagel, C., Pieper, S.: Augmenting Tables by Self-supervised Web Search. In: Löser, A. (ed.) *BIRTE 2010*. LNBIP, vol. 84, pp. 84–99. Springer, Heidelberg (2011)
13. Nebot, V., Llavori, R.B.: Building data warehouses with semantic data. In: *EDBT/ICDT Workshops*. ACM International Conference Proceeding Series. ACM (2010)
14. Niemi, T., Niinimäki, M.: Ontologies and summarizability in OLAP. In: Proc. of the 2010 ACM Symposium on Applied Computing, SAC 2010, pp. 1349–1353. ACM, New York (2010)
15. Niinimäki, M., Niemi, T.: An ETL Process for OLAP Using RDF/OWL Ontologies. In: Spaccapietra, S., Zimányi, E., Song, I.-Y. (eds.) *Journal on Data Semantics XIII*. LNCS, vol. 5530, pp. 97–119. Springer, Heidelberg (2009)
16. Prud’hommeaux, E., Seaborne, A.: *SPARQL Query Language for RDF* (2008), <http://www.w3.org/TR/rdf-sparql-query/>
17. Sequeda, J., Hartig, O.: Towards a query language for the web of data (a vision paper). *CoRR*, abs/1110.3017 (2011)
18. Shoshani, A.: OLAP and statistical databases: similarities and differences. In: *PODS 1997*, pp. 185–196. ACM, New York (1997)
19. Vassiliadis, P.: Modeling multidimensional databases, cubes and cube operations. In: *SSDBM*, pp. 53–62. IEEE Computer Society (1998)

Query-Independent Learning to Rank for RDF Entity Search

Lorand Dali¹, Blaž Fortuna¹, Thanh Tran², and Dunja Mladenić^{1,3}

¹ Jožef Stefan Institute, 1000 Ljubljana, Slovenia

{Lorand.Dali, Blaz.Fortuna, dunja.mladenic}@ijs.si

² Institute AIFB, Geb. 11.40 KIT-Campus Sd, D-76128 Karlsruhe, Germany
duc.tran@kit.edu

³ Jožef Stefan International Postgraduate School, 1000 Ljubljana, Slovenia

Abstract. The amount of structured data is growing rapidly. Given a *structured query* that asks for some entities, the number of matching candidate results is often very high. The problem of ranking these results has gained attention. Because results in this setting equally and perfectly match the query, existing ranking approaches often use features that are independent of the query. A popular one is based on the notion of centrality that is derived via PageRank. In this paper, we adopt *learning to rank* approach to this structured query setting, provide a systematic categorization of *query-independent features* that can be used for that, and finally, discuss how to leverage information in access logs to automatically derive the *training data* needed for learning. In experiments using real-world datasets and human evaluation based on crowd sourcing, we show the superior performance of our approach over two relevant baselines.

Keywords: information retrieval, learning to rank, semantic search.

1 Introduction

With the development of the Semantic Web as a Web of interlinked resource descriptions represented in RDF (e.g. Linked Data), and the continuous increase in the number of publicly available datasets, the problem of retrieval and ranking RDF resources has gained attention. Basically, a *RDF resource description* is a set of *triples*, which capture the *relations* of that resource to other resources, and its *attribute* values. On the Web today, we can find descriptions for different kind of entities, such as organizations, people, and geographic locations. Data that have been made publicly available through the Linking Open Data initiative for instance, include both encyclopedic knowledge captured by general datasets such as DBpedia and specific knowledge in various domains (music, life science, etc.).

For searching RDF resource descriptions (henceforth also called *entity search* because resources stand for real-world entities), the keyword paradigm commonly used for Information Retrieval (IR) has been adopted. Also, interfaces based on *structured query languages*, SPARQL in particular, are widely employed. Basically, SPARQL rests on the notion of graph pattern matching. It is widely used for

retrieving RDF data because RDF triples form a graph, and graph patterns matching subgraphs of this graph can be specified as SPARQL queries. Most endpoints, which provide public Web access to the kind of RDF data mentioned above, support SPARQL queries. While keyword search is clearly easier to use, structured query languages such as SPARQL can provide the expressiveness (technical) users may need in order to capture complex information needs, and to fully harness the structure and semantics captured by the underlying data. In fact, many queries posed on the Web are actually specified using form- and facet-based interfaces (e.g. faceted search provided by Yahoo!, Amazon and EBay). The inputs provided by the users through these interfaces are actually mapped to structured queries.

Since structured queries precisely capture the constraints the candidate answers must satisfy, the underlying engine can return perfectly sound and complete results. That is, all results can be found (*complete*) and every one of them perfectly matches the query (*sound*). However, given the large amount of data, queries may result in a large number of results, while only a few of them may be of interest to the user. In this case, ranking and returning only the top- k results is the standard strategy used in practical scenarios to improve efficiency and response time behavior. Studies have shown that users typically scan results beginning from the top, and usually focus only on the top three or four [1]. However, how do we *rank entity search results in this structured query scenario*, given all entities equally (i.e. perfectly) match the query?

A few specific approaches have been proposed to deal with ranking RDF results [5,7,10]. Most of these approaches [5] assume an ambiguous keyword query such that the ranking problem is mainly understood as the one of computing *content relevance*, i.e. to find out whether the resource's content is relevant with respect to the query. In the structured query setting, all resources are equally relevant. Ranking approaches [10,11] that can be used to distinguish resources in this setting are mainly based on *centrality*, a notion of "popularity" that is derived from the data via PageRank. Besides centrality, we study the use of other features and incorporate them into a *learning to rank* (LTR) framework for ranking entity search results, given structured queries. The main contributions of this paper can be summarized as follows:

- (1) **Learning to rank over RDF data.** LTR [2] is a state-of-the-art IR technique that learns a ranking function from labeled training data (relevance judgments). We show how LTR can be adopted for ranking entity search results over RDF data.
- (2) **Query-independent features.** Critical for the performance of LTR are features. For this specific structured query setting, we systematically identify query-independent features (those that go beyond content relevance) and individually analyze their impacts on ranking performance.
- (3) **Access logs based ground truth and training data.** While LTR offers high performance, it critically depends on the availability of relevance judgments for training. We observed from our experiments based on real users (via a crowd sourcing based evaluation recently proposed in [3]) that the final results strongly correlate with the *number of visits* (#visits) that is captured in the access logs. We provide a detailed analysis of this correlation and for the case where training data and ground truth is not easy to obtain, we propose the use of #visits as an alternative.

Using both cross-domain and domain-specific real world datasets, we evaluate the proposed LTR approach and show its superior performance over two relevant baselines. Results suggest that combining different features yields high and robust performance. Surprisingly, the use of features that are derived from the external Web corpus (features that are independent of the query and local dataset) yields the best performance in many cases.

Structure. The remainder of the paper is structured as follows. We firstly discuss related work in Section 2. Then, our adaptation of LTR is presented in Section 3. Experimental results are discussed in Section 4 before we conclude in Section 5.

2 Related Work

Approaches for ranking in the RDF setting can be distinguished into those which consider the *relevance* of a resource with respect to the query, and the others, which derive different features (e.g. *popularity*) from cues captured in the data such as centrality and frequency.

2.1 Query-Dependent Content Relevance

Approaches using query-dependent content relevance include the main IR approaches that basically, rank a result based on the likelihood its content is relevant, given the query. Different IR approaches have been adopted to rank structured results, and to deal with RDF in particular. An adoption of the *vector space model* has been proposed for ontology-based IR [4]. More specifically focused on the ranking of structured results in RDF is the work from Blanco et al. [5] that is built upon *BM25F*, another IR approach widely adopted in commercial search engines. The idea here is to use different fields for indexing different properties of RDF resources. Different weights are assigned to these fields to recognize that some fields are more important than others in ranking RDF resources. Also, the more recent *language modeling* (LM) paradigm has been adapted to the case of structured results. For ranking structured objects, three different models were studied [6]: The simple unstructured model treats all object attributes and values as vocabulary terms. The structured variant employs different term distributions for different attributes and assigns different weights to attributes (similar to the idea behind *BM25F*). Recently, LM is also used for ranking in the RDF setting [7]. For this, a language model is proposed for the query, and also RDF graphs matching the queries are represented as language models. As opposed to the approaches mentioned before, which model queries and documents based on words, the models employed here are probability distributions over RDF triples. The probability of a given triple should capture its “*informativeness*”, which is measured based on witness counts. The authors implement this by issuing keyword queries for each triple using Web search engines, and used the reported result sizes as witness count estimates.

Ranking as performed by these mentioned approaches is based on the relevance of the content with respect to the given keyword query. We focus on the ranking of

results to structured queries, which as opposed to the ambiguous keyword queries, are precisely defined such that the query semantics can be fully harnessed to produce answers that are equally relevant. Thus in principle, content relevance can be expected to be less important in this case, and other features should be considered for ranking. Among the approaches mentioned above, the only exception that deals with structured queries is the LM-based ranking of RDF triples (graphs). As discussed, this work does not directly capture content relevance but relies on informativeness. We consider this as one baseline and show that using additional features can substantially outperform this. Previous works build upon the vector space model [4], language models [7], and probabilistic IR [5]. In this work, we adopt yet another popular IR paradigm, namely LTR [2]. This paradigm constitutes the state-of-the-art in IR, and is widely used by commercial Web search engines.

2.2 Query-Independent Features

Approaches described in this subsection are not taking the query into account, but rather using query-independent features. An example of such features that are independent of the query is *centrality*, which can be derived from the graph-structured nature of the underlying data using algorithms such as PageRank [8] and HITS [9]. The aim of PageRank is to give a global, query-independent score to each page. The score computed by PageRank for a given page captures the likelihood of a random Web surfer to land on that page.

The first adoption of PageRank in the structured data setting was proposed for Entity-Relation graphs representing databases, and specific approaches for dealing with RDF graphs have been introduced recently. For instance, ResourceRank [10] is such a PageRank adapted metric that is iteratively computed for each resource in the RDF dataset. Also, a two layered version of PageRank has been proposed [11], where a resource gets a high rank if it has a high PageRank within its own graph, and if this graph has a high PageRank in the LOD cloud (which is also considered as a graph where nodes represent datasets). The difficulties in adapting PageRank to the structured data setting is that the graph here – as opposed to the Web graph – has heterogeneous nodes and edges (different types of resources and different relations and attribute edges). A solution is to manually assign weights to different relations, but this approach is only applicable in a restricted domain such as paper-author-conference collections [13].

Instead of centrality, more simple features based on *frequency* counts have also been used in the RDF setting. For instance, structured queries (graph patterns) representing interpretations of keyword queries have been ranked based on the frequency counts of nodes and edges [16]. Just like PageRank scores, these counts aim to reflect the popularity of the nodes and edges in the query pattern such that more popular queries are preferred. The use of frequency has long tradition in IR. Term and inverse document frequencies are commonly used to measure the importance of a term for a document relative to other terms in the collection.

These query-independent features can be directly applied to our structured query setting to distinguish between the results that are equally relevant. In a systematic

fashion, we identify different categories of features that can be used for our LTR approach, including centrality and frequency. We show that besides the features derived from the corpus (i.e. the underlying RDF graph), external information on the Web provides useful features too. We compare and show that the use of different features can outperform the ResourceRank baseline, which is based on centrality.

3 Query Independent Learning to Rank over RDF

We describe how we adapt LTR to the RDF entity search setting, followed by a detailed description of the features which the learning algorithm uses.

3.1 Learning to Rank

LTR [2] is a machine learning technique used to induce a ranking model from training data. We use the pairwise setting, which means that a training example is provided as a pair of entities and we know which of the two entities should be ranked higher in the result set. In what follows we formally describe the pairwise method and how it is adapted to our case.

For a given dataset D , let $Q = \{q_1, q_2, \dots, q_n\}$ be the set of queries. For every query q_i let $R(q_i) = \{r_1^i, r_2^i, \dots, r_k^i\}$ be the set of answers to q_i . We define a feature set as $F := (f_1, f_2, \dots, f_p)$ where f_j are the functions $f_j : R(q_i) \rightarrow [0, 1]$, which assign a real value to each answer, and we normalize the feature values to values between 0 and 1 for each query separately. We refer to $f_j(r)$ as a feature of the resource r . A target feature f_t (also called *label*) is a special feature, which determines the correct ranking as a descending ordering of the resources. It is the ranking based on the target feature f_t , which we want to obtain using a LTR algorithm.

For every answer r_j^i we compute a feature vector $x_j^i := (f_1(r_j^i), f_2(r_j^i), \dots, f_p(r_j^i))$. The feature vector x_j^i does not contain any target feature f_t . The training set consists of all pairs

$$(x_R^i, x_N^i), \quad \forall i \in \overline{1, n}, \quad \forall r_R^i, r_N^i \in R(q_i), \quad R \neq N$$

such that

$$f_t(r_R^i) > f_t(r_N^i).$$

In other words, for each query, we take all the pairs of the feature vectors of the answers to the query such that we put the answer with a higher target feature on the first place. To each pair (x_R^i, x_N^i) we associate a cost C :

$$C := \frac{2f_t(r_R^i)}{f_t(r_R^i) + f_t(r_N^i)} - 1.$$

Intuitively we can think of C as the confidence in the correct ordering of the pair (r_R^i, r_N^i) or as the penalty, which the learning algorithm receives if it makes a mistake on this pair. We can observe that if $f_t(r_R^i) = f_t(r_N^i)$ then $C = 0$, so we are not

confident at all that r_R^i should be ranked higher than r_N^i . On the other hand if $f_t(r_R^i) \gg f_t(r_N^i)$ then the value of C gets close to 1, and the learning algorithm obtains a big penalty for making a mistake on this pair.

The list of pairs (x_R^i, x_N^i) with their associated cost C is the input to the RankSVM [17] algorithm described below. The goal is to learn a weight vector $w \in \mathbb{R}^p$ of the same dimensions as the training vectors x . Then given a new vector y , representing the feature vector of an answer to be ranked, we can compute the score of the answer, which is equal to the inner product between the weight vector w and the vector y ,

$$score = w^T y .$$

The ranking is then obtained by sorting answers by their scores.

3.2 Rank SVM

Linear SVM [21] is a popular way of learning the weight vector w . Originally SVM is formulated as a binary classification problem, where w is the separating hyperplane with maximum margin. The linear soft margin SVM for classification can be adapted to the pairwise ranking problem. The objective is to make the inner product $w \cdot x_R^i$ greater than $w \cdot x_N^i$ by the margin 1 and allowing for some errors ξ . We have

$$w \cdot (x_R^i - x_N^i) \geq 1 - \xi_i, \forall i .$$

The maximum margin separating hyperplane is the one which minimizes

$$\frac{1}{2} \|w\|^2 + C \sum \xi_i .$$

This is called the primal problem, and it is the one which we shall solve as described in [22]. By substituting ξ_i we get the hinge loss

$$\frac{1}{2} \|w\|^2 + C \sum (1 - w \cdot (x_R^i - x_N^i))_+ ,$$

where the function $(\cdot)_+$ is defined as $(\cdot)_+ := \max(0, \cdot)$. We minimize the hinge loss by using the subgradient method, which gives a fast but approximate solution.

3.3 Feature Extraction

The proposed approach uses features which can be grouped into *dataset specific* or *dataset independent* features. Dataset specific features are extracted from the RDF graph. Dataset independent features are extracted from external sources like web search engines or N-gram databases. Note that although the dataset specific features are specific to the dataset, the methodology to extract these features can be applied to any RDF dataset.

We also classify the features into *frequency*-based features obtained by counting different patterns in RDF graphs or counting the number of occurrences in web search

results or n-gram databases, and *centrality*-based features obtained by applying graph theoretic algorithms like PageRank or HITS on the RDF graph.

In the following subsections we shall describe each feature in detail.

3.3.1 Features Extracted From the RDF Graph

In this section we describe several features extracted from the RDF graph. We look at the RDF dataset as a directed graph with resources as nodes and properties as edges. We define the concept of a *feature at level K* as follows. We call anchor node the node, which corresponds to the resource we want to extract the feature for. In Figure 1 and Figure 2 the anchor nodes are shaded. The feature at level 1 is the feature extracted from the anchor node. The feature at level 2 is computed from the nodes one step away from the anchor node. One step away means that we go to the neighbor either in the direction of the edge or in the opposite direction. In general the feature at level K is computed from the nodes which are K-1 steps away from the anchor node. In the experiments presented in this paper we have used the features at levels 1 and 2. In what follows we provide the list of features we extract from the RDF graph

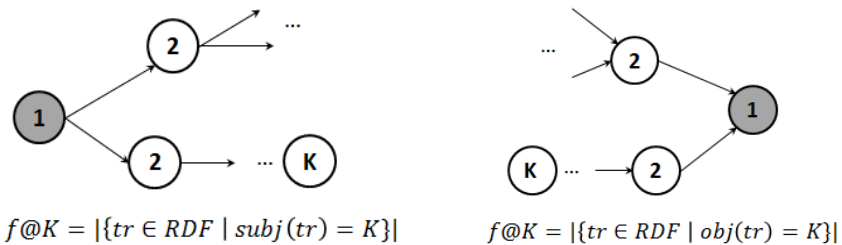


Fig. 1. Number of subjects (left) and objects (right) at level K

Number of subjects @ K. This feature is a count of the triples, which have as subject the node for which we extract the feature. In Figure 1 on the left, the value of this feature at level 1 is 2 (because two arrows go out) and the value of this feature at level 2 is 3 (= 2 + 1).

Number of objects @ K. This feature is computed in a similar way to the number of subjects @ K, the difference being that now the number of triples with the node as object is counted (arrows coming in). The graph on the right side of Figure 1 illustrates the computation of this feature.

Number of types of outgoing predicates @ K. At each level this feature is the count of the elements of the set of predicates occurring at that level. The anchor node is the subject. This feature is illustrated on the left side of Figure 2.

Number of types of incoming predicates @ K. At each level this feature is the count of the elements of the set of predicates occurring at that level. The anchor node is the object. This feature is illustrated on the right side of Figure 2.

Average frequency of outgoing predicate @ K. We compute the frequency counts of all predicates in the dataset. At level K we take the set of predicates P_K and we compute the feature as the average of the frequency counts of the predicates in P_K .

Average frequency of incoming predicate @ K. This feature is similar to the previous feature, the difference being that we take into account the predicates which correspond to edges pointing towards the anchor node.

Number of literals. This feature counts how many times the anchor node occurs as the subject in an RDF triple where the object is a literal.

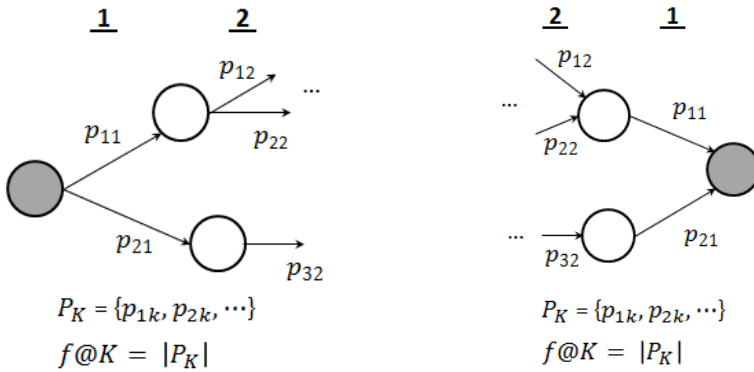


Fig. 2. Number of types of outgoing and incoming predicates at K

3.3.2 PageRank

This section briefly describes the PageRank algorithm and how it applies to our case. PageRank was introduced in the early days of web search out of a need for a global, query independent ranking of the web pages. PageRank assumes a directed graph as input and will give a score to each of the nodes as a result. PageRank is based on the random walk model, which assumes that a very large number of users walk the graph choosing at each step a random neighbor of the current node or jumping to any node in the graph. The score of a node is given by the expected number of users being at the given node at a moment in time. The scores are computed recursively from the following equation:

$$p = d \cdot M \cdot p + (1 - d) \cdot u, \quad p, u \in \mathbb{R}^n, M \in \mathcal{M}(n)$$

Where n is the number of nodes in the graph, p is the PageRank vector containing the score for each node and is initialized with 0, M is the transition matrix constructed such that $M[i, j] = 1$ if there is an edge from node i to node j and 0 otherwise. Moreover, to eliminate nodes which do not link to any other node we consider a sink node k such that $M[i, k] = 1, \forall i$ and $M[k, i] = 0, \forall i$. Finally the columns of M are normalized to sum up to 1; u is the jump vector and its entries are $u[i] = \frac{1}{n}, \forall i$; d is

the damping parameter, and represents the probability of walking to a neighboring node versus jumping. In our experiments we have set the value of d to its typical value of 0.85, and ran the iteration until it converged.

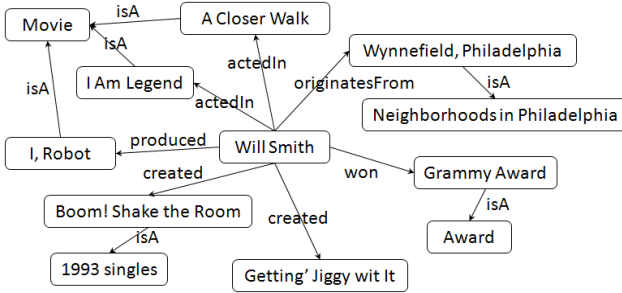


Fig. 3. An example graph representation of a part of the Yago knowledge base

In case of the web, the graph is made of the web pages as nodes and the hyperlinks as edges. In our case the nodes are DBpedia or Yago resources or categories, and the edges are properties. For illustration, Figure 3 shows a subgraph from the Yago knowledge base.

3.3.3 Hubs and Authorities

Hubs and authorities, also known as Hyperlink-Induced Topic Search (HITS) algorithm, is an iterative algorithm which takes as input a directed graph and assigns two scores to each of its nodes. The two scores, the hub score and the authority score, are defined recursively in terms of each other such that a node gets a high hub score if it points to nodes with high authority scores, and a node gets a high authority score if it is pointed to by nodes with high hub scores.

$$\forall p, \quad auth(p) = \sum_{i=1}^n hub(i)$$

$$\forall p, \quad hub(p) = \sum_{i=1}^n auth(i)$$

where p is a node in the graph, n is the total number of nodes connected to p , and i is a node connected to p . $auth(p)$ and $hub(p)$ are initialized to 1.

3.3.4 Search Engine Based

We have used the search services provided by Yahoo! BOSS¹ to measure how many times the label of a resource (which corresponds to an answer to a query) appears on the internet. We do this by searching the web with the resource's label as a query and taking the number of hits as a feature. For instance to compute this feature for the

¹ <http://developer.yahoo.com/search/boss/>

resource corresponding to the person Neil Armstrong, we make a web search with the query ‘Neil Armstrong’ and obtain that the number of search results is 3720000.

3.3.5 Google N-grams

Google released a dataset of all n-grams² (1-grams up to 5-grams) which appear on the internet at least 40 times, together with their frequency counts. We consider as a feature of a resource the frequency count of its label in the n-gram dataset. When the label of a resource is composed of many words we generate all 3-grams from the label and take the sum of the frequencies of the 3-grams as a feature. For instance, to compute this feature for the resource corresponding to the person Neil Armstrong we search for the 2-gram ‘Neil Armstrong’ and obtain that it occurs 132371 times in the Google N-gram database.

4 Experiments

Given RDF datasets and SPARQL queries, we obtained results using a Triple store. In the experiments, we run different versions of the proposed LRT algorithm and baselines to compute different rankings of these results. The goals of the experiments are (1) to compare LTR against the baselines and (2) to analyze the performance of individual features (feature sets). As performance measures, we use the standard measures NDCG and Spearman’s correlation coefficient. We build upon the data, queries and methodology proposed by the recent SemSearch Challenge evaluation initiative [3]

4.1 Datasets and Queries

We have two sets of queries³. The first set is a subset of the entity queries provided by the SemSearch Challenge dataset. It consists of 25 queries, for which we obtain answers from DBpedia and Yago, two datasets containing encyclopedic knowledge that were extracted from Wikipedia infoboxes. Answers from these datasets correspond to Wikipedia articles. We used the Wikipedia access logs from June 2010 to January 2011 (available at <http://dammit.lt/wikistats/>).

The other set consists of 24 queries, whose answers are computed from the Semantic Web Dog Food (SWDF) dataset [18]. SWDF contains information about people, conferences, workshops, papers and organizations from the Semantic Web field. The dataset is built from metadata about conferences such as ISWC and ESWC, starting from the year 2006. For the USEWOD 2011 Data Challenge [19], a dataset⁴ of access logs on the SWDF corpus was released.

² <http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html>

³ http://aidemo.ijs.si/paper_supplement/dali_eswc2012_queries.zip

⁴ <http://data.semanticweb.org/usewod/2011/challenge.html>

While the first set of queries is used to evaluate ranking in a general setting, the second one is used to analyze how the approaches perform in a domain-specific setting.

4.2 Ground Truth: Human Judgments vs. Access Logs Information

We follow the crowd sourcing approach of SemSearch [3] to obtain relevance judgments from human users. We give human evaluators the following task: Given a question and the answers computed by the system, they should vote for the one answer, which should be ranked first. Moreover, they should indicate the confidence in their choice. We used the number of votes (#votes) for an answer as the ranking criterion. For each question twenty evaluators have voted.

We also propose an automatic way to obtain the ground truth by using access logs. We take as the ranking score of a resource the number of times that resource has been visited, where #visits is obtained from access logs.

We now discuss the correlation between the ranking resulting from human judgments and the one based on #visits. Figure 4 shows one example question, namely “List of boroughs in New York City”. The upper bar (red) shows the percentage of visits, and the lower one (blue) shows the percentage of votes this answer has received. It can be seen that the ranking based on #visits is almost the same as the ranking based #votes, especially for the higher ranked answers. To quantify this correlation, we used #votes as the ground truth and computed NDCG for the ranking based on #visits. We obtained $NDCG = 0.993$ for this question, and the average confidence was 0.675, where 1 is the maximum confidence, and 0 is the lowest.

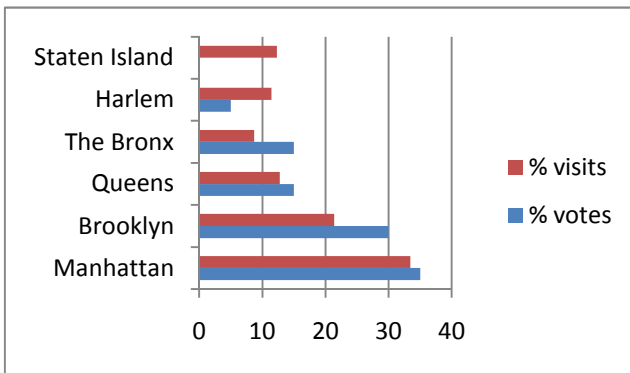


Fig. 4. Percentage of votes and visits for the query “List of boroughs of New York City”, $NDCG = 0.993$, average confidence = 0.675

There are also a few questions where the rankings based on #votes and #visits are not so similar. For instance for the question “Books of Jewish Canon” the NDCG score is only 0.57. However, the average user confidence is also lower in this case (only 0.476). Other questions of this type are “Names of hijackers in the September 11 attacks”, “Ratt albums” and “Ancient Greek city-kingdoms of Cyprus”. All these

questions are relatively specific. We observed in these cases, users indicated relative low confidence, and the agreement between users is also low, suggesting that it was difficult for them to choose the correct answers.

Figure 5 shows the correlations between NDCG scores computed for the ranking based on #visits, confidence and agreement values for each question. By agreement between users we mean the percentage of votes the answer with the highest number of votes has obtained. We can see that in general the ranking based on #votes is quite similar to the ranking based on #visits. More exactly, the average NDCG score is 0.86. For 15 of the 25 queries, the answer with most votes corresponds to the article that is most visited on Wikipedia. Further, we see that the higher the confidence of the users, the higher is also the NDCG based on #visits. Also, NDCG based on #visits correlates with agreement. This means that when users are confident and agree on the results, the ranking based on #visits closely matches the ranking based on #votes.

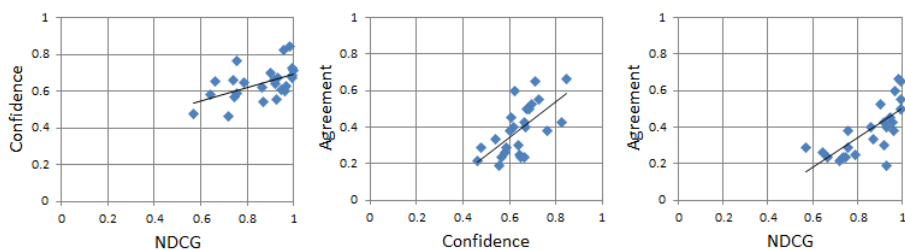


Fig. 5. Correlations between confidence, agreement and NDCG values

In conclusion, we have constructed a golden standard for ranking by asking users to vote. The results suggest that for questions where human users can provide correct answers, this golden standard correlates well with the ranking based on #visits. It does not closely match the standard in a few cases, which represent difficult questions for which the user judgments were also not reliable. Thus, #visits can be seen as a good approximation of the ground truth. This has important implications because then, #visits not only can be used as ground truth to evaluate ranking performance, but also as a target feature for training the LTR model.

4.3 Systems

As baselines for evaluating the proposed ranking models, we have implemented two ranking methods described in the related work. The first is ResourceRank (*ResRank*) [10], which provides a global, query-independent PageRank inspired ranking score. The second baseline (*LM*) is based on building language models for the query and results [20]. As discussed, it actually relies on a rather query-independent metric called witness count, which measures the “informativeness” of RDF triples. This count is estimated based on the number of results obtained from searching the Web with the labels of the subject, predicate and object of the triple as queries. Because the number of triples in DBpedia and Yago is large, it was not feasible for us to submit

the resulting search requests. For this baseline, we could obtain results only for the smaller SWDF dataset. The last one called *Wikilog* is considered as an upper limit baseline, which rank results based on #visits in the access logs.

We have implemented several LTR systems based on different features and labels (target features). In particular, we used the four different categories discussed before, namely (1) features based on graph *centrality*, (2) features based on *external* sources, (3) features based on the *RDF* dataset, and (4) the *complete* set of all features. Two target features were used, namely #votes (systems using these labels for training are denoted by the prefix ‘**H_**’) and #visits (systems denoted by prefix ‘**L_**’).

4.4 Evaluation of Learning to Rank

The rankings produced by our LTR models are evaluated based on leave-one-out cross-validation. This means the model is trained on the data from all queries except one; then the model is tested to rank the answers of the left-out query. The procedure is repeated for each query. For queries on DBpedia and Yago, the metrics are computed using the ground truth obtained from human evaluators. For queries on SWDF, the ground truth is obtained from access logs. This is because SWDF is too specific such results cannot be reliably evaluated by people who are not domain experts. The results are computed for each query separately, and the average values are summarized in Table 1. (detailed experimental results are also available⁵)

WikiLog gives best performance which cannot be surpassed even when training on data with human judgments as target feature (rows with prefix ‘**H_**’). Moreover, we notice that the performance of models trained using ground truth obtained from logs (‘**L_**’) is actually higher than the performance of models trained using ground truth obtained from humans. The main reason for this is that many answers get the same number of votes. This holds especially for the answers, which get few votes or no votes at all. Therefore, many pairs of answers in the training data have no or very low confidence, resulting in much fewer valuable training examples (see Section 3.1). A possible solution is to ask human evaluators to do complete orderings instead of votes. Clearly, this results in a complex task that may be not practical for crowd sourcing. In other words, obtaining training data from humans is difficult.

Comparing to the baselines we noticed that the proposed models are comparable in the general domain (DBpedia and Yago). However, for the domain-specific dataset of SWDF, the improvements are more significant. Further, for all systems, the performance in the specific domain case is lower than in the general domain.

External and Complete seem to be better than other for DBpedia and Yago. For SWDF however, External performs badly, and Centrality and RDF are much better. We think the weak performance of External in the specific domain is because specific resources rarely appear in n-grams and search results. Centrality and RDF, being specific to the dataset, perform much better. Complete have stable and good performance in both settings, mainly because it contains more features which compensate for each other. Notably, for SWDF, Complete, which contains the weakly performing external feature set, still comes out as the best.

⁵ http://aidemo.ijs.si/paper_supplement/dali_eswc2012_eval.zip

Looking at individual features we found that features like the number of search results, the number of objects, number of objects @ 2, the number of different incoming predicates @ 2 and the ngram count are among the best features for both DBpedia and Yago achieving NDCG scores of about 0.8.

Table 1. Experimental results

	DBpedia		Yago		SWDF	
	NDCG	Spearman	NDCG	Spearman	NDCG	Spearman
WikiLog	0.8602	0.5000	0.8602	0.5000	-	-
ResRank	0.8329	0.2552	0.8206	0.3276	0.6803	0.2287
LM	-	-	-	-	0.7191	0.2548
H_Centrality	0.7837	0.1524	0.8035	0.2751	-	-
H_External	0.8339	0.3544	0.8339	0.3544	-	-
H_RDF	0.8339	0.3078	0.7832	0.1627	-	-
H_Complete	0.8322	0.2755	0.7999	0.2955	-	-
L_Centrality	0.8294	0.2593	0.8118	0.3055	0.7376	0.2868
L_External	0.8380	0.3383	0.8380	0.3383	0.6149	0.1201
L_RDF	0.8076	0.2353	0.8228	0.2239	0.7401	0.3019
L_Complete	0.8374	0.2861	0.8435	0.3510	0.7533	0.3160

5 Conclusions and Future Work

We have presented a LTR approach for ranking RDF entity search results that considers a multitude of query-independent features. These features are particularly important in this setting where all results are equally relevant with respect to the query. We show that LTR can outperform the baselines, and the improvement is particularly large for the domain specific dataset. We have analyzed the impact of individual features on the LTR performance. The complete combination of features yields high and consistent performance. Surprisingly, good results could also be obtained when only external features derived from the Web are used.

As future work, we will investigate the use of LTR for ranking RDF results in the keyword query setting, which will require both query-independent and query-specific features.

Acknowledgements. This work was supported by the Slovenian Research Agency, the ICT Programme of the EC under PASCAL2 (IST-NoE-216886), RENDER (ICT-257790-STREP) and PlanetData (ICT-NoE-257641).

References

1. Cutrell, E., Guan, Z.: What are you looking for?: an eye-tracking study of information usage in web search. In: CHI 2007, pp. 407–416 (2007)
2. Liu, T.-Y.: Learning to Rank for Information Retrieval. Foundations and Trends in Information Retrieval 3(3), 225–331 (2009)

3. Blanco, R., Halpin, H., Herzig, D.M., Mika, P., Pound, J., Thompson, H., Tran, D.T.: Repeatable and Reliable Search System Evaluation using Crowd-Sourcing. In: SIGIR 2011, pp. 923–932 (2011)
4. Castells, P., Fernández, M., Vallet, D.: An Adaptation of the Vector-Space Model for Ontology-Based Information Retrieval. *IEEE Trans. Knowl. Data Eng.*, 261–272 (2007)
5. Blanco, R., Mika, P., Vigna, S.: Effective and Efficient Entity Search in RDF 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. 83–97. Springer, Heidelberg (2011)
6. Nie, Z., Ma, Y., Shi, S., Wen, J.-R., Ma, W.-Y.: Web Object Retrieval. In: WWW 2007, pp. 81–90 (2007)
7. Kasneci, G., Elbassuoni, S., Weikum, G.: MING: mining informative entity relationship subgraphs. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, pp. 1653–1656 (2009)
8. Page, L., Brin, S., Motowani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Libraries (1998)
9. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. *J. ACM* 46(3), 604–632 (1999)
10. Hogan, A., Harth, A., Decker, S.: ReConRank: A Scalable Ranking Method for Semantic Web Data with Context. In: SSWS 2006 (2006)
11. Delbru, R., Toupikov, N., Catasta, M., Tummarello, G., Decker, S.: Hierarchical Link Analysis for Ranking Web Data. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010. LNCS, vol. 6089, pp. 225–239. Springer, Heidelberg (2010)
12. Jeh, G., Widom, J.: Scaling personalized web search. In: WWW 2003, pp. 271–279 (2003)
13. Hristidis, V., Hwang, H., Papakonstantinou, Y.: Authority-Based Keyword Search in Databases. *ACM Transactions on Database Systems* 33(1) (2008)
14. Chakrabarti, S.: Dynamic Personalized Pagerank in Entity-Relation Graphs. In: WWW 2007, pp. 571–580 (2007)
15. Nie, Z., Zhang, Y., Wen, J.-R., Ma, W.-Y.: Object-Level ranking: Bringing Order to Web Objects. In: WWW 2005, pp. 567–574 (2005)
16. Thanh, T., Wang, H., Rudolph, S., Cimiano, P.: Top-k Exploration of Query Candidates for Efficient Keyword Search on Graph-Shaped (RDF) Data. In: ICDE 2009, pp. 405–416 (2009)
17. Joachims, T.: Optimizing search engines using clickthrough data. In: KDD 2002, pp. 133–142 (2002)
18. Möller, K., Heath, T., Handschuh, S., Domingue, J.: Recipes for Semantic Web Dog Food — The ESWC and ISWC Metadata Projects. 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. 802–815. Springer, Heidelberg (2007)
19. Berendt, B., Hollink, L., Hollink, V., Luczak-Rösch, M., Möller, K., Vallet, D.: USEWOD 2011. In: WWW (Companion Volume) 2011, pp. 305–306 (2011)
20. Elbassuoni, S., Ramanath, M., Schenkel, R., Sydow, M., Weikum, G.: Language-Model-Based Ranking for Queries on RDF-Graphs. In: CIKM 2009, pp. 977–986 (2009)
21. Joachims, T.: Making Large-Scale SVM Learning Practical. In: Scholkopf, B., Burges, C., Smola, A. (eds.) *Advances in Kernel-Methods - Support Vector Learning*. MIT Press (1999)
22. Rupnik, J.: Stochastic subgradient approach for solving linear support vector machines. In: *SIKDD* (2008)

COV4SWS.KOM: Information Quality-Aware Matchmaking for Semantic Services

Stefan Schulte¹, Ulrich Lampe², Matthias Klusch³, and Ralf Steinmetz²

¹ Distributed Systems Group, Vienna University of Technology, Austria
`s.schulte@infosys.tuwien.ac.at`

² Multimedia Communications Lab, Technische Universität Darmstadt, Germany

³ German Research Center for Artificial Intelligence, Saarbrücken, Germany

Abstract. The discovery of functionally matching services – often referred to as matchmaking – is one of the essential requirements for realizing the vision of the Internet of Services. In practice, however, the process is complicated by the varying quality of syntactic and semantic descriptions of service components. In this work, we propose COV4SWS.KOM, a semantic matchmaker that addresses this challenge through the automatic adaptation to the description quality on different levels of the service structure. Our approach performs very good with respect to common Information Retrieval metrics, achieving top placements in the renowned Semantic Service Selection Contest, and thus marks an important contribution to the discovery of services in a realistic application context.

1 Introduction

From the very beginning of semantic Web service (SWS) research, service discovery and matchmaking have attracted large interest in the research community [9,13,18]. The underlying techniques to measure the similarity between a service request and service offers have been continuously improved, but matchmakers still rely on a particular information quality regarding the syntactic and semantic information given in a service description. There are several reasons why the quality of syntactic and semantic service descriptions differs between service domains. While in one domain, a well-accepted ontology describing the particular (industrial) domain could be available, such an ontology might be missing for other domains. Furthermore, it could be the case that the usage of a certain domain ontology in a specific industry is compulsory due to legal constraints, as it is the case in the energy domain. In an upcoming *Internet of Services*, it is even possible that there will be premium service marketplaces for certain domains, which will only publish a service advertisement if certain quality standards regarding the service description are met. All things considered, the quality of service descriptions will differ from service domain to service domain.

In this paper, we present our work on information quality-aware service matchmaking. We propose an adaptation mechanism for matchmaking, which is based on the usability and impact (with regard to service discovery) of syntactic descriptions and semantic annotations on different levels of the service description

structure. The actual measurement of semantic-based similarities is based on metrics from the field of information theory. Furthermore, a linguistic-based fallback strategy is applied. Our approach is implemented in COV4SWS.KOM, a service matchmaker for WSDL 2.0 and SAWSDL. Unless explicitly defined otherwise, it is possible to transfer all concepts and results from this paper to WSDL 1.1 based on corresponding mappings respectively adaptations of the algorithms. In fact, COV4SWS.KOM also operates on other SWS description standards, most importantly hRESTS in conjunction with MicroWSMO for RESTful services. However, due the existence of a de facto standard test collection for the purposes of evaluation, our focus lies on WSDL in combination with SAWSDL in the work at hand. COV4SWS.KOM extends our former work on semantic matchmaking [14,22] by providing a novel approach to adapt matching results based on the usability of service descriptions in a certain domain.

The remainder of this paper is structured as follows: In Section 2, we provide a brief presentation of SAWSDL. Our general considerations as well as the matchmaking approach are presented in Section 3. We evaluate different configurations of COV4SWS.KOM and compare the results with other matchmaking approaches for SAWSDL (Section 4). As will be presented in the evaluation, COV4SWS.KOM is capable of competing with state-of-the-art matchmakers in terms of Information Retrieval (IR) metrics like precision and recall while offering an adaptation mechanism for different degrees of description quality. Eventually, we comment on the related work (Section 5) and conclude this paper (Section 6).

2 Service Descriptions Using WSDL 2.0 and SAWSDL

To keep this paper self-contained, we will give a short discussion of WSDL 2.0 in the following. We refer to the WSDL 2.0 specification for further details [3].

The abstract part of a WSDL document advertises *what* a service does while the concrete part defines *how* a service can be consumed and *where* it is located. In service discovery, the description of what a service does is of primary interest. Hence, in the following, the abstract part of a WSDL-based service description – interfaces, operations, and message parameters – will be utilized. These service components constitute the *service abstraction levels* of WSDL 2.0: Functionalities, i.e., interactions between a client and a service, are described by abstract operations. A set of operations defines a service interface. For each operation, a sequence of messages a service is able to send or receive may be defined. In WSDL 2.0, messages are defined using (XSD) parameter types [3].

SAWSDL imposes neither restrictions on what a semantic annotation eventually means nor the type of semantic concept that is addressed. However, the SAWSDL specification states that on interface level, a *modelReference* might be a categorization, while on operation level, a *modelReference* might specify a high level description of the operation – both apply to functional semantics as defined by Gomadam et al. [7]. On message respectively parameter level, *modelReferences* most likely define data semantics [6]. This meaning of semantic annotations is compliant with the classification made for *WSMO-Lite* [23] and will be the foundation for the matchmaking approach presented in this paper.

Likewise, the SAWSDL specification does not restrict the type of semantic concepts a `modelReference` should point to. The only requirement is that the concepts are identifiable via URI references. This is an advantage in so far as it allows for a maximum of flexibility in annotating functional service descriptions. Yet, this fact poses a problem if the concepts need to be automatically processed and interpreted in some form. In the context of our work, we will assume the semantic concepts to be formally defined in an OWL DL ontology. As a second constraint, we only consider the first URI from a `modelReference`, all other URIs are not regarded. This constraint is primarily made for practical reasons, as there is no agreement what another `modelReference` actually addresses: It could be a reference to a semantic concept from another domain ontology or address preconditions and effects, as it is done for operations in WSMO-Lite [23].

3 Information Quality-Based Matchmaking

As our matchmaking approach aims at the information quality-aware adaptation of the service discovery process, there are a number of challenges to be met: First, it is necessary to provide the means to compute both syntax- and semantic-based similarity values for different service components, i.e., interfaces, operations, inputs, and outputs. Second, the according similarity values need to be easily combinable in order to derive an overall value for a service request and the service offers that come into consideration. We will investigate these aspects in detail in Section 3.2. Third, the information quality needs to be assessed. Therefore, it is necessary to define what the meaning of information quality actually is:

Information Quality of Service Descriptions. According to [24], data or information quality is a subjective value that needs to be assessed with regard to the task the information will be applied to. Hence, in service matchmaking, information quality depends on the positive impact a service description will have on the outcome of the actual matchmaking process: If a certain information will have a relatively large positive impact on the identification of relevant services, its quality is assumed to be relatively high. Therefore, it is necessary to measure this impact and consequently adapt a matchmaker in order to give information with a higher quality a larger influence on the overall matchmaking results. With respect to service descriptions, information quality could also be interpreted as the degree to which the description is correct regarding the service it purports to describe. However, this would mean to interpret information quality with regard to service description *correctness*, not service discovery.

Information Quality for Service Selection. As mentioned above, it is reasonable to assume different qualities of information in different service domains. While in one domain, there might be no standard for semantic information at all, there are other domains which provide some semantics (e.g., compulsory data semantics as in the already mentioned energy domain) and maybe – in the upcoming Internet of Services – even domains which demand to semantically describe the complete structure of a service description. Hence, it is reasonable

to assume different qualities of semantic and syntactic information on the single abstraction levels of the service description structure. To let a matchmaker learn how to best select relevant services based on the respective degrees of information quality of these descriptions, a matchmaker’s adaptation mechanism should be explicitly based on the positive impact the similarity values from different service abstraction levels will have on the retrieval results. This implies that information from all service abstraction levels should be computed and consequently combined using a weighting that indicates the impact of each level. We allow this by providing the means to automatically learn an optimal weighting based on an offline learner and providing similarity metrics for semantic and syntactic descriptions. The following paragraphs give an overview on COV4SWS.KOM, which will be further discussed in Sections 3.1 to 3.3

Determination of Similarities. For a service request and given service offers, COV4SWS.KOM returns a result set arranged in descending order regarding the computed similarity between request and offers. Information from all levels of the description structure is taken into account to calculate the overall similarity. For this, COV4SWS.KOM determines either the semantical or the syntactical similarity for different abstraction levels of a service description and aggregates the single values (cp. Section 3.2). As operations provide the essential functionality a service requester is looking for, COV4SWS.KOM makes use of an operations-focused, weighted aggregation of similarity values (cp. Section 3.1).

Aggregation of Similarities. To combine the similarity results from the different service abstraction levels, COV4SWS.KOM performs a linear regression analysis using an Ordinary Least Squares (OLS) estimator (cp. Section 3.3) [25]. We assume that the weighted linear combination of similarities on individual levels predicts the aggregated similarity of two operations, and thus, ultimately, two services. The estimator learns the optimal weightings of abstraction levels during an offline training phase. For this, a selection of service requests and service offers, along with their respective mutual relevance rating, is available for training. A subset of a test collection satisfies this condition (cp. Section 4.1).

3.1 Operations-Focused Matching

Generally, for each service request, the most relevant service offers should be identified. Following an operations-focused matching approach, the overall similarity between a service request and a service offer relates to the degree to which their respective operations match. This actually means that for each operation requested, the best matching operation in a service offer should be identified.

This leads to our overall matchmaking process as depicted in Figure 1. For each pair of operations in service request and offer, their respective input (sim_{in}), output (sim_{out}), native operation (sim_{op}), and interface (sim_{iface}) level similarity is computed using the similarity metrics presented in Section 3.2. This means that semantic or syntactic similarity is measured at every single service abstraction level based on the metrics described below. These individual similarities are then combined using specified weights, w_{in} , w_{out} , w_{op} , and w_{iface} ,

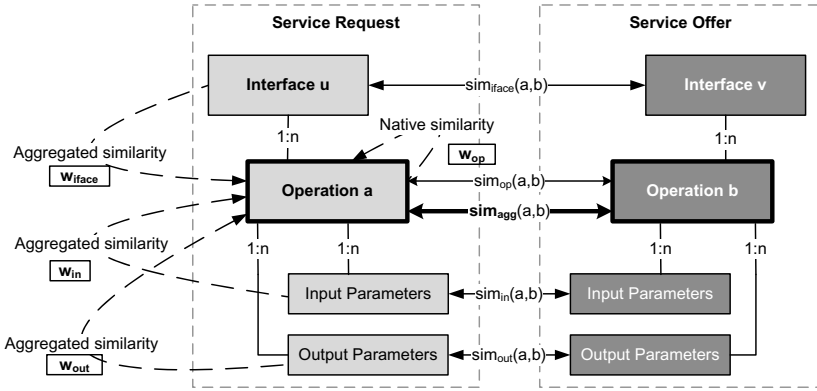


Fig. 1. Matchmaking Process in COV4SWS.KOM

resulting in an aggregated similarity value sim_{agg} for each pair of operations. Formally, for a pair of operations a and b , we define:

$$w_{iface} + w_{op} + w_{in} + w_{out} = 1 \tag{1}$$

$$sim_{agg}(a, b) = sim_{iface}(a, b) * w_{iface} + sim_{op}(a, b) * w_{op} + sim_{in}(a, b) * w_{in} + sim_{out}(a, b) * w_{out} \tag{2}$$

Once similarities between all pairs of operations in a service request and service offer have been computed, the overall service similarity sim_{serv} is derived by finding an optimal matching of operations: The final matching for a pair of services is conducted between their respective union set of operations, disregarding how the operations are organized into interfaces. Formally, let I and J be the sets of operations in a service request R and offer O , respectively. Let x_{ij} be a binary variable, indicating whether $i \in I$ has been matched with $j \in J$. Then,

$$sim_{serv}(R, O) = \frac{1}{|I|} * \sum_{i \in I, j \in J} x_{ij} * sim_{agg}(i, j) \tag{3}$$

The matching of sets of components (specifically, inputs, outputs, and operations) is based on bipartite graphs. It perceives the sets of components of a service request and offer as two partitions of nodes in a graph. Each node in the first partition is connected with each node in the second partition through a weighted edge. The edge weights correspond to the respective similarity between two components. Using the well-established *Hungarian* (or *Kuhn-Munkres*) algorithm, the bipartite graph matching algorithm computes a 1-on-1 assignment of components. Each component of the service request is matched with one component of the service offer while maximizing the overall edge weight. If a component i has been matched with a component j , $x_{ij} = 1$, else $x_{ij} = 0$. To handle differing cardinalities of the sets, an extension of the Hungarian algorithm is applied [4].

Subsequent to the matching process, the weights of all matched edges are summed up and divided by the cardinality of the original sets. This yields the similarity for two sets of components. If the cardinality of the two sets differs, the following strategy is used: Generally, the cardinality of the set associated with the service request is decisive: If an offer lacks requested operations or outputs, its overall similarity decreases. For inputs, the cardinality of the set associated with the service offer is decisive: If an offer requires more inputs than the request provides, its overall similarity decreases. Such procedure does not exclude any services due to a mismatch in the number of parameters or operations. Instead, these offers are implicitly punished by a reduction in overall similarity. The approach is based on the notion that such service offers may still be able to provide a part of the initially requested functionality or outputs, or may be invoked by providing additional inputs.

3.2 Assignment of Similarities

In almost every case, the foundation for semantic-based matchmakers is subsumption reasoning, i.e., the determination of subconcept and superconcept relationships between semantic concepts [1]. Subsumption-based matchmaking suffers from a number of drawbacks. For one, it may reward the annotation with overly generic concepts and thus may lead to suboptimal matchmaking results [2]. This makes it necessary to use further aspects to penalize overly generic annotations. Second, subsumption-based Degrees of Match (DoMs) are quite coarse-grained and do not incorporate additional information available from the ontology structure, such as the distance between two concepts or the degree of increasing specialization between levels. Third, the combination with usually numerical similarity values from IR is generally not easy to achieve but nevertheless necessary in the work at hand as described above. Last, subsumption-based DoMs rely on a ranking, which can to some degree be quite arbitrary [22].

Hence, we make use of a different approach to compute the similarity between two semantic concepts in an ontology, the so-called *semantic relatedness*. The assignment of semantic relatedness of concepts in an ontology or taxonomy is a well-known problem from computational linguistics and artificial intelligence. In contrast to the logic-based subsumption matching usually applied, non-logic-based semantic relatedness possesses a certain degree of uncertainty, as it is the case with IR-based similarity measures: semantically related objects might still not be similar and may lead to both false positives and false negatives [5]. Nevertheless, such approaches provide a multitude of well-explored methodologies and similarity measures. Thus, methods from the field of semantic relatedness might provide a significant contribution to SWS matchmaking. In fact, it has been shown elsewhere that hybrid semantic service matchmaking which combines means of logic-based and non-logic-based semantic matching can outperform each of both significantly [10].

It is quite common in the area of service matchmaking to make use of a graph representation of an ontology, where the graph nodes represent the semantic concepts and the links (edges) between the nodes represent relationships between

the concepts. The most intuitive way to compute semantic relatedness between nodes in a graph would be the measurement of the shortest distance (*path length*) between the graph nodes [5]. In the following, we refer to this measure as sim_{PL} . Furthermore, we make use of the metrics by Resnik [20] and Lin [15]:

$$sim_{Resnik}(A, B) = -\log p(anc(A, B)) \quad (4)$$

$$sim_{Lin}(A, B) = \frac{2 * \log p(anc(A, B))}{\log p(A) + \log p(B)} \quad (5)$$

For these metrics, all concepts in an ontology are augmented with values which indicate the *probability* that an instance of a concept is encountered (e.g., in a service description). The actual similarity of two concepts A and B is based on the probability p assigned to their most informative ancestor $anc(A, B)$. Probabilities are monotonically non-decreasing if moving up the taxonomy; if an ontology possesses a unique top node, its probability is 1. This can be traced back to the fact that classes inherit the probability values of their subclasses. After the probability has been determined, it is possible to derive the *information content* of $p(anc(A, B))$ which is defined as its negative log likelihood [5].

If there are no semantic concepts associated with service components or their processing fails, it might still be possible to measure the syntactic similarity of these components. E.g., it could be the case that on the message parameter level, types are semantically defined, while on operation and interface levels, only the syntax-based names of the components are available. Hence, we include a basic fallback strategy into our matchmaking approach. More precisely, the similarity between associated concept (and alternatively, component) names for a given pair of components is computed using the *WordNet* ontology [16]. Analogue to the semantic-based similarity measures, the similarity is a numerical value. Before the actual similarity can be computed, all names are tokenized. Tokens that do not correspond to a word in the WordNet ontology are additionally scanned for meaningful substrings in a recursive manner. Each set of words constitutes a partition for a bipartite graph. The edge weight corresponds to the inverse distance of a pair of words in WordNet. Consecutively, bipartite graph matching is employed, with the average edge weights in the matching yielding the similarity of the two names and thus, of two service components.

To improve the performance of matchmaking in terms of query response time and scalability, we utilize caching, namely semantic similarity, WordNet distance, and word splitting caches. Caches may be filled both at registration and query time. In the first and generally applied case, each new service offer is matched against all service offers in the repository, thus maximizing the cache population and, subsequently, the potential cache hit rate. In the latter case, only the results from matching the service queries against all service offers are stored.

3.3 OLS-Based Automatic Weight Adaption

The central question regarding information quality-aware service discovery is to which degree different abstraction levels of a service description need to be

regarded in the matchmaking process. As presented in Section 3.1, we allow the weighting of similarity values for interfaces, operations, and input and output parameters. The manual determination of such weightings is to some degree arbitrary. Furthermore, a particular weighting might be suitable for one service domain, as it reflects the information quality on the different levels of the service description structure correctly, but completely wrong for another service domain.

To account for this, COV4SWS.KOM applies an OLS estimator [25] for the determination of optimal level weights. The process is based on the notion that a dependent variable $y^{a/b}$, corresponding to the similarity of two operations a and b according to a numerical scale, can be derived through the linear combination of a set of independent variables $x_L^{a/b}$, corresponding to the individual similarity on a certain service abstraction level L when matching a and b .

In the training phase, COV4SWS.KOM matches all pairs of operations in all service requests and offers. For each pair a and b , the computed similarity on each level L yields a new entry $x_L^{a/b}$ for the *design matrix* X . Furthermore, COV4SWS.KOM retrieves the predefined similarity of operations a and b , which yields a new entry $y^{a/b}$ in the *vector of predictors* y . An example with realistic values is provided in Eq. 6 (a and b are operations in the service request; c , d and e are operations in the service offer; y is based on 4-point graded relevance).

$$(X|y) = \left(\begin{array}{cccc|c} x_{i\text{face}}^{a/c} & x_{op}^{a/c} & x_{in}^{a/c} & x_{out}^{a/c} & y^{a/c} \\ x_{i\text{face}}^{a/d} & x_{op}^{a/d} & x_{in}^{a/d} & x_{out}^{a/d} & y^{a/d} \\ x_{i\text{face}}^{a/e} & x_{op}^{a/e} & x_{in}^{a/e} & x_{out}^{a/e} & y^{a/e} \\ x_{i\text{face}}^{b/c} & x_{op}^{b/c} & x_{in}^{b/c} & x_{out}^{b/c} & y^{b/c} \\ x_{i\text{face}}^{b/d} & x_{op}^{b/d} & x_{in}^{b/d} & x_{out}^{b/d} & y^{b/d} \\ x_{i\text{face}}^{b/e} & x_{op}^{b/e} & x_{in}^{b/e} & x_{out}^{b/e} & y^{b/e} \end{array} \right) = \left(\begin{array}{cccc|c} 0.73 & 0.63 & 0.81 & 0.79 & 0.67 \\ 0.54 & 0.55 & 0.35 & 0.47 & 0.33 \\ 0.95 & 0.85 & 0.67 & 0.63 & 1.00 \\ 0.33 & 0.11 & 0.26 & 0.29 & 0.33 \\ 0.56 & 0.23 & 0.61 & 0.45 & 0.33 \\ 0.11 & 0.11 & 0.33 & 0.35 & 0.33 \end{array} \right) \quad (6)$$

Given the design matrix and vector of predictors, the standard OLS estimator can be applied [25]. It yields the initial estimate of level weights, namely the vector $\hat{\beta}$ (Eq. 7). In order to derive the final level weights, we further process the vector. First, negative level weights, which can potentially result from the OLS estimator, are set to 0, resulting in $\tilde{\beta}$ (Eq. 8). This ensures that increasing similarities on the individual levels do not have a negative impact on the aggregated similarity as it would be contradictory to common sense if higher similarity on one level resulted in diminished overall similarity. Second, the entries are normalized such that their sum matches the maximum relevance, resulting in the final vector w (Eq. 9). This ensures that a pair of operations with perfect similarity on all matching levels is precisely assigned the actual maximum relevance.

$$\begin{aligned} \hat{\beta} &= (X'X)^{-1}X'y = (\hat{\beta}_{i\text{face}}, \hat{\beta}_{op}, \hat{\beta}_{in}, \hat{\beta}_{out}) \\ &= (-0.063, 0.401, 0.506, 0.197) \end{aligned} \quad (7)$$

$$\begin{aligned} \tilde{\beta} &= (\min(0, \hat{\beta}_{i\text{face}}), \min(0, \hat{\beta}_{op}), \min(0, \hat{\beta}_{in}), \min(0, \hat{\beta}_{out})) \\ &= (\tilde{\beta}_{i\text{face}}, \tilde{\beta}_{op}, \tilde{\beta}_{in}, \tilde{\beta}_{out}) = (0, 0.401, 0.506, 0.197) \end{aligned} \quad (8)$$

$$\begin{aligned}
 w &= (\tilde{\beta}_{iface}/s, \tilde{\beta}_{op}/s, \tilde{\beta}_{in}/s, \tilde{\beta}_{out}/s) \\
 &= (w_{iface}, w_{op}, w_{in}, w_{out}) = (0, 0.363, 0.458, 0.178) \\
 s &= \tilde{\beta}_{iface} + \tilde{\beta}_{op} + \tilde{\beta}_{in} + \tilde{\beta}_{out}
 \end{aligned} \tag{9}$$

To summarize, the essential idea of the OLS estimator is to approximate each level's impact on the overall service matching result, based on the computed similarities for the different matching levels. Thus, the matchmaker can dynamically account for missing or non-discriminatory semantic annotations or syntactic descriptions on certain matching levels. The application of OLS does not inflict the runtime performance of COV4SWS.KOM, because new level weights are only learned offline once new services are added to a repository. In our evaluation involving 42 requests and 1080 offers (cp. Section 4), the learning process can be conducted in the magnitude order of ten milliseconds. In general, with n_r denoting the number of requests and n_o denoting the number of offers, the worst-case computational complexity of OLS corresponds to $\mathcal{O}(n_r * n_o)$.

4 Experimental Evaluation

4.1 Evaluation Setup

The matchmaking approach presented in Section 3 has been implemented in *COV4SWS.KOM* using Pellet 2.0¹ as reasoner and JWNL 1.4² as interface to WordNet. COV4SWS.KOM is available as part of the XAM4SWS project³.

As test data collection, SAWSDL-TC3⁴ has been adopted. SAWSDL-TC3 consists of 1080 semantically annotated WSDL 1.1-based Web services, which cover differing domains. The set contains 42 queries. A service request is defined as a service that would perfectly match the request, i.e., requests and offers are both encoded using the same formalism. Furthermore, a binary and graded relevance set for each query is provided which can be used in order to compute IR metrics. As SAWSDL-TC3 is WSDL 1.1-based, it was necessary to convert the test collection to WSDL 2.0, which is the designated service format in the work at hand. For this, a XSLT stylesheet was created⁵, based on a prototypical conversion tool by the W3C⁶. Because the conversion process does not add or remove any semantic or syntactic information, the resulting test collection can serve as a basis for comparison with WSDL 1.1 matchmakers.

In SAWSDL-TC, semantic annotations exist solely at message parameter level. As discussed in Section 3, COV4SWS.KOM incorporates information from the interface, operation, and message parameter levels of SAWSDL. This means

¹ <http://clarkparsia.com/pellet/>

² <http://jwordnet.sourceforge.net/>

³ <http://projects.semwebcentral.org/projects/xam4sws>

⁴ <http://www.semwebcentral.org/projects/sawSDL-tc>

⁵ <http://www.kom.tu-darmstadt.de/~schulte/wsd11to20.xsl>

⁶ <http://www.w3.org/2006/02/wsd11to20.xsl>

that the full potential of COV4SWS.KOM will only be revealed if the annotations address all service abstraction levels. However, SAWSDL-TC is a standard test collection for SWS matchmaking and needs to be employed to accomplish comparability with the results of other approaches. SAWSDL-TC is also used in the *International Semantic Service Selection Contest – Performance Evaluation of Semantic Service Matchmakers* (S3 Contest) [13], which serves as an annual contest to compare and discuss matchmakers for different service formalisms. Nevertheless, we assess our evaluation to be preliminary. We used SME2⁷ to compare our results with other state-of-the-art matchmaking algorithms.

We performed evaluation runs using different configurations of our matchmaker; due to space constraints, we will only present the most important evaluation runs in the following. The interested reader can download the XAM4SWS matchmaker project to conduct evaluation runs using different configurations of COV4SWS.KOM. The applied configurations are depicted in Table 1; they make use of different weightings of service abstraction levels on matchmaking results and either apply sim_{Resnik} , sim_{Lin} , or sim_{PL} , as presented in Section 3.2.

For the OLS-based computation of weightings, the actual weights are identified using k -fold cross-validation [17]. In cross-validation, $k-1$ partitions of a test data collection are applied for training purposes (i.e., the determination of weights) while the remaining partition is applied for testing purposes (i.e., matchmaking). This is repeated k times in order to apply every partition in testing; validation results are averaged over all rounds of training and testing. In the example at hand, $k=42$ since every query and corresponding relevance set from SAWSDL-TC serves as a partition from the service set. The necessary probability values for sim_{Resnik} and sim_{Lin} have been calculated based on SAWSDL-TC, i.e., we counted the appearances of semantic concepts in the service collection and derived the probabilities from this observation.

4.2 Applied Metrics

In accordance with the procedure in the S3 Contest, we evaluated the IR metrics automatically computed by SME2, namely *Average Precision* (AP'), *Q-Measure* (Q'), and *normalized Discounted Cumulative Gain* (nDCG') for each configuration of COV4SWS.KOM [13,21]. While AP' is based on binary relevance, Q' and nDCG' aim at graded relevance. As the apostrophes indicate, all numbers are adapted for incomplete relevance sets, i.e., there may exist relevant services which are *not* part of the relevance sets [21]. We deliberately refrain from the inclusion of *Average Query Response Time* (AQRT) in the evaluation results. In our opinion, the characteristics of the computer that is used for evaluation and the utilization of caches renders absolute AQRT figures largely incomparable. However, we refer the interested reader to the summary slides of the 2010 S3 Contest [13]. The summary provides a comparison of multiple matchmakers regarding the criterion of runtime performance and ranks COV4SWS.KOM – along with our other matchmaker, LOG4SWS.KOM [22] (also included in the

⁷ <http://projects.semwebcentral.org/projects/sme2/>

Table 1. Summary of Evaluation Results for COV4SWS.KOM

COV4SWS.KOM Version No.	Weighting of Levels	Similarity Metric Applied	AP'	Q'	nDCG'
1a	0.0, 0.0, 0.5, 0.5	<i>sim_{Lin}</i>	0.710	0.725	0.787
1b	0.0, 0.0, 0.5, 0.5	<i>sim_{Resnik}</i>	0.734	0.708	0.760
1c	0.0, 0.0, 0.5, 0.5	<i>sim_{PL}</i>	0.755	0.770	0.828
2a	0.1, 0.1, 0.4, 0.4	<i>sim_{Lin}</i>	0.784	0.806	0.873
2b	0.1, 0.1, 0.4, 0.4	<i>sim_{Resnik}</i>	0.796	0.791	0.851
2c	0.1, 0.1, 0.4, 0.4	<i>sim_{PL}</i>	0.806	0.825	0.878
3a	0.25, 0.25, 0.25, 0.25	<i>sim_{Lin}</i>	0.796	0.812	0.867
3b	0.25, 0.25, 0.25, 0.25	<i>sim_{Resnik}</i>	0.808	0.808	0.869
3c	0.25, 0.25, 0.25, 0.25	<i>sim_{PL}</i>	0.806	0.825	0.881
4a	OLS-based	<i>sim_{Lin}</i>	0.802	0.813	0.877
4b	OLS-based	<i>sim_{Resnik}</i>	0.823	0.825	0.884
4c	OLS-based	<i>sim_{PL}</i>	0.801	0.812	0.877

XAM4SWS project) – as the fastest contestant in the SAWSDL track. This can be traced back to the caching mechanisms applied (cp. Section 3.2).

4.3 Results and Discussion

Table 1 shows the evaluation results for the above mentioned configurations of COV4SWS.KOM. The evaluation led to somewhat heterogeneous results. Nevertheless, it is possible to derive some very important conclusions. If we compare the single similarity metrics, *sim_{Lin}* provides mostly better results than *sim_{Resnik}* for Versions 1 and 2; results for Version 3 are similar with a difference smaller than 0.01. For these three versions, *sim_{PL}* leads to the overall best results. Regarding the different versions, the signature-based Version 1 exhibits the worst results with respect to the metrics depicted in Table 1. The integration of similarity values from all service abstraction levels in Versions 2 to 4 clearly leads to an improvement of matchmaking results. Version 3 features better results than Version 2, i.e., the higher the weights for the interface and operation levels, the better the evaluation results. This shows that syntactically described service components make a very important contribution to the overall discovery results. Apart from *sim_{PL}*, the OLS-based Version 4 exhibits the best evaluation results, including the best overall results in Version 4b. Regarding *sim_{PL}*, the differences between Versions 2c, 3c, and 4c, are not significant.

Figure 2 shows the *sim_{Resnik}*-based versions (apart from Version 1b) performing quite similar over all recall levels. However, Version 4b provides better precision for recall levels 0.2-0.55, thus explaining the better results for this version. Figure 2 also shows very nicely that the inclusion of information from all service abstraction levels leads to improvements of results on all recall levels.

We have also compared COV4SWS.KOM with the most relevant contestants from the S3 Contest 2010, i.e., [8, 11, 12, 13, 19, 22]. The evaluation results of these matchmakers are depicted in Table 2. Apart from the AP', our own matchmakers LOG4SWS.KOM and COV4SWS.KOM provide the best results of all current

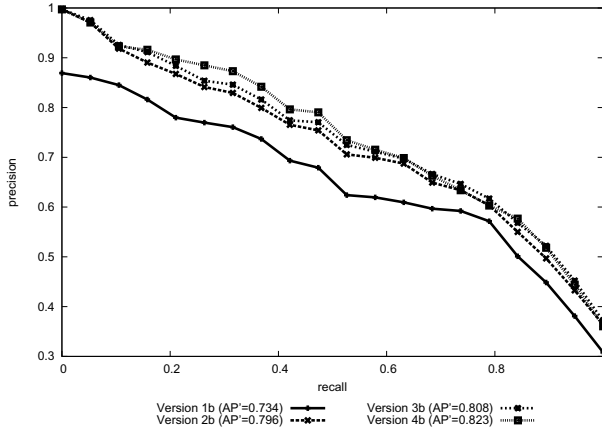


Fig. 2. Recall-Precision-Curves for sim_{Resnik} -based Versions

SAWSDL matchmakers. Based on a Friedman test (with a level of significance of $p=0.05$) [12], the differences for Q' and nDCG' for LOG4SWS.KOM and COV4SWS.KOM are not significant. Hence, the performance of LOG4SWS.KOM and COV4SWS.KOM regarding graded relevance sets (which are currently the state-of-the-art to assess retrieval algorithms in the IR community [21]), is estimated to be equal. So, we were able to show that the automatic adaptation provided by COV4SWS.KOM does provide very competitive matchmaking results: The matchmaker presented in this paper is one of the two current top matchmakers with regard to the evaluated IR metrics *and* provides an adaptation based on the provided information quality on different service abstraction levels. The latter is a feature not offered by other matchmakers.

Finally, we want to discuss the quantitative specifics of COV4SWS.KOM: There are two major differences between common matchmakers and the work at hand. First, semantic matchmaking is usually based on subsumption matching as presented by Paolucci et al. [18]. This approach applies DoMs for discrete elements in a service description and defines the *minimum* DoM found as the overall service (or operation) DoM. This leads to quite a coarse-grained, discrete scale of possible service DoMs. In order to further rank service offers based on

Table 2. Comparison of COV4SWS.KOM with State-of-the-Art Matchmakers

Matchmakers	AP'	Q'	nDCG'
COV4SWS.KOM (Version 4b)	0.823	0.825	0.884
LOG4SWS.KOM [13][22]	0.837	0.851	0.896
iSeM [11][13]	0.842	0.762	0.803
SAWSDL-MX1 [12][13]	0.747	0.767	0.839
iMatcher [8][13]	0.764	0.784	0.855
URBE [13][19]	0.749	0.777	0.850

a service request, additional techniques like text similarity need to be applied. In contrast, COV4SWS.KOM applies a continuous scale which allows a more fine-grained evaluation and ranking of services.

Second, we included an information quality-aware adaptation mechanism. The application of OLS in order to determine to which degree a particular service description level should influence the matchmaking results is an intuitive approach to adapt a matchmaker to a particular service domain. With linear regression analysis it is possible to determine which part(s) of a service level description should be weighted to a disproportionately small or large extent while achieving excellent evaluation results.

Summarized, the evaluation results show that adaptation based on information quality and the usage of metrics which are usually employed to determine semantic relatedness between concepts in ontologies is a promising strategy in order to improve ontology-based matchmaking results.

5 Related Work

Since the seminal paper of Paolucci et al. [18], a large number of different matchmaking approaches has been proposed. In the following, we will consider adaptive matchmakers for SAWSDL, which today provide the best results in terms of IR metrics. For a broader discussion, we refer to Klusch et al. – according to their classification, COV4SWS.KOM classifies as an adaptive and non-logic-based semantic matchmaker [9,13].

iMatcher applies an adaptive approach to service matchmaking by learning different weightings of linguistic-based similarity measures [8,13]. *iSeM* is an adaptive and hybrid semantic service matchmaker which combines matching of the service signature and the service specification [11]. Regarding the former, strict and approximated logical matching are applied, regarding the latter, a stateless, logical plug-in matching is deployed. In *SAWSDL-MX*, three kinds of filtering, based on logic, textual information, and structure are applied; the matchmaker adaptively learns the optimal aggregation of those measures using a given set of services [12]. Notably, COV4SWS.KOM and SAWSDL-MX/*iSeM* have been developed completely independently. *URBE* calculates the syntactic or semantic similarity between inputs and outputs [19]. Furthermore, the similarity between the associated XSD data types for a given pair of inputs or outputs is calculated based on predefined values. Weights may be determined manually.

In our former work, we have presented *LOG4SWS.KOM*, which is also a matchmaker for service formalisms like SAWSDL and hRESTS [14,22]. This matchmaker shares some features with COV4SWS.KOM, especially the fallback strategy and the operations-focused matching approach.

However, *LOG4SWS.KOM* applies a completely different strategy to assess the similarity of service components, as the matchmaker is based on logic-based DoMs respectively their numerical equivalents. Most importantly, an automatic adaptation to different qualities of syntactic and semantic information on different service abstraction levels is not arranged for.

To the best of our knowledge, COV4SWS.KOM is the first matchmaker to apply an adaptation mechanism not aiming at the filtering but on the service description structure. Other matchmakers adapt their behavior by learning how to optimally aggregate different semantic matching filters, but are nevertheless bound to particular presumptions regarding the quality of semantic and syntactic information given on the different levels of the service description. The biggest advantage of COV4SWS.KOM is the direct adaptation to information quality of descriptions on the different service abstraction levels. This feature is so far unprecedented within service matchmaking and allows the automated adaptation and application of COV4SWS.KOM within different service domains. In contrast, other matchmakers might be only applicable in these service domains matching the needs of the matchmaker regarding the provided syntactic and semantic information on every service abstraction level.

6 Conclusion

In this paper, we proposed an information quality-aware approach to service matchmaking. Through the adaptation to different degrees of impact on single service abstraction levels, it is possible to adapt our matchmaker to different service domains. For this, we discussed the usage of similarity metrics from the field of information theory and the OLS-based adaptation of the matchmaking process regarding the quality of semantic and syntactic information on different service abstraction levels. We evaluated different versions of the corresponding matchmaker COV4SWS.KOM for SAWSDL. The combination of operations-focused matching, similarity metrics from the field of information theory, and self-adaptation based on the weights of different service abstraction levels led to top evaluation results regarding IR metrics.

Acknowledgements. This work is supported in part by the E-Finance Lab e.V., Frankfurt am Main, Germany (www.efinancelab.de).

References

1. Baader, F., Nutt, W.: Basic Description Logics. In: The Description Logic Handbook: Theory, Implementation and Applications, ch. 2, pp. 47–100. Cambridge University Press (2003)
2. Bellur, U., Kulkarni, R.: Improved Matchmaking Algorithm for Semantic Web Services Based on Bipartite Graph Matching. In: 2007 IEEE International Conference on Web Services, pp. 86–93 (2007)
3. Booth, D., Liu, C.K. (eds.): Web Service Description Language (WSDL) Version 2.0 Part 0: Primer. W3C Recommendation (June 2007)
4. Bourgeois, F., Lassalle, J.C.: An extension of the Munkres algorithm for the assignment problem to rectangular matrices. *Communications of the ACM* 14(12), 802–804 (1971)
5. Budanitsky, A., Hirst, G.: Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics* 32(1), 13–47 (2006)
6. Farrell, J., Lausen, H. (eds.): Semantic Annotations for WSDL and XML Schema. W3C Recommendation (August 2007)

7. Gomadam, K., Verma, K., Sheth, A.P., Li, K.: Keywords, Port Types and Semantics: A Journey in the Land of Web Service Discovery. In: SWS, Processes and Applications, ch. 4, pp. 89–105. Springer (2006)
8. Kiefer, C., Bernstein, A.: The Creation and Evaluation of iSPARQL Strategies for Matchmaking. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) *ESWC 2008*. LNCS, vol. 5021, pp. 463–477. Springer, Heidelberg (2008)
9. Klusch, M.: Semantic Web Service Coordination. In: Schumacher, M., Helin, H., Schuldt, H. (eds.) *CASCOM: Intelligent Service Coordination in the Semantic Web*, ch. 4, pp. 59–104. Birkhäuser Verlag (2008)
10. Klusch, M., Fries, B., Sycara, K.P.: OWLS-MX: A hybrid Semantic Web service matchmaker for OWL-S services. *Journal of Web Semantics* 7(2), 121–133 (2009)
11. Klusch, M., Kapahnke, P.: iSeM: Approximated Reasoning for Adaptive Hybrid Selection of Semantic Services. 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. 30–44. Springer, Heidelberg (2010)
12. Klusch, M., Kapahnke, P., Zinnikus, I.: Hybrid Adaptive Web Service Selection with SAWSDL-MX and WSDL-Analyzer. 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. 550–564. Springer, Heidelberg (2009)
13. Klusch, M., Küster, U., König-Ries, B., Leger, A., Martin, D., Paolucci, M., Bernstein, A.: 4th International Semantic Service Selection Contest – Retrieval Performance Evaluation of Matchmakers for Semantic Web Services, S3 Contest (2010)
14. Lampe, U., Schulte, S., Siebenhaar, M., Schuller, D., Steinmetz, R.: Adaptive Matchmaking for RESTful Services based on hRESTS and MicroWSMO. In: *Workshop on Enhanced Web Service Technologies (WEWST 2010)*, pp. 10–17 (2010)
15. Lin, D.: An Information-Theoretic Definition of Similarity. In: *Fifteenth International Conference on Machine Learning*, pp. 296–304 (1998)
16. Miller, G.A.: WordNet: a lexical database for English. *Communications of the ACM* 38(11), 39–41 (1995)
17. Mitchell, T.M.: *Machine Learning*. McGraw-Hill (1997)
18. 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)
19. Plebani, P., Pernici, B.: URBE: Web Service Retrieval Based on Similarity Evaluation. *IEEE Trans. on Knowledge and Data Engineering* 21(11), 1629–1642 (2009)
20. Resnik, P.: Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language. *Artificial Intelligence Research* 11, 95–130 (1999)
21. Sakai, T., Kando, N.: On Information Retrieval Metrics designed for Evaluation with Incomplete Relevance Assessments. *Information Retrieval* 11(5), 447–470 (2008)
22. Schulte, S., Lampe, U., Eckert, J., Steinmetz, R.: LOG4SWS.KOM: Self-Adapting Semantic Web Service Discovery for SAWSDL. In: *2010 IEEE 6th World Congress on Services*, pp. 511–518 (2010)
23. Vitvar, T., Kopecký, J., Viskova, J., Fensel, D.: WSMO-Lite Annotations for Web Services. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) *ESWC 2008*. LNCS, vol. 5021, pp. 674–689. Springer, Heidelberg (2008)
24. Wang, R.Y., Strong, D.M.: Beyond Accuracy: What Data Quality Means to Data Consumers. *Management Information Systems* 12(4), 5–33 (1996)
25. Wooldridge, J.M.: *Introductory Econometrics: A Modern Approach*, 2nd edn. Thomson South-Western (2003)

Automatic Identification of Best Answers in Online Enquiry Communities

Grégoire Burel, Yulan He, and Harith Alani

Knowledge Media Institute,
The Open University, Milton Keynes, UK
{g.burel,y.he,h.alani}@open.ac.uk

Abstract. Online communities are prime sources of information. The Web is rich with forums and Question Answering (Q&A) communities where people go to seek answers to all kinds of questions. Most systems employ manual answer-rating procedures to encourage people to provide quality answers and to help users locate the best answers in a given thread. However, in the datasets we collected from three online communities, we found that half their threads lacked best answer markings. This stresses the need for methods to assess the quality of available answers to: 1) provide automated ratings to fill in for, or support, manually assigned ones, and; 2) to assist users when browsing such answers by filtering in potential best answers. In this paper, we collected data from three online communities and converted it to RDF based on the SIOC ontology. We then explored an approach for predicting best answers using a combination of content, user, and thread features. We show how the influence of such features on predicting best answers differs across communities. Further we demonstrate how certain features unique to some of our community systems can boost predictability of best answers.

Keywords: Social Semantic Web, Community Question Answering, Content Quality, Online Communities.

1 Introduction

Nowadays, online enquiry platforms and Question Answering (Q&A) websites represent an important source of knowledge for information seekers. According to Alexa,¹ 14% of Yahoo!'s traffic goes to its Q&A website whereas *Stack Exchange*² (SE) Q&A network boast an average of 3.7 million visits per day.

It is very common for popular Q&A websites to generate many replies for each posted question. In our datasets, we found that on average each question thread received 9 replies, with some questions attracting more than 100 answers. With such mass of content, it becomes vital for online community platforms to put in place efficient policies and procedures to allow the discovery of best answers. This allows community members to quickly find prime answers, and

¹ Alexa, <http://www.alexa.com>

² Stack Exchange, <http://stackexchange.com>

to reward those who provide quality content. The process adopted by Q&A systems for rating best answers range from restricting answer ratings to the author of the question (e.g. the *SAP Community Network*³ (SCN) forums), to opening it up to all community members (e.g. SE). What is common between most of such communities is that the process of marking best answers is almost entirely manual. The side effect is that many threads are left without any such markings. In our datasets, about 50% of the threads lacked pointers to best answer. Although, much research has investigated the automatic assessment of answer quality and the identification of best answers [1], little work has been devoted to the comparison of such models across different communities.

In this paper we apply a model for identifying best answers on three different enquiry communities: the *SCN forums* (SCN), *Server Fault*⁴ (SF) and the *Cooking* community⁵ (CO). We test our model using various combinations of *user*, *content*, and *thread* features to discover how such groups of features influence best answer identification. We also study the impact of community-specific features to evaluate how platform design impacts best answers identification. Accordingly, the main contributions of our paper are:

1. Perform a comparative study on performance of a typical model for best answer identification on three online enquiry communities.
2. Introduce a new set of features based on the characteristics and structure of Q&A threads.
3. Study the influence of user, content, and thread features on best answer identification and show how combining these features increases accuracy of best answer identification.
4. Investigate the impact of platform-specific features on performance of best answer identification, and demonstrate the value of public ratings for best answer prediction.

In addition to the above contributions, we also developed an ontology for representing Q&A features as well as a methodology for converting and extending our model using augmentation functions. Furthermore, we introduce several ratio features, e.g. *ratio of scores of an answers in comparison to others*. We show that such ratio features have a good impact on our model.

In the following section we analyse existing research in *best answer identification*. In the third section, the features used by our model are introduced. Following the *user*, *content* and *thread* features introduction, we present an ontology based methodology for mapping and generating our three different dataset features. The fourth section describes our best answer model and presents our results. The results and future work are discussed in section five. Finally, we conclude our paper in section six.

³ SAP Community Network, <http://scn.sap.com>

⁴ Server Fault, <http://serverfault.com>

⁵ Cooking community, <http://cooking.stackexchange.com>

2 Related Work

Many different approaches have been investigated for assessing content quality on various social media platforms [1]. Most of those approaches are based on estimating content quality from two groups of features; *content* features, and *user* attributes. Content-based quality estimation postulates that the content and the metadata associated with a particular answer can be used for deriving the value of an answer. While user features considers that behavioural information about answerers is relevant for identifying the merit of a post.

Content based assessment of quality has been applied to both textual [2,3,4] and non textual [5,2,3,4] content. Textual features normally include readability measures such as the *Gunning-Fog index*, *n-grams* or *words overlap* [3,4]. Content metadata like *ratings*, *length* and *creation date* [5,2,3,4] have also been investigated in this context. In this paper we also use common content features, such as *content length* and *Gunning-Fog index*, alongside user features and other novel features related to the online community platform.

Some approaches for assessing answer quality rely on assessing the expertise or importance of the users themselves who provided the answers. Such assessment is usually performed by applying link based algorithms such as *ExpertiseRank* [6] which incorporates user expertise with *PageRank*, and *HITS* for measuring popularity or connectivity of users [7,3,8].

Another line of research focused on identifying existing answers to new questions on Q&A systems. Ontologies and Natural Language Processing (NLP) methods have been proposed for extracting relevant entities from questions and matching them to existing answers [9,10,11,12]. Other methods involved more standard Information Retrieval (IR) techniques like Probabilistic Latent Semantic Analysis [13], query rewriting [14] and translation models [5]. Most of these works however focus on measuring the relevance of answers to questions, rather than on the quality of those answers. Other approaches analyse the role of posts, to distinguish between conversational and informational questions [15], or between questions, acknowledges, and answers [16]. Although out of the scope of this paper, such approaches could be used to filter out non-answer posts from discussion threads that could improve best answer prediction.

Our work differs from all the above in that in addition to using common content and user features, we also use thread features that take into account certain characterises of the individual threads; such as scores ratios, order of answers, etc. In addition to those features, we also present a contextual topical reputation model for estimating how knowledgeable the answerer is likely to be. Also, much of previous work concentrated on studying single communities, whereas in this paper we investigate and compare the results across three communities, thus establishing a better idea of how generic the findings are.

3 Predicting Quality of Answers

Measuring content quality and identifying best answers require the training and validation of prediction models and discovering the influence of the various

features on these predictions. For training our answer classifier, we use three main types of features; *content*, *user*, and *thread* features. All these features are strictly generated from the information available at the time of the feature extraction (i.e. future information are not taken into account while generating attributes). The different attributes are described in the following sections.

3.1 User Features

User features describe the characteristic and reputation of authors of questions and answers. Below is the list of 11 user features employed in this study.

- *Reputation*: Represents how active and knowledgeable a user is. It can be approximated from the number of good answers written by the user.
- *Age*: The user age. It measures how old is a user in years.
- *Post Rate*: Average number of questions or answers the user posts per day.
- *Number of Answers*: The number of answers posted by a user.
- *Answers Ratio*: The proportion of answers posted by a user.
- *Number of Best Answers*: The number of best answers posted by a user.
- *Best Answers Ratio*: The proportion of best answers posted by a user.
- *Number of Questions*: The number of questions posted by a user.
- *Questions Ratio*: The proportion of questions posted by a user.
- *Normalised Activity Entropy*: A normalised entropy measure (H_a) represents how predictable is the activity of a user. In enquiry platforms, a user u_i can either post questions (Q) or answers (A). Lower entropy indicates focus on one activity. The normalised activity entropy is calculated from the probabilities of a user posting answers or questions:

$$H_A(u_i) = -\frac{1}{2} (P(Q|u_i) \log P(Q|u_i) + P(A|u_i) \log P(A|u_i)) \quad (1)$$

- *Normalised Topic Entropy*: Calculates the concentration (H_T) of a user's posts across different topics. Low entropy indicates focus on particular topics. In our case, topics are given by the tags associated with a question or the category of the post. Each user's tags T_{u_i} are derived from the topics attached to the questions asked or answered by the user. This can be used to calculate the probability $P(t_j|u_i)$ of having a topic t_j given a user u_i :

$$H_T(u_i) = -\frac{1}{|T_{u_i}|} \sum_{j=1}^{|T_{u_i}|} P(t_j|u_i) \log P(t_j|u_i) \quad (2)$$

- *Topical Reputation*: A measure of the user's reputation with a particular post. It is derived from the topics T_{q_k} associated with the question q_k for which the post belongs. By adding the score values of each user's answers $S(a)$, where $a \in A_{u_i, t_j}$, about a particular topic t_j , we obtain the general user topical reputation $E_{u_i}(t_j)$ for a particular topic. Given a post user u_i ,

the user topical reputation function E_{u_i} and a question q with a set of topics T_q , the reputation embedded within a post related to question q is given by:

$$E_P(q, u_i) = \sum_{j=1}^{|T_q|} E_{u_i}(t_j) \quad (3)$$

$$E_{u_i}(t_j) = \sum_{a \in A_{u_i, t_j}} S(a) \quad (4)$$

3.2 Content Features

Content features represent the attributes of questions and answers, and can be used for estimating the quality of a particular question or answer as well as their importance. We use the following content features in our analysis:

- *Score*: Represents the rating of an answer, and it normally collected from users in the form of *votes* or *thumbs up/thumbs down* flags.
- *Answer Age*: Difference between the question creation date and the date of the answer.
- *Number of Question Views*: The number of views or hits on a question.
- *Number of Comments*: The number of comments associated with a post.
- *Number of Words*: The number of words contained in a post.
- *Readability with Gunning Fog Index*: Used to measure post readability using the Gunning index of a post p_i which is calculated using the average sentence length asl_{p_i} and the percentage of complex words pcw_{p_i} :

$$G_{p_i}(asl_{p_i}, pcw_{p_i}) = 0.4 (asl_{p_i} + pcw_{p_i}) \quad (5)$$

- *Readability with Flesch-Kincaid Grade*: Calculated from the average number of words per sentence $awps_{p_i}$ and average number of syllables per word $aspw_{p_i}$ of a post p_i :

$$FK_{p_i}(awps_{p_i}, aspw_{p_i}) = 0.39 awps_{p_i} + 11.8 aspw_{p_i} - 15.59 \quad (6)$$

3.3 Thread Features

Our final set of features represents relations between answers in a particular thread. Each question tends to have more than one answer and most Q&A platforms allow only one answer to be selected as the best answer. As a consequence, each answer competes for being the best answer. In such context, relational features such as the proportion of votes to a particular answer can be used for estimating the relative importance of a particular post.

- *Score Ratio*: The proportion of scores given to a post from all the scores received in a question thread.
- *Number of Answers*: Number of answers received by a particular question.

- *Answer position*: The absolute order location of a given answer within a question thread (e.g. first, second).
- *Relative Answer Position*: The relative position of an answer within a post thread. Given a question q , its answers a_q , and the position of an answer $pos_{a_{q_i}}$, the relative answer position of an answer a_{q_i} is given by:

$$RP(a_{q_i}) = 1 - \frac{pos_{a_{q_i}}}{|a_q|} \quad (7)$$

- *Topical Reputation Ratio*: The proportion of topical reputation associated with a particular answer. Given the sum of topical reputation of all the answers, the ratio of topical reputation attributed to a particular answer.

3.4 Core vs Extended Feature Sets

As mentioned, we want to investigate the impact of platform-specific features on predictability of best answers. Hence the features above contain some that are not common across our datasets. For example, in SCN only the owner of a question can rate its answers, and select the best answer, whereas in SF and CO communities anyone with over 200 points of reputation can vote for any answer, and hence the selections of best answers can emerge collectively. The platform that supports SF and CO offer more features than SCN. In Table 3.4 we list the *core features set*, which is shared across all three datasets, and the *extended features set*, which is only valid for SF and CO datasets.

Table 1. Differences between the Core Features Set and the Extended Features Set

Type	Features Set	
	Core Features Set (19)	Extended Features Set [†] (23)
User	<i>Reputation, Post Rate, Normalised Activity Entropy, Number of Answers, Answers Ratio, Number of Best Answers, Best Answers Ratio, Number of Questions, Questions Ratio, Normalised Topic Entropy, Topical Reputation.</i> (10)	<i>Reputation, Age, Post Rate, Normalised Activity Entropy, Number of Answers, Best Answers Ratio, Number of Questions, Questions Ratio, Normalised Topic Entropy, Topical Reputation.</i> (11)
Content	<i>Answer Age, Number of Question Views, Number of Words, Gunning Fog Index, Flesch-Kinkaid Grade Level.</i> (5)	<i>Score, Answer Age, Number of Question Views, Number of Comments, Number of Words, Gunning Fog Index, Flesch-Kinkaid Grade Level.</i> (7)
Thread	<i>Number of Answers, Answer Position, Relative Answer Position, Topical Reputation Ratio.</i> (4)	<i>Score, Number of Answers, Answer Position, Relative Answer Position, Topical Reputation Ratio.</i> (5)

[†]Only valid for the *Server Fault* and *Cooking* datasets.

4 Datasets

Our experiments are conducted on three different datasets. The first two are subs communities extracted from the April 2011 *Stack Exchange* (SE) public datasets:⁶ *Server Fault* (SF) user group and the non technical *Cooking* website

⁶ As part of the public *Stack Exchange* dataset, the *Server Fault* and *Cooking* datasets are available online at <http://www.clearbits.net/get/1698-apr-2011.torrent>

(CO) composed of cooking enthusiasts. The other dataset is obtained from the *SAP Community Network* (SCN) forums and consists of posts submitted to 33 different forums between December 2003 and July 2011.⁷

4.1 SAP Community Network

The *SAP Community Network* (SCN) is a set of forums designated for supporting SAP customers and developers. SCN integrates traditional Q&A functionalities systems such as best answer selection, user reputation and moderation. Each SCN thread is initiated with a question and each answer in that thread is a reply to that question. Thread authors can assign a limited number of points to the answers they like (unlimited two-points for *helpful answers*, two sets of six-points for *very helpful answers* and one ten-points for the *best answer*). Points given to answers add to the reputation of their authors. Users can be flagged as topic experts, get promoted to moderators, or be invited to particular SAP events if their online reputation is high.

Our dataset consists of 95,015 threads and 427,221 posts divided between 32,942 users collected from 33 different forums between December 2003 and July 2011. Within those threads, we only select threads that have best answers. Our final dataset consists of 29,960 (32%) questions and 111,719 (26%) answers.

4.2 Server Fault

Server Fault (SF) is a Q&A community of IT support professionals and is hosted on the SE platform. SE provides social features such as *voting*, *reputation* and *best answer selection* while making sure that each posted answer is self-contained. However, SF differences reside in its rewarding program where each user gains access to additional features like ability to vote and advertising removal depending on their reputation.

Compared to SCN, SF editing policy is completely community driven. Depending on the user reputation, each community member is allowed to refine other people's questions and answers. Hence, instead of adding additional posts for elaborating questions or answers, SF users can directly edit existing content.

To keep the community engaged, the SF platform offers rewards and badges for various type of contributions. For example, users can earn the *Autobiographer* badge if they fill their profiles completely. SF users' reputation is calculated from the votes that have been cast on a particular question or answer. For each post, community members vote up or down depending on the quality and usefulness that is then pushed to the post owner. As community members gain/lose reputation, they gain/lose particular levels and abilities. Our SF dataset is extracted from the April 2011 public dataset, and consists of 71,962 questions, 162,401 answers and 51,727 users. Within those questions we selected only the questions that have best answers. The final SF dataset consist of 36,717 (51%) questions and 95,367 (59%) answers.

⁷ SAP is planning to migrate their community to a new platform in February 2012, with several new features that were not available at the time of our data collection.

4.3 Cooking Websites

The *Cooking* community (CO) is composed of enthusiasts seeking cooking advice and recipes. It is also hosted on the SE platform and thus shares the same attributes and functionalities as SF above. CO is a smaller dataset with 3,065 questions, 9,820 answers and 4,941 users. Similarly to the other datasets, we only select the questions that have best answers. The final dataset is composed of 2,154 (70%) questions and 7,039 (72%) answers.

4.4 Features Inferencing and Representation

Our three datasets come in different formats and structures. To facilitate their representation, integration, and analysis, we converted all three datasets to a common RDF format and structure (Figure 1). Data is dumped into an SQL database (1) then converted to RDF based on SIOC⁸ ontology using the D2RQ⁹ (2). RDF is then loaded into a triple store where knowledge augmentation functions are executed (3). Such functions simply extend the knowledge graph of each dataset by adding additional statements and properties (i.e. topical reputation, answer length, votes ratio, etc.). This workflow serves as the input of the learning algorithms used for predicting content quality (4). We extended SIOC to represent Q&A vocabulary¹⁰. The flexibility of RDF enabled us to add features without requiring schema redesign. Summary of mappings of our datasets to SIOC classes is illustrated in Table 5.

5 Best Answer Identification

Ability to accurately identify best answers automatically is not only a complement to the fitness and preciseness of the prediction model, but also to the fit of the community and platform features that are enabling such task to be performed accurately. If a platform fails to support the gathering of information that correlates with content quality, then automating content quality prediction becomes much harder. More importantly, such difficulty will also be faced by the users who need to quickly find the best solving answers to their problems.

The experiment described next aims at measuring the importance of our core and extended feature sets for best answer prediction, as well as highlighting how each feature impacts prediction accuracy in a given platform.

5.1 Experimental Setting

In our experiments we train a categorical learning model for identifying the best answers in our three datasets. For each thread, the *best answer* annotation is used for training and validating the model. Because SCN *best answer* annotation is

⁸ SIOC Ontology, <http://sioc-project.org>

⁹ D2RQ Platform, <http://www4.wiwiwiss.fu-berlin.de/bizer/d2rq>

¹⁰ Q&A Vocabulary, <http://purl.org/net/qa/ns#>

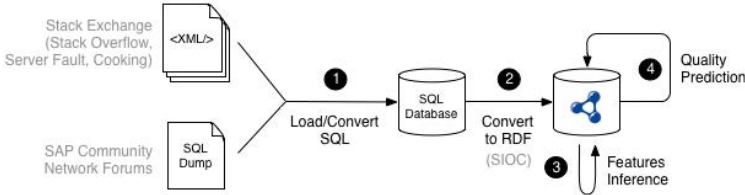


Fig. 1. Dataset Conversion and Inferencing Workflow

Table 2. SIOC Class Mappings of the *Stack Exchange* and *SCN Forums* Datasets

Input Dataset		
SCN	SF and CO	RDF Output
User	User	sioc:OnlineAccount/foaf:Person
Thread (first thread Post)	Question	sioc:Question
Post (not in first position)	Answer	sioc:Answer
Post (with 10 points)	Best Answer	sioc:BestAnswer
-	Comment	sioc:Comment
Forum	Tag	sioc:Tag (topic)

based on the author ratings, we use the *best answer* rating (i.e. 10) as the model class and discard the other ratings (i.e. 2 and 6) for training the SCN model.

A standard 10-folds cross validation scheme is applied for evaluating the generated model. Each model uses the features described earlier in the paper. Decision tree algorithms have been found to be the most successful in such contexts [3,17]. We use the *Multi-Class Alternating Decision Tree* learning algorithm due to its consistent and superior results to other decision tree algorithms we tested (*J48*, *Random Forests*, *Alternating Tree* and *Random Trees*).

To evaluate the performance of the learning algorithm, we use precision (P), recall (R) and the harmonic mean F-measure (F_1) as well as the area under the Receiver Operator Curve (ROC) measure. The precision measure represents the proportion of retrieved best answers that were real best answers. Recall measures the proportion of best answers that were successfully retrieved. We also plot the ROC curve and use the Area Under the Curve (AUC) metrics for estimating the classifier accuracy.

We run two experiments, the first compare the performance of our model for identifying best answers across all three datasets, using the core and extended feature sets. The second experiment focuses on evaluating the influence of each features on best answers identification.

5.2 Results: Model Comparison

For our first experiment, we train the *Multi-Class Alternating Decision Tree* classifier on different features subsets and compare the results using the metrics that we described in the previous section (Table 5.2).

Baseline Models: We used the *number of words* feature to train a baseline model since it was argued to be a good predictor [53]. Additionally, for the SF

Table 3. Average *Precision*, *Recall*, F_1 and *AUC* for the *SCN Forums*, *Server Fault* and *Cooking* datasets for different feature sets and extended features sets (marked with +) using the *Multi-Class Alternating Decision Tree* classifier

Feature	SCN Forums				Server Fault				Cooking			
	<i>P</i>	<i>R</i>	F_1	<i>AUC</i>	<i>P</i>	<i>R</i>	F_1	<i>AUC</i>	<i>P</i>	<i>R</i>	F_1	<i>AUC</i>
Words	0.536	0.732	0.619	0.616	0.592	0.621	0.537	0.567	0.671	0.705	0.644	0.651
Answer Score	-	-	-	-	0.643	0.656	0.625	0.673	0.751	0.760	0.753	0.797
Answer Score Ratio	-	-	-	-	0.808	0.809	0.806	0.848	0.866	0.868	0.866	0.916
Users	0.716	0.746	0.687	0.752	0.637	0.651	0.626	0.664	0.687	0.714	0.681	0.686
Content	0.712	0.740	0.659	0.678	0.647	0.659	0.628	0.679	0.708	0.727	0.707	0.754
Thread	0.820	0.827	0.817	0.865	0.753	0.756	0.749	0.809	0.765	0.772	0.751	0.785
All	0.833	0.839	0.831	0.880	0.770	0.769	0.760	0.827	0.777	0.784	0.767	0.816
Users+	-	-	-	-	0.637	0.651	0.626	0.664	0.687	0.714	0.681	0.686
Content+	-	-	-	-	0.700	0.707	0.699	0.761	0.788	0.793	0.789	0.842
Thread+	-	-	-	-	0.844	0.845	0.844	0.910	0.867	0.869	0.867	0.919
All+	-	-	-	-	0.848	0.847	0.844	0.912	0.870	0.872	0.870	0.919

and CO datasets, we also train another basic model based on *answer scores* and *answer scores ratios* since such features are normally especially designed as a rating of content quality and usefulness.

Surprisingly, our results from all three datasets do not confirm previous research on the importance of content length for quality prediction. For each of our datasets, *precision* and *recall* were very low with a F_1 median of 0.619 (SCN: 0.619/SF: 0.537/Cooking: 0.644). This might be due to the difference of our data to those from literature which were taken from general Q&A communities such as Yahoo! Answers [3] and the Naver community [5]).

The SF and CO models trained on the *answer scores* highlight positive correlations between *best answers* and *scores*. However, this positive influence is reduced when the data grow in SF over CO. CO shows high F_1 results with 0.753 with Answer Score, whereas SF result is 0.625. Training the SE models on *Answer Score Ratios* shows even higher results with a F_1 of 0.806 for SF and 0.866 for *Cooking*. Overall, *answer score ratio* appear to be a good predictor for answer quality which shows that SF and CO collaborative voting models are effective. In particular, it shows that taking into account the relative voting proportions between answers (i.e. *scores ratio*) is a better approach than only considering absolute *scores*.

Core Features Models: Here we focus on the comparison of feature types (i.e. *users*, *content* and *threads*) and the impact of using the extended feature set on the identification process. We trained a model for each dataset and features set. Results in Table 5.2 show that using the thread features we introduced in this paper increases accuracy in all three datasets over user and content features. Results also show that F_1 when combining all core user, content, and thread features was 11%, 9.3%, and 5.4% higher for SCN, SF, and CO respectively, than the best F_1 achieved when using these features individually.

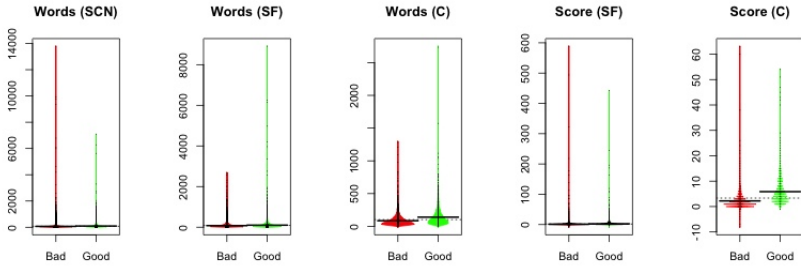


Fig. 2. Bean Plots representing the distribution of different features and best answers for the *SCN Forums* (SCN), the *Server Fault* (SF) and *Cooking* (C) datasets

Overall, when using all the core features (common to all datasets), SCN performed better than SF (+7.1%) and CO (+6.4%). Predictions for CO were slightly more accurate than for SF, probably due to its smaller size. However, results in Table 5.2 show that F_1 with all core features is lower than the *Answer Score Ratio* by 4.6% for SF and 9.9% for CO. This reflects the value of this particular feature for best answer identification on such platforms.

Figure 2 shows the distributions of best answers (good) and non-best answers (answer) for posts length for all our datasets and answer scores for SF and CO. Best answers seem to likely be shorter in SCN, and longer in SF and CO. This variation could be driven by the difference in data sizes and topics as well as external factors such as community policies (e.g. community editing in SE).

Extended Features Models: Now we recompute the models using extended *users*, *content* and *threads* feature sets. Remember that the extended features (Table 3.4) are only supported by SF and CO. No change in accuracy can be witnessed when extending the user features. However, F_1 increases by average of 8.3% for SF and 14.9% for CO when extending content and thread features. The only difference between the core user features and extended ones is the *user age* attribute. Hence the age of the answerer does not seem to have an effect on best answers identification. As for extended content and thread features, they contain extra features such as *number of comments* and *scores*, as well as the *scores ratios* which we compute per thread.

Table 5.2 shows that the F_1 for SF and CO when using all extended features combined (*All+* in 5.2) has increased by 8.4% and 10.3% for SF and CO respectively over using core features (*All* row in Table 5.2). This is mainly due to the addition of the *scores/ratings* based features. Furthermore, F_1 from the combined extended features was even higher than the *Answer Score Ratio* model, by 3.8% for SF and a mere 0.4% for CO.

In general, we can see that thread features are consistently more beneficial than others for identifying best answers. When available, scoring (or rating) features improve prediction results significantly, which demonstrates the value of community feedback and reputation for identifying valuable answers.

5.3 Results: Feature Comparison

Following on from the previous experiments, our second round of analysis focus on evaluating the importance of each feature for best answer identification. For each dataset, we rank all our predictors using Information Gain Ration (IGR) with respect to the best answers annotations. The top 15 are shown in Table 5.3.

Table 4. Top features ranked by Information Gain Ratio for the *SCN*, *Server Fault* and *Cooking* datasets. Type of feature is indicated by *U/C/T* for *User/Content/Thread*

R.	SCN		Server Fault		Cooking	
	IG	Feature	IG	Feature	IG	Feature
1	0.217	<i>Topic. Rep. Ratio (T)</i>	0.332	<i>Score Ratio (T)</i>	0.430	<i>Score Ratio (T)</i>
2	0.196	<i>No. Answers (T)</i>	0.275	<i>No. Answers (T)</i>	0.190	<i>Score (C)</i>
3	0.108	<i>Bests Ratio (U)</i>	0.126	<i>Answer Position (T)</i>	0.164	<i>No. Answers (T)</i>
4	0.105	<i>Questions Ratio (U)</i>	0.117	<i>Topic. Rep. Ratio (T)</i>	0.120	<i>Answer Position (T)</i>
5	0.105	<i>Answers Ratio (U)</i>	0.097	<i>Relative Position (T)</i>	0.083	<i>Topic. Rep. Ratio (T)</i>
6	0.104	<i>Relative Position (T)</i>	0.070	<i>Score (C)</i>	0.074	<i>Bests Ratio (U)</i>
7	0.097	<i>Reputation (U)</i>	0.056	<i>Q. Views (C)</i>	0.070	<i>No. Bests (U)</i>
8	0.093	<i>Topic. Rep. (U)</i>	0.046	<i>Bests Ratio (U)</i>	0.069	<i>Reputation (U)</i>
9	0.090	<i>No. Bests (U)</i>	0.037	<i>No. Comments (C)</i>	0.065	<i>Answer Age (C)</i>
10	0.089	<i>Activity Entropy (U)</i>	0.022	<i>Topic Entropy (U)</i>	0.055	<i>Topic Entropy (U)</i>
11	0.064	<i>Answer Position (T)</i>	0.021	<i>Answer Age (C)</i>	0.054	<i>No. Comments (C)</i>
12	0.048	<i>No. Answers (U)</i>	0.019	<i>Post Rate (U)</i>	0.054	<i>No. Words (C)</i>
13	0.035	<i>Topic Entropy (U)</i>	0.018	<i>Reputation (U)</i>	0.053	<i>No. Answers (U)</i>
14	0.033	<i>Q. Views (C)</i>	0.017	<i>No. Bests (U)</i>	0.045	<i>Relative Position (T)</i>
15	0.027	<i>No. Words (C)</i>	0.016	<i>No. Answers (U)</i>	0.039	<i>Topic. Rep. (U)</i>

Core Features: First we focus the analysis on the *core features set*. Table 5.3 shows that SCN’s most important feature for best answer identification appear to be the *topical reputation ratio*, which also came high up the list with 3rd rank in SF and 5th in CO. The *number of answers* also comes high in each dataset: 2nd for SCN and SF, and 3rd for CO. Note that our training datasets only contained threads with best answers. Hence the shorter the thread is (i.e. less answers) the easier it is to identify the best answer. Similarly, *best answers ratio* and *number of best answers* also proved to be good features for best answer prediction. Figure 3 shows the correlations with best answers (good) and non-best answers (bad) for the top five features in each datasets.

Distribution of SCN *topical reputation* in Figure 3 is narrower than the distribution of SF and CO. This highlights the difference between the SCN and SE reputation models. Contrary to SE, SCN only allow positive reputation. For core features, SF, CO, and SCN have a generally similar mode of operation. However, SCN is less affected by *answer position* due to the difference of platform editing policies. SE favours small thread whereas SCN does not. Such difference leads to a better correlation of *number of answers* with best answers in SE.

According to Table 5.3, user features appear to be dominant, with some thread features amongst the most influential. Number of thread answers and historical activities of users are particularly useful (e.g. number and ratio of user’s best answers). User reputation in SCN plays a more important role than in SF and

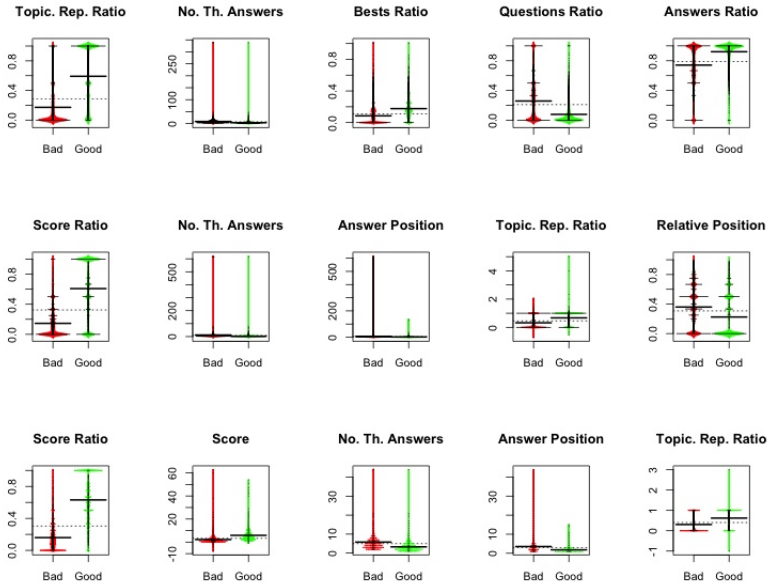


Fig. 3. Bean Plots representing the distribution of different the top five features for the *SCN Forums* (first row), the *Server Fault* (second row) and *Cooking* (third row) datasets

CO, which is probably a reflection of the community policies that puts emphasis on members' reputation.

In SAP's SCN, user activity focus seems to play a notable role (*topical reputation, answer and question ratios, activity entropy, etc.*). These features are further down the list for SF and CO.

Extended Features: The evaluation of extended features establishes the importance of *scores*. For SF and CO datasets, it is clear once again that the *score* features are the most important for identifying best answers.

SF has a *score ratio* IG of 0.332 and CO have IG score of 0.430 representing respectively around +5.7% and +24% more gain than the second ranked feature.

As in the general model evaluation, thread features compare the score of a single answer with the score of other thread answers. The higher the ratio, the better the answer. Note that the selection of best answers in SF and CO is left to the user who posted the question, who may or may not consider the scores given by the community or general site visitors.

6 Discussion and Future Work

Understanding which features impact best answer identification is important for improving community platform designs. However, different types of online communities tend to have different characteristics, goals, and behaviours. It is

therefore difficult to generalise any findings without a broad base of experimentation. Such observation is reinforced with the difference between our findings and previous research [53] concerning the value of content length. In this work, we widened our analysis to three communities to give our findings more scope. Our communities bear much similarity in terms of type, goals, and properties, and hence we can argue that our findings are transferable across such communities.

Reliable automatic identification of best answers can solve the common problem of scarcity of such valuable information in most online Q&A communities. However, automated methods must also find out when no best answers exist in a given thread. To this end, we are developing measures of answer quality, and will test them on threads with and without best answers. This will help ensuring that no best answers are enforced when none are above a certain quality.

Identifying best answers becomes more important the longer the threads are. It might be worth focusing such analysis on threads with more than one answer. It is worth mentioning that in SCN, SF, and CO datasets, the median number of answers per thread was 5, 3, and 4 respectively, with averages of 13, 8.5, 5.

For the SF and CO communities, we showed that the ratings given by community members to existing answers were good predictors of best answers. Although only the authors of questions can currently pick the best answers, their choices seem to be positively correlated with those of the public. Our results showed that the accuracy of using public ratings for best answer selection can be improved further when other features are considered. SCN currently lacks this feature altogether. Interestingly, SAP's SCN is migrating itself to the Jive Engage platform¹¹ in 2012. Jive offers many social features, including collaborative rating of answers.

7 Conclusions

Many popular online enquiry communities receive thousands of questions and answers on daily basis. Our work identified that around 50% of posted questions do not have best answers annotations, thus forcing site visitors to check all existing answers for identifying correct answers. We studied three online Q&A communities to learn about the influence of the various features they have on our automated best answer identification model which is based on a wide selection of *user*, *content* and *thread* features. Some of those features were common across all three communities, and some were community-specific. We achieved 83% accuracy with SCN community, 84% with SF and 87% with CO.

We found out that contrary to previous work [53], *answer length* seems uncorrelated with best answers. We also discovered that best answers in communities that support community-based answer ratings (i.e. SF and CO) can be identified much more accurately, with over 0.8 F_1 using this feature alone (*answer score ratio*). Our thread-based features proved to be very influential for best answer identification in all three communities.

¹¹ Jive Software, <http://jivesoftware.com>

Acknowledgments. This work is funded by the EC-FP7 project Robust (grant number 257859). The authors would like to thank SE for publicly sharing their data. Also thanks to Adrian Mocan from SAP for providing SCN data to ROBUST project.

References

1. Chai, K., Potdar, V., Dillon, T.: Content quality assessment related frameworks for social media. In: Proc. Int. Conf. on Computational Science and Its Applications (ICCSA), Heidelberg (2009)
2. Liu, Y., Agichtein, E.: You've got answers: towards personalized models for predicting success in community question answering. In: Proc. 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers, Columbus, Ohio (2008)
3. Agichtein, E., Castillo, C., Donato, D., Gionis, A., Mishne, G.: Finding high-quality content in social media. In: First ACM Int. Conf. on Web Search and Data Mining, Palo Alto, CA (2008)
4. Bian, J., Liu, Y., Zhou, D., Agichtein, E., Zha, H.: Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In: Int. World Wide Web Conf., Madrid (2009)
5. Jeon, J., Croft, W.B., Lee, J.H., Park, S.: A framework to predict the quality of answers with non-textual features. In: SIGIR, Washington. ACM Press (2006)
6. Zhang, J., Ackerman, M., Adamic, L.: Expertise networks in online communities: structure and algorithms. In: Proc. 16th Int. World Wide Web Conf., Banff (2007)
7. Jurczyk, P., Agichtein, E.: Discovering authorities in question answer communities by using link analysis. In: ACM 16th Conf. Information and Knowledge Management, CIKM 2007 (2007)
8. Suryanto, M., Lim, E., Sun, A., Chiang, R.: Quality-aware collaborative question answering: methods and evaluation. In: Proc. 2nd ACM Int. Conf. on Web Search and Data Mining, Barcelona (2009)
9. McGuinness, D.: Question answering on the semantic web. *IEEE Intelligent Systems* 19(1) (2004)
10. Narayanan, S., Harabagiu, S.: Question answering based on semantic structures. In: Proc. 20th Int. Conf. on Computational Linguistics, Geneva (2004)
11. Lopez, V., Pasin, M., Motta, E.: AquaLog: An Ontology-Portable Question Answering System for the Semantic Web. In: Gómez-Pérez, A., Euzenat, J. (eds.) *ESWC 2005*. LNCS, vol. 3532, pp. 546–562. Springer, Heidelberg (2005)
12. Wang, Y., Wang, W., Huang, C.: Enhanced semantic question answering system for e-learning environment. In: 21st Int. Conf. on Advanced Information Networking and Applications Workshops, AINAW, vol. 2 (2007)
13. Qu, M., Qiu, G., He, X., Zhang, C., Wu, H., Bu, J., Chen, C.: Probabilistic question recommendation for question answering communities. In: Proc. 18th Int. World Wide Web Conf., Madrid (2009)
14. Kwok, C., Etzioni, O., Weld, D.: Scaling question answering to the web. *ACM Transactions on Information Systems (TOIS)* 19(3) (2001)

15. Harper, F., Moy, D., Konstan, J.: Facts or friends?: distinguishing informational and conversational questions in social Q&A sites. In: Proc. 27th Int. Conf. on Human Factors in Computing Systems, CHI, Boston, MA (2009)
16. Kang, J., Kim, J.: Analyzing answers in threaded discussions using a Role-Based information network. In: Proc. IEEE Int. Conf. Social Computing, Boston, MA (2011)
17. Rowe, M., Angeletou, S., Alani, H.: Predicting Discussions on the Social Semantic Web. 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. 405–420. Springer, Heidelberg (2011)

Characterising Emergent Semantics in Twitter Lists

Andrés García-Silva¹, Jeon-Hyung Kang², Kristina Lerman²,
and Oscar Corcho¹

¹ Ontology Engineering Group,
Facultad de Informática, Universidad Politécnica de Madrid, Spain
{hgarcia,ocorcho}@fi.upm.es

² Information Sciences Institute,
University of Southern California, USA
{jeonhyuk,lerman}@isi.edu

Abstract. Twitter lists organise Twitter users into multiple, often overlapping, sets. We believe that these lists capture some form of emergent semantics, which may be useful to characterise. In this paper we describe an approach for such characterisation, which consists of deriving semantic relations between lists and users by analyzing the co-occurrence of keywords in list names. We use the vector space model and Latent Dirichlet Allocation to obtain similar keywords according to co-occurrence patterns. These results are then compared to similarity measures relying on WordNet and to existing Linked Data sets. Results show that co-occurrence of keywords based on members of the lists produce more synonyms and more correlated results to that of WordNet similarity measures.

1 Introduction

The active involvement of users in the generation of content on the Web has led to the creation of a massive amount of information resources that need to be organized so that they can be better retrieved and managed. Different strategies have been used to overcome this information overload problem, including the use of tags to annotate resources in folksonomies, and the use of lists or collections to organize them. The bottom-up nature of these user-generated classification systems, as opposed to systems maintained by a small group of experts, have made them interesting sources for acquiring knowledge. In this paper we conduct a novel analysis of the semantics of emergent relations obtained from Twitter lists, which are created by users to organize others they want to follow.

Twitter is a microblogging platform where users can post short messages known as tweets. Twitter was started in 2006 and has experienced a continuous growth since then, currently reaching 100 million users¹. In this social network users can follow other users so that they can receive their tweets. Twitter users

¹ <http://blog.twitter.com/2011/09/one-hundred-million-voices.html>

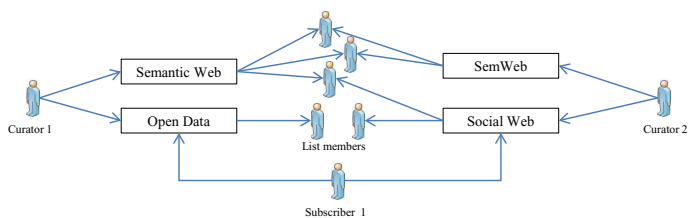


Fig. 1. Diagram showing different user roles in twitter lists. Boxes indicate list names.

are allowed to classify people into lists (see figure 1). The creator of the list is known as the curator. List names are freely chosen by the curator and consist of keywords. Users other than the curator can then subscribe to receive tweets from the listed users. Similarly to what happens with folksonomies [7,19], the classification system formed by connections between curators, subscribers, listed users, and list names, can be considered as a useful resource for knowledge extraction. In this work we analyze term co-occurrence patterns in these lists to identify semantic relations between all these elements. Co-occurrence may happen due to the simultaneous use of keywords in different lists created by curators, or in lists followed by subscribers, or in lists under which users are listed.

For instance, table 1 summarizes the lists under which an active and well known researcher in the Semantic Web field has been listed. The first column presents the most frequent keywords used by curators of these lists, while the second column shows keywords according to the number of subscribers. We can see that *semantic_web* and *semweb* are frequently used to classify this user, which suggests a strong relationship between both keywords. In fact, these keywords can be considered as synonyms since they refer to same concept. Though less frequent, other keywords such as *semantic*, *tech* and *web_science* are also related to this context. The other keywords according to the use given by subscribers (e.g., *connections*) are more general and less informative for our purposes.

We consider that Twitter Lists represent a potentially rich source for harvesting knowledge, since they connect curators, members, subscribers and terms. In this paper we explore which of such connections lead to emergent semantics and produce most related terms. We analyze terms using the vector space model [24] and a topic modeling method, the Latent Dirichlet Allocation [5]. Then we use metrics based on the WordNet synset structure [10,26,16] to measure the semantic similarity between keywords. In addition, we ground keywords to Linked Open Data and present the relations found between them. This type of analysis lays the foundation for the design of procedures to extract knowledge from Twitter lists. For instance, ontology development can benefit of the emerging vocabulary that can be obtained from these user generated sources.

In the following we present the models used to obtain relation between keywords from Twitter lists. In section 3 we introduce the similarity metrics based on WordNet, and we describe the technique used to gather relations from linked data. Next we present, in section 4, the results of our study. Finally we describe the related work in section 5, and present the conclusions in section 6.

Table 1. Most frequent keywords found in list names where the user has been listed

Curators		Subscribers	
semantic_web	39	semantic_web	570
semweb	22	semweb	100
semantic	7	who-my-friends-talk-to	93
tech	7	connections	82
web_science	5	rock_stars	55

2 Obtaining Relations between Keywords from Lists

We use the vector space model [24] to represent list keywords and their relationships with curators, members and subscribers. Each keyword is represented by three vectors of different dimension according to the type of relation represented. The use of vectors allows calculating similarity between them using standard measures such as the angle cosine.

Twitter lists can be defined as a tuple $TL = (C, M, S, L, K, R_l, R_k)$ where C, M, S, L , and K are sets of curators, members (of lists), subscribers, list names, and keywords respectively, $R_l \subseteq C \times L \times M$ defines the relation between curators, lists names, and members, and $R_k \subseteq L \times K$ represents keywords appearing in a list name. A list ϕ is defined as $(c, l, M_{c,l})$ where $M_{c,l} = \{m \in M | (c, l, m) \in R_l\}$. A subscription to a list can be represented then by $(s, c, l, M_{c,l})$. To represent keywords we use the following vectors:

- For the use of a keyword k according to curators we define $k_{curator}$ as a vector in $\mathfrak{R}^{|C|}$ where entries in the vector $w_c = |\{(c, l, M_{c,l}) | (l, k) \in R_k\}|$ correspond to the number of lists created by the curator c that contain the keyword k .

- For the use of a keyword k according to members we use a vector k_{member} in $\mathfrak{R}^{|M|}$ where entries in the vector $w_m = |\{(c, l, m) \in R_l | (l, k) \in R_k\}|$ correspond to the number of lists containing the keyword k under which the member m has been listed.

- For the use of a keyword k according to subscribers we utilize a vector $k_{subscriber}$ in $\mathfrak{R}^{|S|}$ where entries in the vector $w_s = |\{(s, c, l, M_{c,l}) | (l, k) \in R_k\}|$ correspond to the number of times that s has subscribed to a list containing the keyword k .

In the vector space model we can measure the similarity between keywords calculating the cosine of the angle for the corresponding vectors in the same dimension. For two vectors k_i and k_j the similarity is $sim(k_i, k_j) = \frac{k_i \cdot k_j}{\|k_i\| \cdot \|k_j\|}$.

We also use Latent Dirichlet Allocation (LDA) [5] to obtain similar keywords. LDA is an unsupervised technique where documents are represented by a set of topics and each topic consists of a group of words. LDA topic model is an improvement over *bag of words* approaches including the vector space model, since LDA does not require documents to share words to be judged similar. As long as they share similar words (that appear together with same words in other documents) they will be judged similar. Thus documents are viewed as a mixture of probabilistic topics that are represented as a T dimensional random

variable θ . For each document, the topic distribution θ has a Dirichlet prior $p(\theta|\alpha) \sim Dir(\alpha)$. In generative story, each document is generated by first picking a topic distribution θ from the Dirichlet prior and then use each document's topic distribution to sample latent topic variables z_i . LDA makes the assumption that each word is generated from one topic where z_i is a latent variable indicating the hidden topic assignment for word w_i . The probability of choosing a word w_i under topic z_i , $p(w_i|z_i, \beta)$, depends on different documents.

We use the bag of words model to represent documents as input for LDA. For our study keywords are documents and words are the different users according to their role in the list structure. To represent keywords we use the following sets:

- For a keyword k according to curators we use the set $k_{bagCurator} = \{c \in C | (c, l, m) \in R_l \wedge (l, k) \in R_k\}$ representing the curators that have created a list containing the keyword k .

- For a keyword k according to members we use a set $k_{bagMember} = \{m \in M | (c, l, m) \in R_l \wedge (l, k) \in R_k\}$ corresponding to the users who have been classified under lists containing the keyword k .

- For a keyword k according to subscribers we use a set $k_{bagSubscriber} = \{s \in S | (s, c, l, M_{c,l}) \wedge (l, k) \in R_k\}$, that is the set of users that follow a list containing the keyword k .

LDA is then executed for all the keywords in the same representation schema (*i.e.*, based on curators, members, or subscribers) generating a topic distribution θ for each document. We can compute similarity between two keywords k_i and k_j in the same representation schema by measuring the angle cosine of their corresponding topic distributions θ_i and θ_j .

3 Characterising Relations between Keywords

We investigate the relevance of the relations between keywords obtained from twitter lists using state of the art similarity measures based on WordNet. In addition, given the limited scope of WordNet we complement our study using knowledge bases published as linked data.

3.1 Similarity Measures Based on WordNet

To validate the relations found from keyword co-occurrence analysis in Twitter lists, we use similarity measures that tap into WordNet [10]. WordNet is a lexical database where synonyms are grouped on synsets, with each synset expressing a concept. Synsets are linked according to semantic relations that depend on the synsets part-of-speech category. Nouns and verbs are arranged in a hierarchy defined by a super-subordinate relation (is-a) known as hyperonymy. In addition, there are meronymy relations (part-of) for nouns, troponym relations (specific way of) for verbs, antonym relations for adjectives, and synonym relations for adverbs. WordNet consists of four sub nets, one for each part of speech category.

A natural measure of similarity between words is the length of the path connecting the corresponding synsets [22,16]. The shorter the path the higher the similarity. This length is usually calculated in the noun and verb is-a hierarchy according to the number of synsets in the path connecting the two words. In the case of two synonyms, both words belong to the same synset and thus the path length is 1. A path length of 2 indicates an is-a relation. For a path length of 3 there are two possibilities: (i) both words are under the same hypernym known as *common subsumer*, and therefore the words are siblings, and (ii) both words are connected through an in-between synset defining an indirect is-a relation. Starting with 4 the interpretation of the path length is harder.

However, the weakness of using path length as a similarity measure in WordNet is that it does not take into account the level of specificity of synsets in the hierarchy. For instance, *measure* and *communication* have a path length of 3 and share *abstraction* as a common subsumer. Despite low path length, this relation may not correspond to the human concept of similarity due to the high level of abstraction of the concepts involved.

Abstract synsets appear in the top of the hierarchy, while more specific ones are placed at the bottom. Thus, Wu and Palmer [26] propose a similarity measure which includes the depth of the synsets and of the least common subsumer (see equation 1). The least common subsumer *lcs* is the deepest hypernym that subsumes both synsets, and depth is the length of the path from the root to the synset. This similarity range between 0 and 1, the larger the value the greater the similarity between the terms. For terms *measure* and *communication*, both synsets have depth 4, and the depth of the *lcs abstraction* is 3; therefore, their similarity is 0.75.

$$wp(\text{synset}_1, \text{synset}_2) = 2 * \text{depth}(\text{lcs}) / (\text{depth}(\text{synset}_1) + \text{depth}(\text{synset}_2)) \quad (1)$$

Jiang and Conrath [16] propose a distance measure that combines hierarchical and distributional information. Their formula includes features such as local network density (*i.e.*, children per synset), synset depth, weight according to the link type, and information content *IC* of synsets and of the least common subsumer. The information content of a synset is calculated as the inverse log of its probability of occurrence in the WordNet hierarchy. This probability is based on the frequency of words subsumed by the synset. As the probability of a synset increases, its information content decreases. Jiang and Conrath distance can be computed using equation 2 when only the information content is used. A shorter distance means a stronger semantic relation. The *IC* of *measure* and *communication* is 2.95 and 3.07 respectively while *abstraction* has a *IC* of 0.78, thus their semantic distance is 4.46.

$$jc(\text{synset}_1, \text{synset}_2) = IC(\text{synset}_1) + IC(\text{synset}_2) - 2 * IC(\text{lcs}) \quad (2)$$

We use, in section 4, the path length, Wu and Palmer similarity, and Jiang and Conrath distance to study the semantics of the relations extracted from Twitter lists using the vector space model and LDA.

3.2 Linked Data to Identify Relation Types

WordNet-based analysis is rather limited, since WordNet contains a small number of relations between synsets. To overcome this limitation and improve the detection of relationships, we use general purpose knowledge bases such as DBpedia [4], OpenCyc [2] and UMBEL [3], which provide a wealth of well-defined relations between concepts and instances. DBpedia contains knowledge from Wikipedia for close to 3.5 million resources and more than 600 relations. OpenCyc is a general purpose knowledge base with nearly 500K concepts around 15K types of relations. UMBEL is an ontology with 28,000 concepts and 38 relations. These knowledge bases are published as linked data [3] in RDF and with links between them: DBpedia resources, and classes are connected to OpenCyc concepts using *owl:sameAs*, and to UMBEL concepts using *umbel#correspondsTo*.

Our aim is to bind keywords extracted from list names to semantic resources in these knowledge bases so that we can identify which kind of relations appear between them. To do so we harness the high degree of interconnection in the linked data cloud offered by DBpedia. We first ground keywords to DBpedia [12], and then we browse the linked data set for relations connecting the keywords.

After connecting keywords to DBpedia resources we query the linked data set to search for relations between pairs of resources. We use a similar approach to [14] where SPARQL queries are used to search for relations linking two resources r_s and r_t . We define the path length L as the number of objects found in the path linking r_s with r_t . For $L = 2$ we look for a *relation_i* linking r_s with r_t . As we do not know the direction of *relation_i*, we search in both directions: 1) r_s *relation_i* r_t , and 2) r_t *relation_i* r_s . For $L = 3$ we look for a path containing two relationships and an intermediate resource *node* such as: r_s *relation_i* *node*, and *node* *relation_j* r_t . Note that each relationship may have two directions and hence the number of possible paths is $2^2 = 4$. For $L = 4$ we have three relationship placeholders and the number of possible paths is $2^3 = 8$. In general, for a path length L we have $n = \sum_{l=2}^L 2^{(l-1)}$ possible paths that can be traversed by issuing the same number of SPARQL queries [4] on the linked data set.

For instance, let us find the relation between the keywords *Anthropology* and *Sociology*. First both keywords are grounded to the respective DBpedia resources, in this case *dbpr:Anthropology* and *dbpr:Sociology*. Figure 2 shows linked data relating these DBpedia resources. To retrieve this information, we pose the query shown in Listing 1.1 [5]. The result is the triples making up the path between

² OpenCyc home page: <http://sw.opencyc.org/>

³ UMBEL home page: <http://www.umbel.org/>

⁴ Note that for large L values the queries can last long time in large data sets.

⁵ Property paths, in SPARQL 1.1 specification, allow simplifying these queries.

the resources. In our case we discard the initial *owl:sameAs* relation between DBpedia and OpenCyc resources, and keep the assertion that Anthropology and Sociology are Social Sciences.

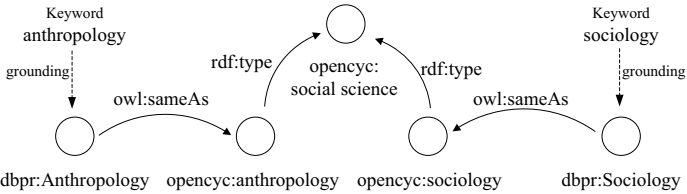


Fig. 2. Linked data showing the relation between the anthropology and sociology

```
SELECT *
WHERE{<dbpr:Anthropology> ?relation1 ?node1. ?node1 ?relation2 ?node2.
      <dbpr:Sociology> ?relation4 ?node3. ?node3 ?relation3 ?node2.}
```

Listing 1.1. SPARQL query for finding relations between two DBpedia resources

4 Experiment Description

Data Set: Twitter offers an Application Programming Interface (API) for data collection. We collected a snowball sample of users and lists as follows. Starting with two initial seed users, we collected all the lists they subscribed to or are members of. There were 260 such lists. Next, we expanded the user layer based on current lists by collecting all other users who are members of or subscribers to these lists. This yielded an additional set of 2573 users. In the next iteration, we expanded the list layers by collecting all lists that these users subscribe to or are members of. In the last step, we collected 297,521 lists under which 2,171,140 users were classified. The lists were created by 215,599 distinct curators, and 616,662 users subscribe to them⁶. From list names we extracted, by approximate matching of the names with dictionary entries, 5932 unique keywords; 55% of them were found in WordNet. The dictionary was created from article titles and redirection pages in Wikipedia.

Obtaining Relations from Lists: For each keyword we created the vectors and the bags of words for each of the three user-based representations defined in section 2. We calculated cosine similarity in the corresponding user-based vector space. We also run the LDA algorithm over the bags of words and calculated the cosine similarity between the topic distribution produced for each document. We kept the 5 most similar terms for each keyword according to the Vector-space and LDA-based similarities.

⁶ The data set can be found here: <http://goo.gl/vCYyD>

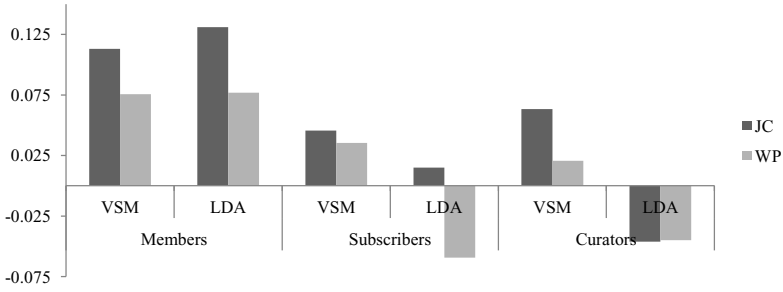


Fig. 3. Coefficient of correlation between Vector-space and LDA similarity with respect to WordNet measures

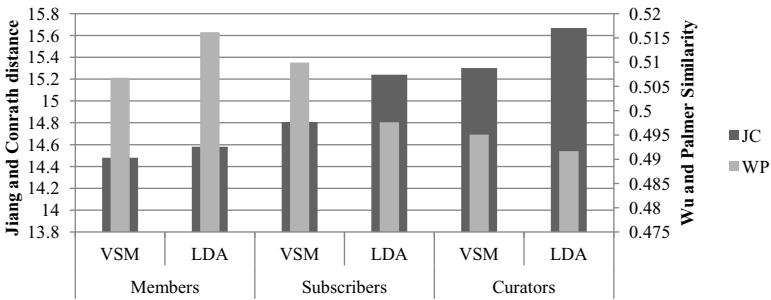


Fig. 4. Average Jiang and Conrath distance and Wu and Palmer similarity

WordNet Analysis: For each pair of similar keywords we calculated their similarity according to Jiang and Conrath (JC) and Wu and Palmer (WP) formulas. To gain an initial insight about these measures we calculate the correlation between them (see Figure 3). We use the Pearson’s coefficient of correlations which divides the covariance of the two variables by the product of their standard deviations.

In general these results show that Vector-space and LDA similarity based on members produce the most similar results to that of WordNet measures. Vector-space similarity based on subscribers and curators also produces correlated results, although significantly lower. LDA similarity based on subscribers results is correlated to JC distance but not to WP similarity. Finally LDA based on curators produces results that are not correlated to WordNet similarities.

Correlation results can be partially explained by measuring the average of JC distance and WP similarity⁷ (see figure 4). Vector-space and LDA similarities based on Members have the shortest JC distance, and two of the top tree WP similarity values. Vector-space similarity based on subscribers has also a short JC distance, and a high WP similarity. For the rest of similarities JC distances are longer and WP similarity lower.

⁷ The averages were calculated over relations for which both terms were in WordNet.

To identify the type of relations found by Vector-space and LDA similarities we calculate, as shown in table 2 the path length of the corresponding relations in WordNet. To guarantee a base similarity, we use a threshold of 0.1; similarities under this value were discarded. Note that in WordNet different part of speech categories have distinct hierarchies and hence the path length can be calculated only for terms in the same category. According to the path length, the similarity based on members produce the highest number of synonyms (path length=1), reaching a 10.87% of the relations found in WordNet for the case of LDA similarity. In this case, the LDA model analyzes co-occurrence of groups of members across different keywords to identify related keywords. Unlike the vector space model, which requires exact members to be present in similar keywords, LDA allows synonyms, i.e., different members that tend to co-occur with the same sets of keywords, to contribute to keyword similarity.

Table 2. Path length in WordNet for similar Keywords according to Vector-space and LDA models

Path Length	Members		Subscribers		Curators	
	VSM	LDA	VSM	LDA	VSM	LDA
1	8.58%	10.87%	3.97%	3.24%	1.24%	0.50%
2	3.42%	3.08%	1.93%	0.47%	0.70%	0.00%
3	2.37%	3.77%	2.96%	2.06%	2.38%	4.03%
>3	67.61%	65.50%	67.27%	67.56%	77.83%	75.81%

Similarity based on subscribers and curators produce a significative lower number of synonyms. Likewise, similarity based on members produces the highest number of direct is-a relations (path length=2). LDA similarity based on curators produce the highest number of keywords directly related by a common superclass or an indirect is-a relation (path length=3).

Given that the majority of relations found in WordNet have a path length greater than or equal to 3, we decided to categorize them according to whether the relation is based on a common subsumer or whether it is based on linked is-a relations. In average 97.65% of the relations with a path length ≥ 3 involve a common subsumer.

As it was argued before, the depth of the least common subsumer influences the relevance of a relation. A manual inspection of the WordNet hierarchy shows that synsets being at a distance greater than or equal to 5 from the root may be considered as more specific. Figure 5 shows the percentage of relations according to the depth of the least common subsumer in the WordNet hierarchy. For a depth of the LCS greater than or equal to 5 and to 6 the Vector-space similarity based on subscribers produces the highest percentage of relations (39.19% and 20.62% for each case) followed by the Vector-space similarity based on members (37.07% and 17.96%). Starting from a depth of the LCS greater than or equal to 7 until 9 the LDA and Vector-space similarity based on members gathers the highest percentage of relations.

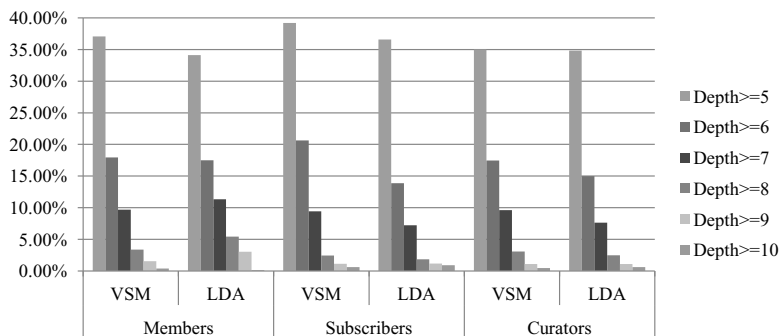


Fig. 5. Relations according to the depth of the least common subsumer LCS

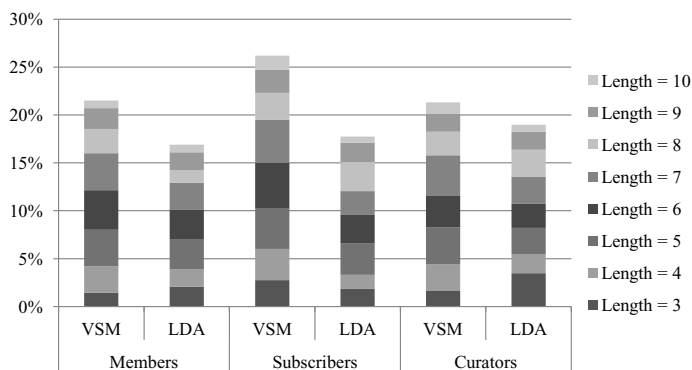


Fig. 6. Relations according to the path length for those cases where the least common subsumer has depth greater or equal to 5

In addition to the depth of the LCS, the other variable to explore is the length of the path setting up the relation. The stacked columns in figure 6 show the cumulative percentage of relations found by Vector-space and LDA models according to the path length of the relation in WordNet, with a depth of the least common subsumer greater than or equal to 5. From the chart we can state that Vector-space similarity based on subscribers produces the highest percentage of relations (26.19%) with a path length ≤ 10 . This measure also produces the highest percentage of relations for path lengths ranging from 9 to 4. The Vector-space similarity based on members produces the second highest percentage of relations for path lengths from 10 to 6.

In summary, we have shown that similarity models based on members produce the results that are most directly related to the results of similarity measures based on WordNet. These models find more synonyms and direct relations is-a when compared to the models based on subscribers and curators. These results suggest that some users are classified under different lists named with synonyms or with keywords representing a concept in a distinct level of specificity. We also

discovered that the majority of relations found by any model have a path length ≥ 3 and involve a common subsumer. Vector-space model based on subscribers produces the highest number of relations that can be considered specific (depth of LCS ≥ 5 or 6). However, for more specific relations ($7 \leq \text{depth of LCS} \leq 9$) similarity models based on members produce a higher number. In addition we considered the path length, for those relations containing a LCS placed in a depth ≥ 5 in the hierarchy, as a variable influencing the relevance of a relation. Vector-space model based on subscriber finds the highest number of relations with $4 \leq \text{length} \leq 10$. In general similarity models based on curators produce a lower number of relations. We think this may be due to the scarcity of lists per curator. In our dataset each curator has created 1.38 lists in average.

Linked Data Analysis: Our approach found DBpedia resources for 63.77% of the keywords extracted from Twitter Lists. In average for the 41.74% of the relations we found the related keywords in DBpedia. For each relation found by Vector-space or LDA similarity we query the linked data set looking for patterns between the related keywords. Figure 7 shows the results according to the path length of the relations found in the linked data set. These results are similar to the ones produced by WordNet similarity measures. That is, similarity based on Members produce the highest number of synonyms and direct relations though in this case Vector-space similarity produces more synonyms than LDA. Vector-space similarity based on subscribers has the highest number of relations of length 3, followed by Vector-space and LDA similarity based on members.

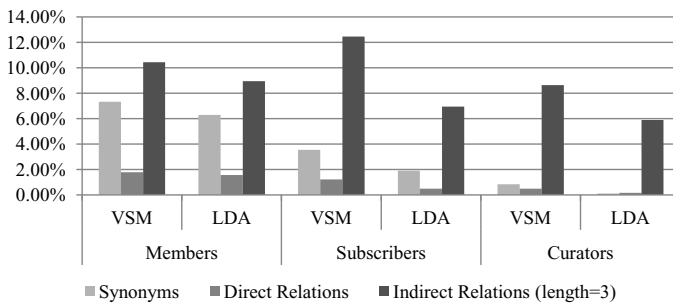


Fig. 7. Relations identified from linked data queries

Given that the Vector-space model based on members found the majority of direct relations, we present, in table 3, the relations identified in the linked data set. *Broad term* and *subClassOf* are among the most frequent relations. This means that members of lists are usually classified in lists named with keywords representing a concept with a different level of specificity. Other relations that are difficult to elicit from traditional lexicons are also obtained, such as *developer*, *genre* or *largest city*.

Table 3. Direct relations established by the Vector-space model based on members

Relation type	Example of keywords		
Broader Term	26%	life-science	biotech
subClassOf	26%	authors	writers
developer	11%	google	google_apps
genre	11%	funland	comedy
largest city	6%	houston	texas

Table 4. Indirect relations of length 3 found in the linked data set for the relations established by the Vector-space model based on subscribers

		$r_s \xrightarrow{relation_1} object$	$object \xleftarrow{relation_2} r_t$		
Relations		Example			
type	type	67.35%	nokia	→ company	← intel
subClassOf	subClassOf	30.61%	philanthropy	→ activities	← fundraising

		$r_s \xleftarrow{relation_1} object$	$object \xrightarrow{relation_2} r_t$		
Relations		Example			
genre	genre	12.43%	theater	← Aesthetica	→ film
genre	occupation	10.27%	fiction	← Adam Maxwell	→ writer
occupation	occupation	8.11%	poet	← Alina Tugend	→ writer
product	product	7.57%	clothes	← ChenOne	→ fashion
product	industry	9.73%	blogs	← UserLand Software	→ internet
occupation	known for	5.41%	author	← Adeline Yen Mah	→ writing
known for	known for	3.78%	skeptics	← Rebecca Watson	→ atheist
main interest	main interest	3.24%	politics	← Aristotle	→ government

In addition we also investigate the type of relations of length 3 elicited using the Vector-space model based on subscribers. The most common patterns found in the linked data set were $r_s \xrightarrow{relation_1} object \xleftarrow{relation_2} r_t$, and $r_s \xleftarrow{relation_1} object \xrightarrow{relation_2} r_t$ with 54.73% and 43.49% of the relations respectively. Table 4 shows the obtained relations according to each pattern.

With respect to the first pattern, 97.96% of the related keywords can be considered siblings since they are associated via *typeOf* or *subClassOf* relations with a common class. That is, some subscribers follow lists that share a common super concept. On the other hand, the second pattern shows a wider range of relations. Keywords are related since they are *genres*, *occupations*, *products*, *industries*, or *main interest* that appear together in the description of an individual in the linked data set.

5 Related Work

Twitter has been investigated from different perspectives including network characteristics, user behaviors, and tweet semantics among others. Twitter network

properties, geographical features, and users have been studied in [15,17]. In [15] authors use the HITS algorithm to identify hubs and authorities from the network structure, while in [17] authors categorise users according to their behaviors. To identify the tweet semantics some proposals [2,1,23,6] annotate them with semantic entities using available services such as Zemanta, Open Calais, and DBpedia Spotlight [21]. In [2] tweets are linked to news articles and are enriched with semantic annotations to create user profiles. These semantic annotations of tweets have been used in a faceted search approach [1]. In [23] tweets and their semantic annotations are represented according to existing vocabularies such as FOAF, Dublin Core, and SIOC, and are used to map tweets to websites of conferences and events. In [6] authors use the semantic entities identified in Tweets to obtain the concepts associated with user profiles. In addition some classifiers have been proposed in [8] to extract players and events from sport tweets. Twitter allows the use of hashtags as a way to keep conversation around certain topics. In [18] authors have studied hashtags as candidate identifiers of concepts.

With respect to Twitter Lists, they have been used to distinguish elite users, such as celebrities, media, organizations, and bloggers [25]. In this work authors provide an analysis on the information flow of Twitter, and show dueling importance of mass media and opinion leaders. In addition, in [9] lists have been used as a source for discovering latent characteristics of users.

In the broader context of the Web 2.0 the emerging semantics of folksonomies have been studied under the assumption that it is possible to obtain a vocabulary from these classification systems. In folksonomies the set of tags around resources tends to converge [13] and users in the same social groups are more likely to use the same set of tags [20]. The semantics of the emerging relations between tags have been studied in [7,19]. A survey of the state of the art on this matter can be found in [11].

6 Conclusions

In this paper we have described different models to elicit semantic relations from Twitter lists. These models represent keyword co-occurrence in lists based on three user roles: curators, subscribers and members. We measure similarity between keywords using the vector-space model and a topic based model known as LDA. Then we use Wordnet similarity measures including Wu and Palmer, and Jiang and Conrath distance, to compare the results of the vector-space and LDA models.

Results show that applying vector-space and LDA metrics based on members produce the most correlated results to those of WordNet-based metrics. We found that these measures produce relations with the shortest Jiang and Conrath distance and high Wu and Palmer similarities. In addition, we categorize the relations found by each model according to the path length in WordNet. Models based on members produce the highest number of synonyms and of direct is-a relations. However, most of the relations have a path length ≥ 3 and have a common subsumer. We analyze these relations using the depth of the LCS

and the path length as variables that help to identify the relevance of relations. This analysis shows that the vector-space model based on subscribers finds the highest number of relations when relevance is defined by a depth of $LCS \geq 5$, and the path length of relations is between 10 and 4.

We also investigate the type of relations found by each of the models using general knowledge bases published as linked data. We categorize the relations elicited by each model according to the path length in the linked data set. These results confirm that the models based on members produce the highest number of synonyms and direct relations. In addition, we find that direct relations obtained from models based on members are mostly *Broader Term* and *subclassOf*. Finally, we study the type of relations obtained from the vector-space model based on subscribers with a path length of 3 and find that mostly they represent sibling keywords sharing a common class, and subjects that are related through an individual.

Acknowledgments. This work is supported by the project myBigData, and the FPI grant (BES-2008-007622) of the Spanish Ministry of Science and Innovation.

References

1. Abel, F., Celik, I., Houben, G.-J., Siehndel, P.: Leveraging the Semantics of Tweets for Adaptive Faceted Search on Twitter. 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. 1–17. Springer, Heidelberg (2011)
2. Abel, F., Gao, Q., Houben, G.-J., Tao, K.: Semantic Enrichment of Twitter Posts for User Profile Construction on the Social Web. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011. LNCS, vol. 6644, Part II, pp. 375–389. Springer, Heidelberg (2011)
3. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems, IJISWIS (2009)
4. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. Journal of Web Semantic 7(3), 154–165 (2009)
5. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
6. Cano, A.E., Tucker, S., Ciravegna, F.: Follow me: Capturing entity-based semantics emerging from personal awareness streams. In: Making Sense of Microposts (#MSM 2011), pp. 33–44 (2011)
7. Cattuto, C., Benz, D., Hotho, A., Stumme, G.: Semantic Grounding of Tag Relatedness in Social Bookmarking Systems. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 615–631. Springer, Heidelberg (2008)
8. Choudhury, S., Breslin, J.: Extracting semantic entities and events from sports tweets. In: Proceedings of the ESWC 2011 Workshop on 'Making Sense of Microposts'. CEUR Workshop Proceedings, vol. 718 (May 2011)
9. Dongwoo Kim, Y.J.: Analysis of Twitter lists as a potential source for discovering latent characteristics of users. In: Workshop on Microblogging at the ACM Conference on Human Factors in Computer Systems (CHI 2010), Atlanta, CA, USA (2010)

10. Fellbaum, C.: *WordNet and wordnets*, 2nd edn., pp. 665–670. Elsevier, Oxford (2005)
11. García-Silva, A., Corcho, O., Alani, H., Gómez-Pérez, A.: Review of the state of the art: discovering and associating semantics to tags in folksonomies. *The Knowledge Engineering Review* 27(01), 57–85 (2012)
12. García-Silva, A., Szomszor, M., Alani, H., Corcho, O.: Preliminary results in tag disambiguation using dbpedia. In: *Knowledge Capture (K-Cap 2009)-Workshop on Collective Knowledge Capturing and Representation-CKCaR (2009)*
13. Golder, S.A., Huberman, B.A.: Usage patterns of collaborative tagging systems. *Journal of Information Science* 32(2), 198–208 (2006)
14. Heim, P., Lohmann, S., Stegemann, T.: Interactive Relationship Discovery via the Semantic Web. 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. 303–317. Springer, Heidelberg (2010)
15. Java, A., Song, X., Finin, T., Tseng, B.: Why We Twitter: An Analysis of a Microblogging Community. In: Zhang, H., Spiliopoulou, M., Mobasher, B., Giles, C.L., McCallum, A., Nasraoui, O., Srivastava, J., Yen, J. (eds.) *WebKDD 2007. LNCS*, vol. 5439, pp. 118–138. Springer, Heidelberg (2009)
16. Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy. *CoRR*, cmp-lg/9709008 (1997)
17. Krishnamurthy, B., Gill, P., Arlitt, M.: A few chirps about twitter. In: *Proceedings of the First Workshop on Online Social Networks, WOSN 2008*, pp. 19–24. ACM, New York (2008)
18. Laniado, D., Mika, P.: Making Sense of Twitter. 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. 470–485. Springer, Heidelberg (2010)
19. Markines, B., Cattuto, C., Menczer, F., Benz, D., Hotho, A., Gerd, S.: Evaluating similarity measures for emergent semantics of social tagging. In: *Proceedings of the 18th International Conference on World Wide Web, WWW 2009*, pp. 641–650. ACM, New York (2009)
20. Marlow, C., Naaman, M., Boyd, D., Davis, M.: Ht06, tagging paper, taxonomy, flickr, academic article, to read. In: *Proceedings of the Seventeenth Conference on Hypertext and Hypermedia*, pp. 31–40. ACM Press (2006)
21. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia spotlight: Shedding light on the web of documents. In: *Proceedings of the 7th International Conference on Semantic Systems, I-Semantics (2011)*
22. Pedersen, T., Patwardhan, S., Michelizzi, J.: Wordnet: Similarity - measuring the relatedness of concepts. In: *AAAI*, pp. 1024–1025. AAAI Press / The MIT Press (2004)
23. Rowe, M., Stankovic, M.: Mapping tweets to conference talks: A goldmine for semantics. In: *Social Data on the Web Workshop, International Semantic Web Conference (2010)*
24. Salton, G., McGill, M.J.: *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York (1986)
25. Wu, S., Hofman, J.M., Mason, W.A., Watts, D.J.: Who says what to whom on twitter. In: *Proceedings of the 20th International Conference on World Wide Web, WWW 2011*, pp. 705–714. ACM, New York (2011)
26. Wu, Z., Palmer, M.: Verb semantics and lexical selection. In: *Proc. of the 32nd Annual Meeting on Association for Computational Linguistics*, pp. 133–138 (1994)

Crowdsourcing Taxonomies*

Dimitris Karampinas and Peter Triantafillou

Computer Engineering & Informatics Department, University of Patras
{karampin,peter}@ceid.upatras.gr

Abstract. Taxonomies are great for organizing and searching web content. As such, many popular classes of web applications, utilize them. However, their manual generation and maintenance by experts is a time-costly procedure, resulting in static taxonomies. On the other hand, mining and statistical approaches may produce low quality taxonomies. We thus propose a drastically new approach, based on the proven, increased human involvement and desire to tag/annotate web content. We define the required input from humans in the form of explicit structural, e.g., supertype-subtype relationships between concepts. Hence we harvest, via common annotation practices, the collective wisdom of users with respect to the (categorization of) web content they share and access. We further define the principles upon which crowdsourced taxonomy construction algorithms should be based. The resulting problem is NP-Hard. We thus provide and analyze heuristic algorithms that aggregate human input and resolve conflicts. We evaluate our approach with synthetic and real-world crowdsourcing experiments and on a real-world taxonomy.

Keywords: Collective Intelligence, Crowdsourcing, Taxonomy, Tagging.

1 Introduction

Social media applications and research are increasingly receiving greater attention. A key defining characteristic is the increased human involvement. Even before today's success of social media applications, many applications became extremely successful due to the clever exploitation of implicit human inputs (e.g., Google's ranking function), or explicit human input (e.g., Linux open source contributions). Social media and the social web have taken this to the next level. Humans contribute content and share, annotate, tag, rank, and evaluate content. Specialized software aggregates such human input for various applications (from content searching engines to recommendation systems, etc). The next wave in this thread comes from *crowdsourcing systems* in which key tasks are performed by humans (either in isolation or in conjunction with automata) [7]. Lately, within the realm of data and information retrieval systems, crowdsourcing is gaining momentum as a means to improve system performance/quality [3,13]. A

* This work was partially funded by the EIKOS research project, within the THALES framework, administered by the Greek Ministry for Education, Life Long Learning, and Religious Affairs.

recent contribution suggests to engage humans during the processing of queries for which humans are better suited for the task (e.g., entity disambiguation) [8]. Further, (anthropocentric) data systems are proposed whose functioning (including semantics enrichment and related indices) depends on the users' contributions and their collective intelligence [21].

The central idea is to respect and exploit the fact that for some tasks humans can provide excellent help. The challenge then rests on our ability to harness and properly aggregate individual input to derive the community wisdom and exploit it to solve the problem at hand. One particularly interesting problem is that of constructing taxonomies. Taxonomies provide great help for structuring and categorizing our data sets. As such, currently they are at the heart of many web applications: Products are available that exploit taxonomic knowledge in order to improve results in product search applications [1,2]. In local search, location taxonomies are used to improve results by utilizing a querying user's location and the known location of search result items to improve result quality. Google's search news, localized results, and product categories is a prime example for these. As another example, domain-specific (a.k.a. vertical) search engines utilize taxonomies (topic hierarchies) which are browsable and searchable using complex queries and receiving ranked results.

Our work rests on the realization that social media users (contributing and annotating content) have a good understanding of the fundamental (subtype-supertype) relationships needed to define a taxonomy for their content. For instance, biologists collectively can help define the taxonomy to be used, for example in a vertical (biological) search engine. Users from different locations can collectively define a locality taxonomy used in search engines with localization services. Traders in e-shops can easily collectively come up with product categorizations. These examples show that **humans can** offer great help! Further, the vast success enjoyed by a great number of social web applications, prove that **humans are willing** to provide such annotations. Hence, the human's willingness and ability can help solve a problem that is close to the heart of the semantic web community and which is addressed here: Crowdsourcing taxonomies inherently promise to provide a way to come up with high-quality taxonomies, based on the collective knowledge of its users, which will be dynamic and reflect the user-community's understanding of the data space.

2 Problem Statement, Rationale, and Challenges

Our model does not depend on any "experts". We adopt an automated approach, with the additional feature that users explicitly provide us with relations between the keywords (so-called "tags") they employ to annotate the content they share. Humans have a good understanding of the supertype-subtype relations between various thematic categories, since these naturally exist around them. So, we aim to exploit **extended tagging** and a categorization capability in order to develop high-quality taxonomies. We ask users to contribute with metadata in this format: $tag_a \rightarrow tag_d$. Here, tag_a is a supertype topic and represents a higher

node to a potential concept hierarchy whilst tag_d is a subtype topic. The arrow between them connotes an *Ancestor* \rightarrow *Descendant* ($A \rightarrow D$) relation. In figure

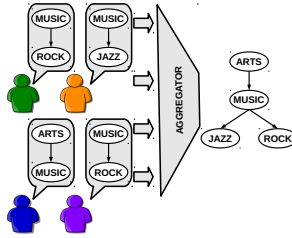


Fig. 1. Crowdsourcing Taxonomy Example

□ we demonstrate the basic idea of our effort. A community of users, forming a crowdsourcing environment, provides the system with tag relations. These can either be different between each other or depict the same relationship (i.e. Music \rightarrow Rock). We refer to these tag relations also with the term “votes”, since they incorporate users’ personal opinions for parts of the taxonomy tree. Our goal is to aggregate all given tag relations (votes) and produce a taxonomy that is derived using our community’s knowledge/wisdom.

The problem is not easy! The following challenges arise:

- **Individuals are prone to errors.** Sometimes they specify incorrect tag relations, e.g., because of constrained knowledge and highly complex datasets. When building large scale taxonomies, the granularity level between nodes in adjacent levels is “fine” (especially at the lower levels of the taxonomy) and thus the frequency of such “false votes” may be high. So, contradicting opinions arise. But, this happens not only directly, but also indirectly, as a result of a combination of various relationships. Our goal is to resolve such conflicts. Since we depend on the crowd’s wisdom, a natural discourse is to have the majority opinion prevail.
- **Incomplete (structural) information.** Users’ knowledge might not be wide enough to completely cover an ideal, “golden rule” taxonomy (e.g. as constructed by experts). For example, in figure □, suppose that a vote Arts \rightarrow Jazz (which is valid) were given instead of the Arts \rightarrow Music one. In this scenario we have evidence that both Arts and Music nodes are ancestors of node Jazz but we have no insight as far as their relative relationship is concerned. In this case, the system must be able to implicitly utilize the users’ given input and fill in the missing structural information. This filling may be done incorrectly. Our approach will be to make a best guess (according to some metrics) and rely on future incoming votes to correct any such mistakes.
- **Incremental, online taxonomy development.** The problem is set in a dynamic environment. Users provide us with relations in an online way. The objective here is to introduce any newly incoming relations into the current structure in a cumulative manner. For every new vote, we shall be able to modify and alter the current taxonomy state without having to destroy or

build it from scratch. Based on the above example, suppose that we eventually receive an Arts \rightarrow Music vote. If previously we had made a mistake when filling the lack of knowledge between Arts and Music, an efficient rectification must take place based on the now completed information. To sum up, dynamic, piece-wise, online taxonomy maintenance is a key characteristic.

The “human-centric” approach we describe exploits users’ knowledge (which is very difficult to correctly derive by automatic means) and produces taxonomies that are in accordance to the beliefs of the system’s user base. In a sense, instead of constraining user input to characterizing the content (as is typically done in social media environments), we go one step further and allow users to provide with input that will lead to the construction of taxonomized datasets.

2.1 The Model, Solution Invariants, and Assumptions

As mentioned, the shape and structure of the taxonomy emerge from the crowd’s *subjective* will, as it evolves. As the number of participants increases, in general, the higher the output quality becomes. When conflicts arise, several conflicting taxonomy states emerge as alternatives. One of them will be associated with the greatest number of votes. In this way, the new accepted state of the taxonomy will emerge. At the end, this process will converge to a structure, entirely defined from the community’s aggregated knowledge. At this point, we can claim that this final product *objectively* depicts a complete taxonomy. But how do we evaluate such a taxonomy? We wish we could compare it against a golden rule taxonomy and see how they differ; but there is no standardized, “ideal” structure on which everyone agrees. Even if we compare taxonomies created by experts there will not be a 100% match, since both the rules for creating the taxonomy and the input data are often contradictory and obscure.

Users are asked to provide us with *Ancestor* \rightarrow *Descendant* relations but any given vote has its own interpretation, depending of the current state of the taxonomy. Any incoming tag relation will be classified to a category, based on the relative positions of the nodes it touches. For example, in figure [1](#) the following possible scenarios can arise as far as an incoming vote is concerned:

- **Ancestor \rightarrow Descendant (A-D)**: i.e. Arts \rightarrow Jazz. These relations practically increase our confidence for the current state of the taxonomy and generally leave it intact.
- **Descendant \rightarrow Ancestor (D-A)**: i.e. Rock \rightarrow Arts. These votes form what we call **backedges** and create cycles on the current structure. They require special handling and may change the current state of the taxonomy. They are not necessarily “false votes” according to a golden rule taxonomy and they are usefull, since they can restore possible invalid relations which were based on previous erroneous input or assumptions.
- **Crosslinks**: Relations like Jazz \rightarrow Rock, do not belong to any of the former classes. This type of links interconnects nodes that have a common ancestor. If according to the golden rule taxonomy, there is a supertype-subtype

relationship between these nodes, our algorithms for handling this situation will be able to eventually yield the proper tree structure. If, however, there is no such relation between the nodes of a crosslink, then our algorithms will inescapably produce a supertype-subtype relation between the two. When a relation like this arises, special handling is required: our current idea for this involves users supplying “negative” votes when they see incorrect crosslink relationships established in the current taxonomy. We leave discussion of this to future work.

Before we continue with the problem formulation we specify two **solution invariants** our approach maintains and give some insight of the taxonomy building algorithm that follows.

Tree Properties: This is the primary invariant we maintain. A tree is an undirected graph in which any two vertices are connected by exactly one path. There are no cycles and this is a principle we carry on throughout the taxonomy evolution. Starting with many shallow subtrees, as votes enter the system, we detect relations between more and more tags. The independent trees gradually form a forest and we use a common “Global Root” node to join them.

Maximum Votes Satisfiability: We also wish to preserve a *quantitative* characteristic. Our purpose is to utilize every incoming *Ancestor* \rightarrow *Descendant* relation and embed it on the current structure. If this raises conflicts, our solution to this is to derive a taxonomy structure which as a whole satisfies the maximum number of users’ votes. At this point we need to mention that according to our model, there is no constraint to the number of votes a user can submit to the system (see “free-for-all tagging” at [15]). Satisfiability is measured not on per individual basis but over all votes, overall.

Finally, we need to state that our solution does not take any measures to face synonyms or polysemy issues. Although according to [12] these are not major problems for social media, we admit that users tend to annotate their content with idiosyncratic tags which in our case can lead to wrong keyword interpretation and create links that users do not intent to recommend. This issue is orthogonal to our work since we focus on structural development and thus we can assume a controlled vocabulary without loss of generality.

3 Formal Formulation and Analysis

Leaving aside the added complexity due to the online nature of our venture, we show that even an offline approach yields an NP-Hard problem.

Note, that at first thought, one could suggest the following solution to our problem: First, create a directed graph $G(V, E)$ where each vertex $v \in V$ represents a given tag and each $e \in E$ represents a relationship between the two nodes. Edges bear weights w reflecting the number of votes from users for this relation. Intuitively, this calls to mind minimum/maximum spanning tree algorithms. Thus, second, run a “variation” of one of any well-known algorithms to retrieve a Maximum Spanning Tree. Because, however, our graph is directed, what we need here is a ‘maximum weight arborescence’ (which is defined as the

directed counterpart of a maximum spanning tree). However, this simplistic idea has a major flaw. If the graph is not strongly connected (something that regularly happens especially during the early stages of the taxonomy development) then a number of nodes has to be omitted from the final output.

Our problem is formalized as follows:

INPUT: Complete graph $G = (V, E)$, weight $w(e) \in Z_0^+$ for each $e \in E$.

OUTPUT: A spanning tree T for G such that, if $W(\{u, v\})$ denotes the sum of weights of the edges on the path joining u and v in T , then find B where:

$$B = \max\left(\sum_{u, v \in V} W(\{u, v\})\right) \quad (1)$$

This problem is a straightforward instance of the Optimum Communication Spanning Tree problem in [9] and has been proven to be NP-Hard [1]. The key idea here for the matching of the two problems, is that **shortcuts** (edges weighting 0) are permitted. Obviously, as an algorithm proceeds online maintaining the tree invariant, there will be cases when nodes are connected with their edge having zero weights (as users may have not supplied yet any votes for this edge). B represents the maximum number of votes that is satisfied and along with the tree notion meets the standards set by the invariants in the previous section.

4 Crowdsourced Taxonomy Building Algorithm

As noted, our problem is NP-Hard. For n nodes, an optimal solution would require an exhaustive search of all possible n^{n-2} spanning trees and the selection of the one that maximizes value B in (1). So, we adopt a heuristic approach. We relax the second invariant: we demand the maximum number of satisfied votes, not overall, but only between consecutive algorithmic steps. Each vote is embodied into the taxonomy via an algorithm that introduces a series of transformations for every incoming vote (tag pair).

4.1 The Core Algorithm: CrowdTaxonomy

Algorithm 1 is called on every incoming vote. For every vote ($u \rightarrow v$) we first need to identify whether the named nodes are new to the system or if they are already part of the tree. In case both nodes already exist (line: 9) we need to specify their relative position and thus we call a Lowest Common Ancestor (LCA) routine that returns the LCA node w . If $w = null$ (line: 11), there is no common ancestor and nodes u and v belong to different trees. If w coincides with u (line: 14), then u is already an ancestor of v , which is something that strengthens the evidence we have for the current state of the taxonomy. When w equals v (line: 16) the $v \rightarrow u$ relation introduces a conflict and implies that a modification may be needed. If we accept this edge, the structure's constraints are violated since a cycle is created.

¹ We consider the requirements equal to the standard basis vector and refer to the Optimization version.

Algorithm 1. VOTE PROCESSING

Require: A vote $tag_x \rightarrow tag_y$

```

1: Node  $u \leftarrow \text{hash}(tag_x)$ 
2: Node  $v \leftarrow \text{hash}(tag_y)$ 
3: if  $((u = \text{null}) \text{ and } (v = \text{null}))$  then
4:   CREATE NEW TREE
5: else if  $((u \neq \text{null}) \text{ and } (v = \text{null}))$  then
6:   ATTACH NEW CHILD
7: else if  $((u = \text{null}) \text{ and } (v \neq \text{null}))$  then
8:   MERGE
9: else
10:  Node  $w \leftarrow \text{LCA}(u, v)$ 
11:  if  $(w = \text{null})$  then
12:    MERGE
13:  else if  $(w = u)$  then
14:    CREATE FORWARD EDGE
15:  else if  $(w = v)$  then
16:    BACKEDGE CONFLICT RESOLUTION
17:  else
18:    EXPAND VERTICALLY
19:  end if
20: end if

```

Lastly, in case w is a separate node on the tree (line: [18](#)), the new relation forms a crosslink and is handled appropriately. Hereafter, we describe every tree transformation triggered by each of these cases.

TRFSM CREATE NEW TREE: In this simple scenario the taxonomy does not yet include any of the two nodes of the new vote. So the ancestor node u is attached to the global root via a *shortcut* ($R \rightarrow u$) and node v plays the role of its child (see figure [2a](#)).

Definition 1 Shortcut: An “artificial” Parent \rightarrow Child link that is not an explicit user supplied vote,. Its weight is 0, and it is utilized to preserve structural continuation.

This addition forms a new tree with only two nodes. In the future, it will be expanded with more nodes or get merged to another expanding tree.

TRFSM ATTACH NEW CHILD: This is another straightforward case. Node u pre-exists and the incoming relation can be easily assimilated by adding an extra child to it.

TRFSM MERGE: MERGE is used in two similar cases. In line [8](#) of the core algorithm we ask to attach a new node u to our taxonomy but in a generic scenario its descendant node v does already have a parent node. So does happen

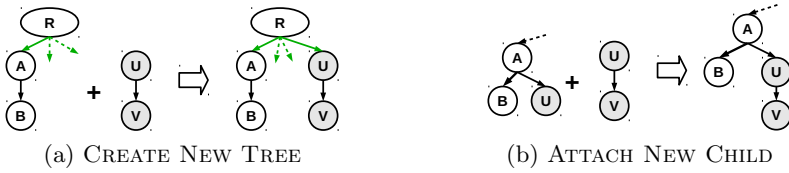


Fig. 2. Creating a new tree or attaching a new child

in line 12 where we need to annex u 's participating tree to that one of v 's with a link between them. In figure 3a we observe that both node C and u “compete for the paternity” of v . In order to maintain the tree properties, only one of the potential parents can be directly connected with v . We arbitrarily choose node C to be the direct ancestor (parent) of v and set node u to be parent of C which is in accordance with the *Maximum Vote Satisfiability* invariant since an *Ancestor* \rightarrow *Descendant* ($u \rightarrow v$) relation takes place. The (temporary) state formed in the middle of figure 3a suffers from the same “paternity conflict” problem - now between A and u over C . Following the same reasoning, we finally place node u on top of v 's tree being now the parent of v 's root. Since there is no given relation yet between u and A we form a shortcut between them. We also maintain a *forward edge* from u to v so not to lose the information we have regarding the vote for the $u \rightarrow v$ relationship.

Definition 2 Forward Edge: A latent relation between two nodes. The source node is an ancestor in the taxonomy and the target is a descendant. Forward edges do not refer to Parent \rightarrow Child links and remain hidden since they violate tree's properties.

The idea behind this transformation is that since we don't have enough evidence to decide on the partial order of v 's ancestors we temporarily send u node to the root. Relations that will follow will illustrate the correct order.

If v is a root of a tree, an ATTACH NEW CHILD transformation is called.

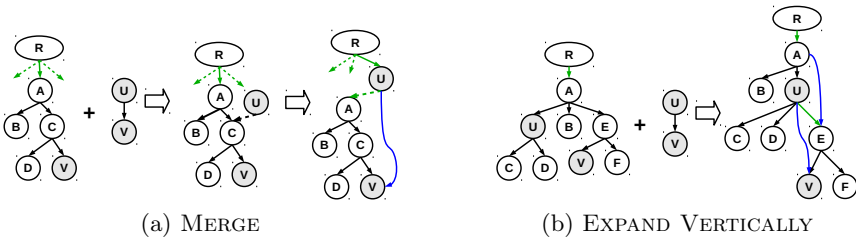


Fig. 3. The MERGE and EXPAND VERTICALLY transformations

TRSFM EXPAND VERTICALLY: In this case the newly incoming vote is interpreted as a crosslink according to the current state of the taxonomy. As shown in figure 3B the logic we follow resembles at some point the MERGE transformation. First, we locate the common ancestor A of u and v and break the link between it and the immediate root E of the subtree that includes v . The independent subtree is now linked with u via a shortcut formed between u and its root E . Two forward edges are spawned to indicate latent *Ancestor* \rightarrow *Descendant* relations. For the sake of completeness we report here that in contrast to *common* edges between nodes, shortcuts are not converted to forward edges when the link that touches the two nodes brakes.

TRSFM CREATE FORWARD EDGE: Straightforward scenario. A forward edge is added from u to v unless their node distance is 1 (*Parent* \rightarrow *Child*). In any other case (distance = 1) no operation takes place.

TRSFM BACKEDGE CONFLICT RESOLUTION (BCR): At line 16 of the VOTE PROCESSING algorithm we need to handle a vote whose interpretation is against the relationship of the nodes it touches in the current taxonomy state and any attempt to adjust it, leads to a backedge.

Definition 3 Backedge: *A latent relation between two nodes. The source node appears as a descendant in the current taxonomy whereas the target as an ancestor. Backedges remain hidden since they violate the tree's properties.*

Besides the cycle that it creates, a backedge might also violate the *Maximum Votes Satisfiability* invariant. The idea to solve this problem is to isolate the strongly connected component that the newly incoming backedge created and resolve any vote conflict locally.

We will present the logic of this transformation with a specific example on figure 4. For the sake of simplicity we assume that the incoming vote appears in a “triple form” and will be processed as such. On the left we observe the initial state of the isolated subgraph. We are asked to embody a $u \rightarrow v$ relation whose weight (number of votes) equals 3. We apply a what-if analysis. We instantiate all possible states, which the subgraph component can reach, every time we apply a vertical rotation to its nodes. These are presented on the right part of the figure. The state whose backedges add up to the minimum weight is picked as a final state. In this example we arbitrary choose between state 3 or 4. Node A and its in- or out- going edges does not take part into this computation and is displayed here just to illustrate how it interfaces with the rest of the graph.

Formally, the aforementioned is an instance of the *All-pairs Bottleneck Paths* problem [20] and aims to find out the state in which we displease (or dissatisfy) the least number of votes. It's dual equivalent problem is the *All-pairs Maximum Capacity* and can be respectively interpreted as an effort to preserve our second invariant (*Maximum Votes Satisfiability*).

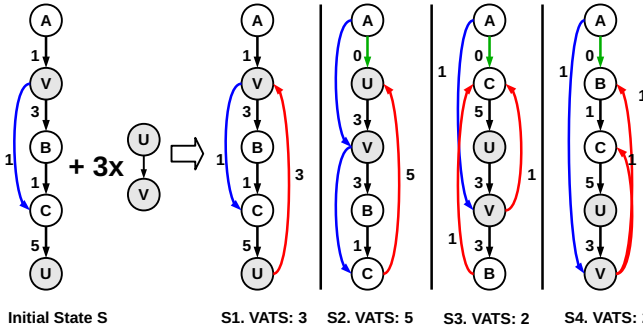


Fig. 4. BACKEDGE CONFLICT RESOLUTION What-if Analysis. VATS: Votes Against This State.

Elimination of Rising Crosslinks

In fig. 5 we observe the big picture during a BCR. The change caused by the appearance of $(u \rightarrow v)$, practically decomposes the taxonomy into 3 components. The one on top, T , right above v , remains put. The lower ones are being inverted after we identify and break the weakest edge between them. This transformation has some corner cases. Former forward edges, such as $b \rightarrow d$ now appear as crosslinks, violating the first invariant. A solution to this anomaly is offered by Algorithm 2, which simply certifies that all forward edges on path $\{v, \dots, b\}$ adhere to the obvious ancestor-descendant relation.

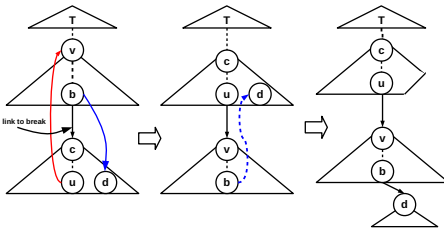


Fig. 5. The big picture of BCR and the algorithm for crosslinks elimination.

Algorithm 2. CROSSLINK ELIMINATION

```

1: for all Nodes  $b_i \in \{v \dots b\}$  do
2:   for all forward edges  $(b_i, d)$  do
3:     if  $(b_i \neq LCA(b_i, d))$  then
4:       EXPAND VERTICALLY( $b_i, d$ )
5:     end if
6:   end for
7: end for

```

4.2 Asymptotic Complexity Analysis

The core Algorithm 1 is called once for every new vote. In turn, it calls at most one of the transformations. Therefore, its worst-case asymptotic complexity is equal to the worst of the worst-case complexities of each of the transformations. Having computed all these partial complexities we state that the worst one, is the one corresponding to BACKEDGE CONFLICT RESOLUTION. Every time a backedge appears it interconnects two nodes and the path connecting them (coined BCR path) has a maximum length of n , where n denotes the number of

total nodes. This is an extreme scenario where all the tree nodes form a chain. As we already stated, at this point, the BCR algorithm forms n possible tree instances, with every one of them representing a unique cyclic rotation of the nodes making up the BCR path. For every one of these n instances, we iterate over the n nodes it consists of and explore their outgoing edges to verify whether they form a backedge or not. The number of these outgoing edges is obviously also at most $n-1$. Therefore, the overall asymptotic worst-case complexity of the BACKEDGE CONFLICT RESOLUTION transformation is $O(n^3)$.

Theorem 1. *If s denotes the number of votes, the worst-case asymptotic complexity of CrowdTaxonomy Algorithm is $O(s * n^3)$.*

Proof. VOTE PROCESSING is called s times, once for each incoming vote. Every time a single transformation is applied. The worst-case complexity of the latter equals $O(n^3)$. Therefore the worst-case asymptotic complexity of the CrowdTaxonomy algorithm is $O(s * n^3)$.

This analysis depicts a worst-case scenario and is basically presented for the sake of completeness, vis a vis the NP-Hard result presented earlier. As the experiments showed, the algorithm's behaviour in matters of absolute time is approximately linear to the number of votes. This is because (i) BCR paths are much smaller than n and (ii) outgoing links per node are also much smaller than n . In future work we plan to present an analysis of the average complexity and provide with better estimations than n which is a very relaxed upper-bound.

5 Experimentation

We evaluate the CrowdTaxonomy algorithm and demonstrate its quality under: i) Lack of (structural) information and ii) the presence of conflicting votes. Further, we perform a real world crowdsourcing experiment: we test our initial assertion, that users are capable of providing valuable input when creating a taxonomy. We also test the quality of the taxonomy produced by CrowdTaxonomy with real-world input. Our metrics for evaluating the resulting taxonomy are based on the similarity between it and the golden rule one. Consider a taxonomy T_o as the golden rule taxonomy, and T as the one derived from our algorithm. Let R_o be the set of all *Parent* \rightarrow *Child* relations on the T_o and R the set of all *Parent* \rightarrow *Child* relations on T . We define:

$$Recall = \frac{|R_o \cap R|}{|R_o|}, Precision = \frac{|R_o \cap R|}{|R|}, FScore = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision}$$

$|S|$ denotes the cardinality of a set S .

Recall certifies the completeness of the taxonomy, whereas *Precision* measures its correctness. *FScore* is a combination of the former into a harmonic mean.

We use the ACM Computing Classification System (version 1998) as the golden rule taxonomy. It contains 1473 concepts, forming a four-level tree. We

“break” this tree into distinct node (pair) relations, generate additional conflicting pairs and feed (correct and incorrect pairs) to our algorithm.

Our algorithm was written in C and we used GLib. The interface of the crowd-sourcing experiment was implemented in HTML/PHP and ran over Apache/MySQL.

5.1 Synthetic Experiments

A key challenge we face is to provide a high-quality taxonomy under incomplete structural information. The votes are given ad-hocly and there is no guarantee that they form a complete taxonomy. To examine this characteristic we form all possible *Ancestor* \rightarrow *Descendant* relations of the ACM tree and gradually feed the algorithm with a fraction of them. The result is shown in figure 6.

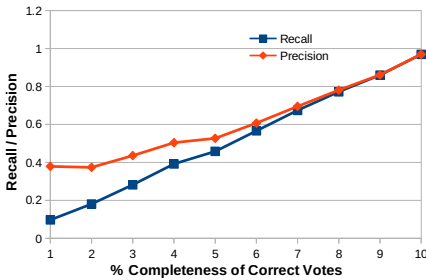


Fig. 6. Recall and Precision using a percentage of Correct votes

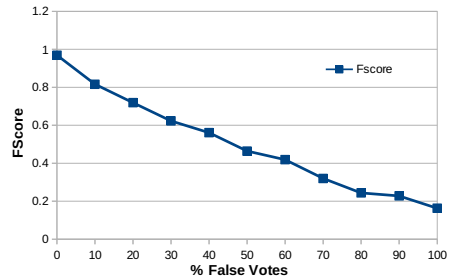


Fig. 7. FScores having a mixture of Correct and False votes

We repeatedly increase by 10% the number of available votes and measure the quality of our taxonomy. The absence of *Descendant* \rightarrow *Ancestor* relations keeps the number of *permanent* backedges to zero. Any backedges that arise are instantly resolved and one, and only one, tree instance (representing 0 conflicts) is produced. The Recall value is linear to the size of input as expected. The Precision metric equals 0.4 at the beginning and gradually increases. The taxonomy edges consist of a set of correct edges and a number of additional shortcuts, with a fraction of them playing a role of ‘noise’. As we reach 100% input completeness, the assumptions (shortcuts) are gradually eliminated and both metrics end to 1.0 verifying a sanity-check, since the experiment configuration deals with only correct votes. Next we ‘infect’ the input with additional false votes. The number of correct *Ancestor* \rightarrow *Descendant* votes remains 100% but on every iteration we increase the number of false (*Descendant* \rightarrow *Ancestor*) votes. We form these by reversing the direction of a relation between two nodes. In figure 7 we observe that even with a high number of bad input, the taxonomy quality is high. For instance, even with 30% false votes, the F score remains above 60%. Under normal circumstances, where users can provide a largely complete and correct set of A-D relations, our algorithms produce a high quality taxonomy. Note that in this configuration $Recall = Precision = FScore$.

5.2 Crowdsourcing Experiment

We asked students of our department to voluntarily participate in crowdsourcing a taxonomy. We pregenerated all the possible *Ancestor* \rightarrow *Descendant* relations and derived all the ‘false’ *Descendant* \rightarrow *Ancestor* counterpart votes. We removed all misleading concepts (e.g., with labels ‘General’ or ‘Miscellaneous’) and attached their child nodes to their direct ancestors. As the total number of possible A-D relations is very high we used a fraction of the tree with 245 nodes which led to 620 relations (A-D) plus their counterparts (D-A). The users were presented with pairs of votes, the original correct vote, and the (inverted) false one. They had to choose between the one that depicted a $A \rightarrow D$ relation, linking two concepts of the ACM tree nodes. The user task consisted of 25 pairs of votes that were presented in groups of 5. Users could also select not to answer a vote-question marking it as *unspecified*. We counted 102 distinct http-sessions but the number of collected votes was smaller than the theoretical (2550), since some users dropped out early. In contrast to commercial crowdsourcing hubs, our user base had no financial or other type of gain. We collected input for a 3-day period. Some statistics of the experiment are shown on Table 1. We observe that the false/correct votes ratio is 0.283, which testifies that users are capable of providing high-quality input. Overall, the input accounted for 94.7% of all correct relations. With these statistics, we ran a synthetic experiment with controlled input, which predicted an *FScore* (*Predicted*) whose value is close to the one in the real-world experiment. Thus, we can conclude that (i) the real-world experiment corroborated our conclusions based on the synthetic one; (ii) users can indeed provide high-quality input en route to a crowdsourced taxonomy, and (iii) despite the voluntary nature of user participation and the heuristic nature of our algorithms, the end result is a taxonomy preserving the large majority of the relationships found in expertly constructed taxonomies.

Table 1. Crowdsourcing Experiment Statistics

Total Votes	2155
Correct Votes	1501
False Votes	435
Unspecified	229
FScore	0.486
FScore (Predicted)	0.519

6 Related Work

[4], [5], [19] and [18] apply association mining rules to induce relations between terms and use them to form taxonomies. For text corpora, Sanderson and Croft automatically derive a hierarchy of concepts and develop a statistical model where term x subsumes term y if $P(x|y) \geq 0.8$ and $P(y|x) < 1$ where $P(a|b)$ defines the probability of a document to include term a assuming that term b is contained. Schmitz extended this and applied additional thresholds in order

to deal with problems caused by uncontrolled vocabulary. [6] and [14] underline the importance of folksonomies and the need to extract hierarchies for searching, categorization and navigation purposes. They present approaches that operate based on agglomerative clustering. A similarity measure is used to compute the proximity between all tags and then a bottom-up procedure takes place. Nodes under a threshold are merged into clusters in a recursive way and eventually compose a taxonomy. Heyman & Garcia-Molina in [11] present another technique with good results. Given a space of resources and tags, they form a vector for every tag and set to the i -th element the number of times it has been assigned to object i . They also use cosine similarity to compute all vectors' proximities and represent them as nodes of a graph with weighted edges that correspond to their similarity distance. To extract a taxonomy they iterate over the nodes in descending centrality order and set every of its neighbours either as children or to the root based on a threshold.

As Plangprasopchok et al. note in [17] and [16] all these approaches make the assumption that frequent words represent general terms. This does not always hold and any threshold tuning approach leads to a trade-off between accurate but shallow taxonomies against large but noisy ones. Also, all above works assume a static tag space, despite its dynamicity [10].

7 Conclusions

We have presented a drastically new approach to create taxonomies, exploiting the wisdom of crowds and their proven desire and ability to provide rich semantic metadata on several social web applications. Our contributions include the definition and analysis of the problem of crowdsourcing taxonomies. We showed how to model the problem and the required human input and the (meaningful in a crowdsourcing environment) invariant of maximum vote satisfiability. Then we proceeded to show that the resulting problem is NP-Hard. Next, we contributed a novel heuristic algorithm to online aggregate human input and derive taxonomies. We conducted both synthetic and real-world crowdsourcing experiments. Our synthetic experiments showed that when the human input is adequately complete and correct, our solution can derive high-quality taxonomies. Conversely, even when the input is incomplete and incorrect to a significant extent, still a good quality taxonomy can be constructed. Our real-world crowdsourcing experiment additionally showed that indeed humans can provide high quality input in terms of completeness and correctness (at least for the taxonomy examined). And as our synthetic results showed, when fed into our algorithms, good quality taxonomies emerge. To the best of our knowledge this is the first work to study this problem and provide a promising solution.

Future work includes optimizations, straddling the complexity-quality trade-offs, and appropriate measures for crowdsourced taxonomy quality evaluation.

References

1. Endeca, <http://www.endeca.com/>
2. Facetmap, <http://facetmap.com/>
3. Alonso, O., Lease, M.: Crowdsourcing 101: Putting the wisdom of crowds to work for you: A tutorial. In: International Conference on WSDM (February 2011)
4. Au Yeung, C.-M., Gibbins, N., Shadbolt, N.: User-induced links in collaborative tagging systems. In: Proceeding of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009 (2009)
5. Barla, M., Bieliková, M.: On deriving taxonomies: Keyword relations coming from crowd. In: Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems (2009)
6. Brooks, C.H., Montanez, N.: Improved annotation of the blogosphere via autotagging and hierarchical clustering. In: Proceedings of the 15th International Conference on World Wide Web, WWW 2006 (2006)
7. Doan, A., Ramakrishnan, R., Halevy, A.Y.: Crowdsourcing systems on the worldwide web. *Commun. ACM* (2011)
8. Franklin, M.J., Kossmann, D., Kraska, T., Ramesh, S., Xin, R.: Crowddb: answering queries with crowdsourcing. In: ACM SIGMOD Conference (2011)
9. Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of NP-Completeness
10. Halpin, H., Robu, V., Shepherd, H.: The complex dynamics of collaborative tagging. In: 16th WWW Conference (2007)
11. Heymann, P., Garcia-Molina, H.: Collaborative creation of communal hierarchical taxonomies in social tagging systems. Technical report (2006)
12. Heymann, P., Paepcke, A., Garcia-Molina, H.: Tagging human knowledge. In: Third ACM International Conference on Web Search and Data Mining, WSDM 2010 (2010)
13. Ipeirotis, P.: Managing crowdsourced human computation: A tutorial. In: International Conference on WWW (March 2011)
14. Liu, K., Fang, B., Zhang, W.: Ontology emergence from folksonomies. In: 19th ACM CIKM (2010)
15. Marlow, C., Naaman, M., Boyd, D., Davis, M.: Position Paper, Tagging, Taxonomy, Flickr, Article, ToRead. In: Collaborative Web Tagging Workshop at WWW 2006 (2006)
16. Plangprasopchok, A., Lerman, K.: Constructing folksonomies from user-specified relations on flickr. In: 18th WWW Conference (2009)
17. Plangprasopchok, A., Lerman, K., Getoor, L.: Growing a tree in the forest: constructing folksonomies by integrating structured metadata. In: 6th ACM SIGKDD Conference (2010)
18. Sanderson, M., Croft, B.: Deriving concept hierarchies from text. In: 22nd ACM SIGIR Conference (1999)
19. Schmitz, P.: WWW 2006 (2006)
20. Shapira, A., Yuster, R., Zwick, U.: All-pairs bottleneck paths in vertex weighted graphs. In: 18th ACM-SIAM SODA Symposium (2007)
21. Triantafyllou, P.: Anthropocentric data systems. In: 37th VLDB Conference (Visions and Challenges) (2011)

Generating Possible Interpretations for Statistics from Linked Open Data

Heiko Paulheim

Technische Universität Darmstadt
Knowledge Engineering Group
paulheim@ke.tu-darmstadt.de

Abstract. Statistics are very present in our daily lives. Every day, new statistics are published, showing the perceived quality of living in different cities, the corruption index of different countries, and so on. Interpreting those statistics, on the other hand, is a difficult task. Often, statistics collect only very few attributes, and it is difficult to come up with hypotheses that explain, e.g., *why* the perceived quality of living in one city is higher than in another. In this paper, we introduce *Explain-a-LOD*, an approach which uses data from Linked Open Data for generating hypotheses that explain statistics. We show an implemented prototype and compare different approaches for generating hypotheses by analyzing the perceived quality of those hypotheses in a user study.

1 Introduction

Statistical data plays an important role in our daily lives. Every day, a new statistic is published, telling about, e.g., the perceived quality of living in different cities (used as a running example throughout the following sections), the corruption in different countries, or the box office revenue of films. While it is often possible to retrieve a statistic on a certain topic quite easily, *interpreting* that statistic is a much more difficult task. The raw data of a statistic often consists only of a few attributes, collected; in the extreme case, it may only comprise a source and a target attribute, such as a city and its score. Therefore, formulating hypotheses, e.g., *why* the perceived quality of living is higher in some cities than in others is not easy and requires additional *background information*.

While there are tools for discovering correlations in statistics, those tools require that the respective background information is already contained in the statistic. For example, the quality of living in a city may depend on the population size, the weather, or the presence of cultural institutions such as cinemas and theaters. For discovering those correlations, the respective data has to be contained in the dataset. For creating useful hypotheses, the dataset should contain a larger number of attributes, which makes the compilation of such a dataset a large amount of manual work.

More severely, the selection of attributes for inclusion in a statistical dataset introduces a bias: attributes are selected since the person creating the dataset

already assumes a possible correlation. For *discovering new and unexpected hypotheses*, this turns out to be a chicken-and-egg problem: we have to know what we are looking for to include the respective attribute in the dataset. For example, if we assume that the cultural life in a city influences the quality of living, we will include background information about theaters and festivals in our dataset.

For many common statistical datasets (e.g. datasets which relate real-world entities of a common class with one or more target variables), there is background information available in Linked Open Data [2]. In the quality of living example, information about all major cities in the world can be retrieved from the semantic web, including information about the population and size, the weather, and facilities that are present in that city. Thus, Linked Open Data appear to be an ideal candidate for generating attributes to enhance statistical datasets, so that new hypotheses for interpreting the statistic can be found.

In this paper, we introduce *Explain-a-LOD*, a prototype for automatically generating hypotheses for explaining statistics by using Linked Open Data. Our prototype implementation can import arbitrary statistics files (such as CSV files), and uses *DBpedia* [3] for generating attributes in a fully automatic fashion. While our main focus is on enhancing statistical datasets with background information, we have implemented the full processing chain in our prototype, using correlation analysis and rule learning for producing hypotheses which are presented to the user.

The rest of this paper is structured as follows. In Sect. 2, we introduce our approach, show a proof-of-concept prototype, and discuss the underlying algorithms. Section 3 discusses the validity of the approach and the individual algorithms with the help of a user study. In Sect. 4, we review related approaches. We conclude with a summary and an outlook on future research directions.

2 Approach

We have developed an approach for using Linked Open Data in a way that new hypotheses for interpreting statistics can be generated. The approach starts with a plain statistic, e.g., a CSV file, and comes up with hypotheses, which can be output in a user interface. To that end, three basic steps are performed: first, the statistical data is enhanced such that additional data from Linked Open Data is added, second, hypotheses are sought in this enhanced data set by means of correlation analysis and rule learning, and third, the hypotheses that are found are presented to the user. The basic workflow of our approach is depicted in Fig. 1. We have implemented that approach in a proof-of-concept prototype.

2.1 Data Preparation

In the first step, the statistical data is prepared using our feature generation toolkit *FeGeLOD* [15]. FeGeLOD itself performs three steps: entity recognition, feature generation, and feature selection, as depicted in Fig. 2.

In the first step, the entities that the statistic is about – cities in the quality of living example – have to be mapped to corresponding URIs in Linked

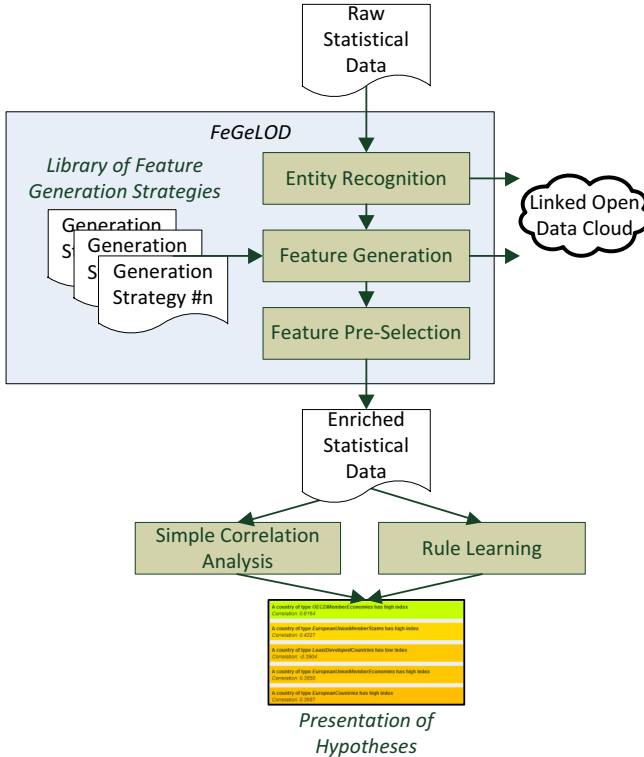


Fig. 1. Basic prototype of the Explain-a-LOD prototype

Open Data, so that additional information about those entities can be retrieved. For the first prototype of FeGeLOD, we have used a very basic mechanism for entity recognition: it retrieves all possible matching resources, e.g., such as <http://dbpedia.org/resource/Vancouver> for the city name *Vancouver*, and performs an optional type check, e.g. for `dbpedia-owl:City`. If the landing page of the first step is a disambiguation page, all disambiguated entities are followed, and the first one matching the type checks is used.

2.2 Generation of Hypotheses

Once an entity is recognized, attributes (or *features*, as they are called in data mining) can be generated for that entity as a second step. In the prototype, FeGeLOD supports six different generation strategies:

- *Simple datatype properties*, such as the population of a city.
- *Class information*. For example, a city can be of type `dbpedia-owl:City`, among others. Since DBpedia also uses *YAGO* types [17], there are also a lot of very specific types that can be used as features, such as `yago:Populated-CoastalPlacesInCanada`.

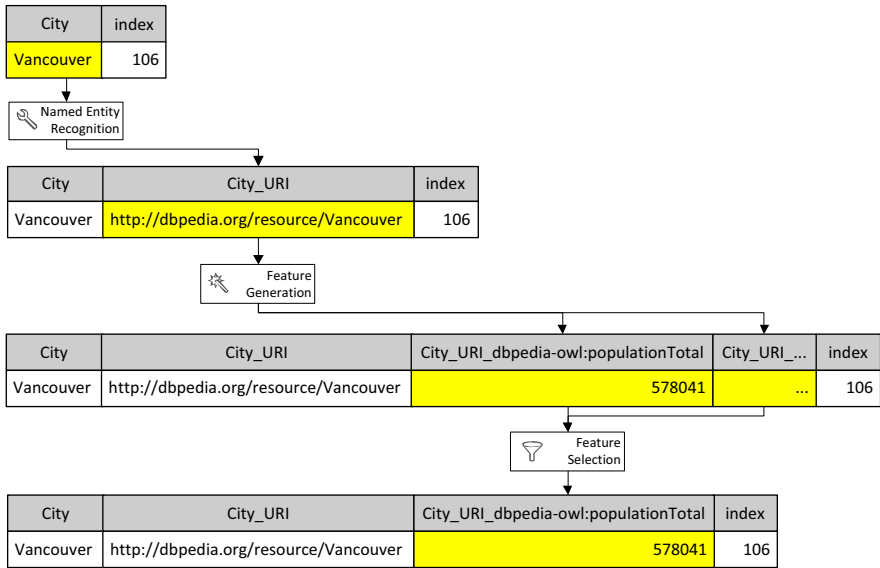


Fig. 2. The three steps performed at the data preparation stage

- *Unqualified relations.* Features are generated for incoming and outgoing relations without any information about the related entity. For example, a city may have incoming relations of type `dbpedia-owl:foundationPlace`. Those features can be generated as boolean (incoming/outgoing relations of the specified type exist or not) or numeric (counting the related entities) features¹.
- *Qualified relations.* Unlike unqualified relations, boolean or numeric features are generated including the type of the related entity, e.g., the presence or number of entities of type `dbpedia-owl:Company` which have a `dbpedia-owl:foundationPlace` relation to a city. The detailed YAGO typing system leads to a lot of very specific features, such as *number of airlines that are founded in 2000 that are located in a city*.

To illustrate how the individual strategies work, Fig. 3 shows an excerpt of DBpedia, depicting some information about the city of Darmstadt. Table 1 shows the features that are generated in this example by the individual generators.

Not all of the features generated by the different strategies are equally helpful. For example, the generator for class information may generate a feature for the

¹ We are aware that the creation of such attributes neglects the two central semantic principles of Linked Open Data, i.e., the open world assumption and the non-unique name assumption. For example, the actual number of companies and organizations founded in a city will probably be higher than that in DBpedia. However, re-interpreting that feature, e.g., as *approximate number of important companies/organizations founded in a city* fixes these issues, and still serves as a useful feature for analysing the statistic.

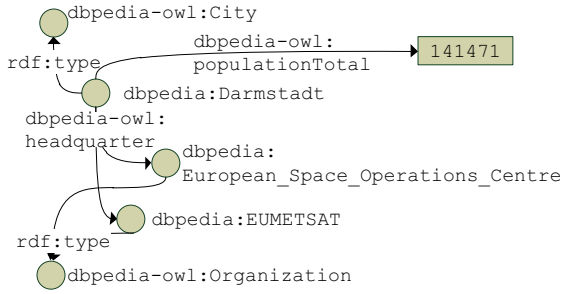


Fig. 3. An excerpt from DBpedia, showing data about Darmstadt

classes `dbpedia-owl:City` or even `owl:Thing`, which are **true** for all entities. Likewise, qualified relations may yield a large number of features which are not useful, such as the number of entities of type `yago:ArtSchoolsInParis` which are located in a city: this attribute will have a non-zero value only for one entity, i.e., Paris.

Since those features are very unlikely to produce useful hypotheses, we apply a simple heuristic to filter them out before processing the dataset in order to improve the runtime behavior of the remaining processing steps. Given a threshold p , $0 \leq p \leq 1$, we discard all features that have a ratio of more than p unknown, equal, or different values (different values, however, are not discarded for numeric features). In our previous experiments, values of p between 0.95 and 0.99 have proved to produce data sets of reasonable size without reducing the results' quality significantly [15].

The result of the data preparation step is a table with many additional attributes. That table can then be further analyzed to generate possible hypotheses. Currently, we pursue two strategies for creating hypotheses:

- The correlation of each attribute with the respective target attribute is analyzed. Attributes that are highly correlated (positively or negatively) lead to a hypothesis such as “Cities with a high value of population have a low quality of living”.
- Rule learning is used to produce more complex hypotheses which may take more than one feature into account. We have used the standard machine learning library *Weka* [4] for rule learning. Possible algorithms are class association rule mining [1], the use of separate-and-conquer rule learners [6], where in the latter case, only the first, i.e., most general rules are used, as the subsequent rules are often not valid on the whole data set.

2.3 Presentation of Hypotheses

After importing and processing a statistics file, the hypotheses found are presented to the user in a user interface, as depicted in Fig. 4. To that end, all hypotheses are verbalized. For example, a positive correlation between the type `yago:EuropeanCapitals` and the quality of living is turned into a sentence such

Table 1. Features generated for the example shown in Fig. 3

Generator	Feature Name	Feature Value
Data properties	dbpedia-owl:populationTotal	141471
Types	type_dbpedia-owl:City	true
Unqualified relations boolean	dbpedia-owl:headquarter_boolean	true
Unqualified relations numeric	dbpedia-owl:headquarter_numeric	2
Qualified relations boolean	dbpedia-owl:headquarter_type _dbpedia-owl:Organization_out_boolean	true
Qualified relations numeric	dbpedia-owl:headquarter_type _dbpedia-owl:Organization_out_numeric	2

as *In cities of type European Capitals, the quality of living is high.* Likewise, learned rules are verbalized.

All hypotheses have a quality measure. For simple correlations, it is the correlation coefficient itself. Rules learned by a rule learning algorithm also come with a confidence or accuracy measure provided by the algorithm. Therefore, the hypotheses may be sorted, presenting the most likely ones on top. Furthermore, to improve the usability, we use a color coding schema, depicting the best rated hypotheses in green, going over to red for the worst rated hypotheses.

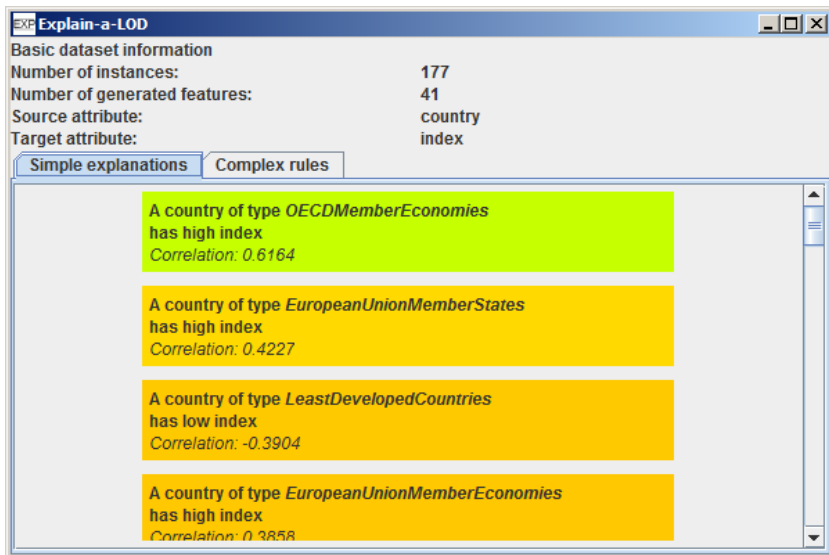


Fig. 4. Screenshot of the Explain-a-LOD User Interface

Table 2. Number of features generated for the two data sets used in the study. This table shows the numbers without any post-processing feature selection. The boolean and numerical variants of relations and qualified relations produce an equal number of features.

	Mercer	Transparency International
Data	1,205	614
Types	622	237
Relations	2,414	1,523
Qualified Relations	48,441	34,302

3 Experimental Evaluation

In order to examine the quality of the hypotheses generated by Explain-a-LOD, also with respect to the different feature generation strategies, we have asked a number of users to evaluate those hypotheses. To that end, users were presented a list of hypotheses generated by Explain-a-LOD, and they were asked to rate those hypotheses by the perceived plausibility. Furthermore, all participants were asked a number of general questions on the approach in the end.

3.1 Setup

We have conducted the user study with 18 voluntary participants, who were undergraduate and graduate students as well as researchers at Technische Universität Darmstadt. The participants were between 24 and 45 years old, 15 of them were male, 3 female.

For the evaluation, we have used two statistics datasets: the already mentioned Mercer quality of living survey with data², which comprises 218 cities, and the corruption perception dataset by Transparency International³, which comprises 178 countries. With our entity recognition approach, we could map 97.7% of the cities and 99.4% of the countries to the corresponding URIs in DBpedia.

For each data set, we have generated hypotheses with the approaches discussed above, using the different feature generation algorithms, and used the top three hypotheses from both the simple correlation analysis and the rule learning approach. Table 2 depicts the number of features generated and used in each dataset.

For the rule learning approach, we also used a joint set of all feature generators, so that rules involving features from different generators could also be found. As the joint set of features cannot produce any new hypotheses when only regarding correlations of single features, that dataset was only used with the rule learning approach. After removing duplicates (a hypothesis with only one feature can be found by both approaches), we had 37 hypotheses for the Mercer dataset and

² Data available at <http://across.co.nz/qualityofliving.htm>

³ Data available at http://www.transparency.org/policy_research/surveys_indices/cpi/2010/results

38 hypotheses for the Transparency International dataset. Each participant was asked to evaluate all 75 hypotheses⁴.

From those hypotheses, we have constructed a questionnaire listing those hypotheses for both datasets in random order, and asking for the plausibility of each hypothesis on a scale from 1 (worst) to 5 (best).

At the end of the questionnaire, the users were asked to which degree they feel that the hypotheses in total are useful, surprising, non-trivial, and trustworthy. Filling out a questionnaire took the participants between 15 and 20 minutes.

3.2 Results

The first goal was to understand which strategies for feature generation and for creating the hypotheses work well, also in conjunction. To that end, we analyzed the ratings of the respective hypotheses. Figure 5 shows the results for the Mercer dataset, Figure 6 shows the respective results for the Transparency International dataset. The intra-class correlation (i.e., the agreement score of the participants) was 0.9044 and 0.8977, respectively.

The first basic observation is that the evaluations for both datasets are very different. For the Mercer dataset, simple correlations produce the more plausible hypotheses, while for the Transparency International dataset, rule learning is significantly better in some cases. In both cases, the best rated hypotheses are produced when using the type features. In both cases, joining all the attributes in a common dataset did not lead to significantly better rules.

For the Mercer dataset, the best rated hypotheses were *Cities in which many events take place have a high quality of living* (found with correlation analysis from unqualified relations, average rating 3.94), and *Cities that are European Capitals of Culture have a high quality of living* (generated from a type feature with type `yago:EuropeanCapitalsOfCulture`, found both by correlation analysis and with a rule learner, rating 3.89). The worst rated hypotheses were *Cities where at least one music record was made and where at least 22 companies or organizations are located have a high quality of living* (generated with a rule learner from unqualified relations, rating 1.5) and *Cities that are the hometown of at least 18 bands, but the headquarter of at most one airline founded in 2000, have a high quality of living* (generated with a rule learner from qualified relations, rating also 1.5).

For the Transparency International Dataset, the two best rated hypotheses are *Countries of type Least Developed Countries have a high corruption index* (generated by correlation analysis from a type feature with type `yago:Least-DevelopedCountries`, rating 4.29), and *Countries where no military conflict is carried out and where no schools and radio stations are located have a high corruption index* (generated by rule learning from three different qualified relation features, rating 4.24). The two worst rated hypotheses are *Countries with many mountains have a low corruption index* and *Countries where no music groups*

⁴ The hypotheses used in the evaluation are listed at

<http://www.ke.tu-darmstadt.de/resources/explain-a-lod/user-study>

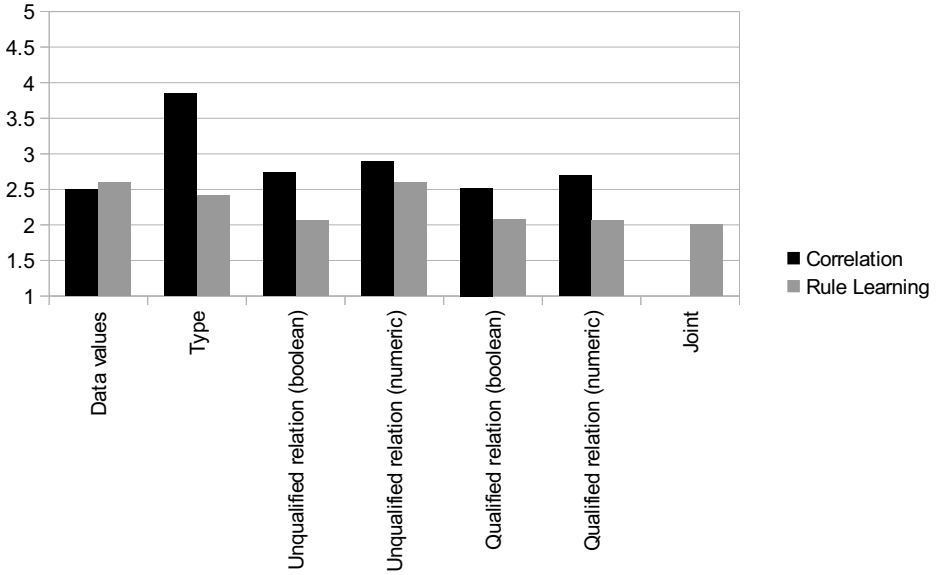


Fig. 5. Average user ratings of the hypotheses generated for the Mercer dataset, analyzed by feature generation and hypothesis generation strategy

that have been disbanded in 2008 come from have a high corruption index (both generated by correlation analysis from a qualified relation feature, ratings 1.39 and 1.28, respectively).

There are some hypotheses that are rated badly, because the explanations they hint at are not trivial to see. For example, one hypothesis generated for the Mercer dataset is *Cities with a high longitude value have a high quality of living* (average rating 1.52). When looking at a map, this hypothesis becomes plausible: it separates cities in, e.g., North America, Australia, and Japan, from those in, e.g., Africa and India. Interestingly enough, a corresponding hypothesis concerning the latitude (which essentially separates cities in the third world from those in the rest of the world) was rated significantly ($p < 0.05$) higher (rating 3.15). Another example for an hypothesis that is not trivial to interpret is the following: *Countries with an international calling code greater than 221 have a high corruption index* (rating 1.69). Those calling codes mostly identify African countries. On the other hand, the following hypothesis is rated significantly higher (rating 4.0): *Countries in Africa have a high corruption index*.

The second goal of the user study was to get an impression of how the overall usefulness of the tool is perceived. Figure 7 shows the results of the general questions. The hypotheses got positive results on three of the four scales, i.e., the users stated that the results were at least moderately useful, surprising, and non-trivial. The latter two are significantly better than the average value of three with $p < 0.05$. The trustworthiness of the results, on the other hand, was not rated well ($p < 0.01$). These results show that the tool is well suited

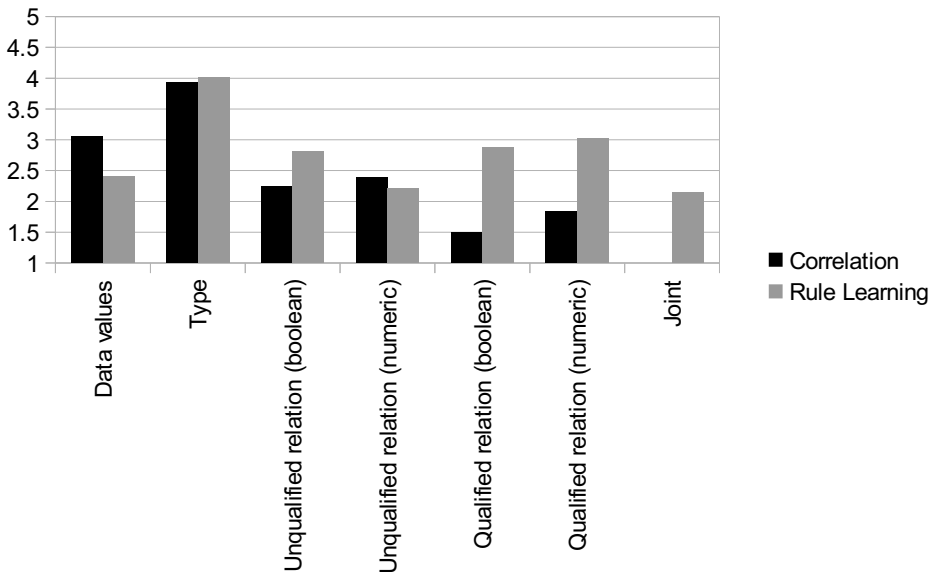


Fig. 6. Average user ratings of the hypotheses generated for the Transparency International dataset, analyzed by feature generation and hypothesis generation strategy

for generating *hypotheses*, but these hypotheses always need a human judging whether these hypotheses are valid explanations or not.

At the end of the questionnaire, users were asked to give some additional comments. One user was asking for detail information on certain explanations, e.g., showing the average corruption of African and non-African states for a hypothesis such as *Countries in Africa have a high corruption index*. Another user remarked that some rules are hard to comprehend without background knowledge (such as those involving latitude/longitude values, as discussed above).

Another user remarked that longer hypotheses were in general less plausible. This may partly explain the bad performance of the rule-based approaches on the Mercer dataset. Rule learning approaches most often seek to find rules that have a good coverage and accuracy, i.e., split the dataset into positive and negative examples as good as possible. Since rule learning algorithms may choose combinations of arbitrary features for that, it may happen that an unusual combination of features leads to a good separation of the example space, but that the resulting rule is not perceived as a very plausible one.

One example is the following rule, which was among the worst rated hypotheses (average rating of 1.5): *Cities which are the hometown of at least 18 bands, but are the headquarter of at most one airline founded in 2000, have a high quality of life*. While the second condition may increase the rule's accuracy by some percent, it decreases the perceived plausibility of the rule, mostly since there is no obvious coherence between bands and airlines. In contrast, the following rule received a significantly ($p < 0.01$) higher average rating (2.72): *Cities that are the origin of at least 33 artists and bands have a high quality of life*. On the other

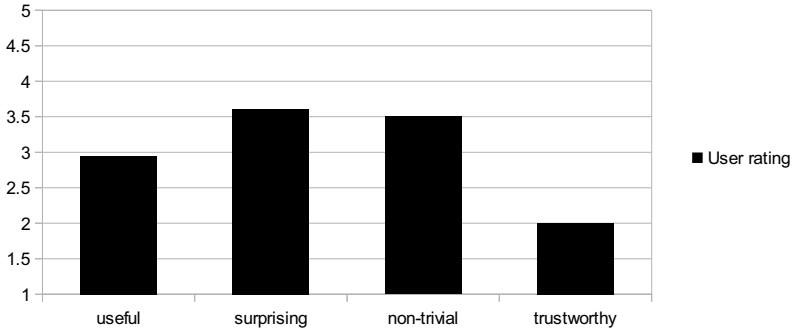


Fig. 7. Results of the overall rating of the Explain-a-LOD hypotheses

hand, the first rule has an accuracy of 98.0%, while the second rule has an accuracy of only 88.6%. This shows that, while the accuracy of rules may increase with additional, non-related features, this does not necessarily imply an increase in the perceived plausibility. A similar observation can be made for correlation analysis: the best-rated hypothesis for the Transparency International dataset, *Countries of type Least Developed Countries have a high corruption index* is actually the one in the set of hypotheses with the lowest correlation (Pearson's correlation coefficient 0.39, rating 4.33).

Another observation made is that rule-based approaches are capable of finding very exact conditions, i.e., they find the value which separates best between positive and negative examples. One example are the following two corresponding rules: *Countries with a high HDI have a low corruption index* (average rating: 4.0, found with correlation analysis), and *Countries with a HDI less than 0.712 have a high corruption index* (average rating: 3.39, found with rule learner). While both hypotheses express the same finding, the second one, which is formulated in a more specific way, is rated significantly ($p < 0.05$) lower. These examples show that very accurate rules are not always perceived as plausible at the same time.

4 Related Work

There is a vast body of work that is concerned with the analysis of statistical data [14]. Given a statistic, there are various methods to find out correlations and interrelations of the attributes contained in those statistics. Highly developed toolkits such as *R* [9] can be used for performing such analyses.

Those methods always assume that all the possible attributes are known, and thus, they are only capable of finding correlations between attributes that are included in the statistic. The work presented in this paper can be seen as a complement to those approaches, as it enhances a dataset by a multitude of attributes that can then be examined by such statistical analysis algorithms and tools.

One of the works closest to Explain-a-LOD is proposed by Zapilko et al. [20]. The authors propose a method for publishing statistical data as linked data,

which allows for combining different of such data sets. Kämpgen and Harth suggest a similar approach for analyzing statistical linked data with online analytical processing (OLAP) tools [11]. They discuss a common schema for such data and present various case studies. While OLAP allows for asking for specific correlations (i.e., the user has to come up with the hypotheses by himself upfront), our approach *generates* hypotheses automatically. Furthermore, while we are able to exploit any arbitrary, general-purpose datasets, such as DBpedia, the authors of the two approaches are restricted to specialized statistical datasets, following a specific schema. Nevertheless, including such specific statistical linked data sets in our approach may help increasing the quality of our hypotheses significantly.

g-SEGS [13] uses ontologies as background knowledge in data mining tasks. Ontologies are used as additional taxonomic descriptions for nominal attributes. For example, a nominal attribute with the values *Student*, *Apprentice*, *Employee*, *Self-employed*, and *Unemployed* may be augmented with a taxonomy of those values. Thus, regularities that hold for all people in education (regardless of whether they are students or apprentices) may be found better. In contrast to our approach, *g-SEGS* uses T-Box information, while we use A-Box information. Furthermore, in *g-SEGS*, the ontology has to be known in advance and mapped to the dataset manually. This makes it difficult to discover *new* hypotheses, since the designer of the ontology can be tempted to model only those facts in the ontology that are considered relevant for the mining problem at hand.

SPARQL-ML [10] is an approach that foresees the extension of the SPARQL query language [18] with a specialized statement to learn a model for a specific concept or numeric attribute in an RDF dataset. Such models can be seen as explanations in the way we use them in *Explain-a-LOD*. However, the approach requires support of the endpoint in question, e.g., DBpedia, to support the SPARQL-ML language extensions. In contrast, our approach works with any arbitrary SPARQL endpoint providing Linked Open Data.

Mulwad et al. have proposed an approach for annotating tables on the web [12]. The authors try to automatically generate links to DBpedia both for entities in the table as well as for column names, which are linked to classes in ontologies. Unlike the approach presented in this paper, the authors are not concerned with creating hypotheses. Since tables are typical ways to present statistical data on the web, their approach could be a useful complement to the *Explain-a-LOD* for generating hypotheses on arbitrary tabular statistical data found on the web.

5 Conclusion and Future Work

In this paper, we have introduced *Explain-a-LOD*, an approach for using Linked Open Data as a means to interpret statistics. Given a “plain” statistics file, i.e., containing only a source and a target variable, such as a city and a numerical indicator for that city, we map the values of the source variable to entities in Linked Open Data, gather additional attributes from those Linked Open Data entities, and use those attributes to generate hypotheses for explaining the statistic using correlation analysis as well as rule learning. The whole process from

loading the statistic to presenting the hypotheses can be performed in a fully automatic manner.

We have conducted a user study, in which we asked people to rate hypotheses generated by Explain-a-LOD for two different datasets, as well as to give a general impression of the tool. The hypotheses received mixed ratings: while some approaches produce hypotheses of high value (especially those exploiting the types in DBpedia), others are not suitable for producing good hypotheses. In the overall rating, the study participants stated that the hypotheses generated by Explain-a-LOD are useful, surprising, and non-trivial.

Although often useful, the hypotheses generated by Explain-a-LOD should be handled with care. The data preparation algorithm disrespects some essential fundamentals of Linked Open Data, such as the open world assumption, when generating attributes such as *number of organizations with headquarter in that city*. Furthermore, there might be cultural biases in the Linked Open Data sets used. When generating a feature such as *number of (famous) persons born in that city*, a larger amount of information on, e.g., US celebrities may introduce a cultural bias [5]. Likewise, we have observed a slight bias in DBpedia towards facts from popular culture, since many of our hypotheses were concerned with bands and music records.

Additionally, Explain-a-LOD cannot distinguish correlations from causal relations: from an explanation such as *countries where many companies have their headquarter are less corrupt*, we cannot tell whether companies tend to choose such countries with low corruption as headquarters, or whether a flourishing economy leads to a lower corruption.

In the future, we want to extend our approach to other Linked Open Data sets, such as Freebase, and compare the quality of hypotheses that can be obtained with data preprocessing using such different datasets. Since many datasets are already linked to DBpedia, drawing background information from those datasets is not difficult once the entities in the statistic are mapped to DBpedia. It may also be useful to produce deeper features, such as *number of companies in that country that have more than X Mio.\$ turnover*, which do not only take direct neighbors of the entity into account, but also further information about those entities. However, the explosion of the search space has to be taken care of in that case.

We have implemented Explain-a-LOD as a proof of concept prototype with a set of simple algorithms and toolkits. That prototype can be improved with respect to many aspects. The entity recognition step can be enhanced by using frameworks such as the DBpedia lookup service, or by adapting the algorithm by Mulwad et al. discussed above. The generation of hypotheses can be enhanced by adding further mechanisms, such as subgroup discovery [19], and more sophisticated algorithms for feature selection [8].

We have also recognized that some of the hypotheses generated by rule learners are not considered plausible if the conditions are not semantically coherent, such as in *Cities which are the hometown of at least 18 bands, but are the headquarter of at most one airline founded in 2000, have a high quality of life*, where

bands and airlines are not semantically close, which lowers the total plausibility. A rule with two conditions involving, e.g., bands and TV stars, or airlines and logistics companies, would probably be perceived more plausible, since the semantic distance between the conditions is lower. Therefore, an interesting research direction would be finding accurate, but semantically coherent rules.

Concerning the presentation of the hypotheses, several improvements can be thought of. The sorting of hypotheses by their rating is essential to the user, since the best hypotheses are expected to be on top. However, our user study showed that the natural ratings (such as the correlation coefficient for simple attributes) do not always reflect the perceived plausibility. In future user studies, we want to explore the impact of different rating measures for hypotheses. Furthermore, the verbalization of hypotheses does not always work too well because of mixed quality of labels used in the datasets [7]. Here, we aim for more intuitive and readable verbalizations, such as proposed in [16].

Finally, an interactive user interface would be helpful, where the user can mark implausible hypotheses (such as a correlation between the number of mountains in a country and the country's corruption index) and receive an explanation and/or an alternative hypothesis. Taking the informal feedback from the user study into account, it would also be helpful to provide evidence for hypothesis, e.g., list those instances that fulfill a certain condition. Such a functionality might also help to improve the trust in the hypotheses generated by Explain-a-LOD, which was not perceived very high in our user study.

In summary, we have introduced an approach and an implemented prototype that demonstrates how Linked Open Data can help in generating hypotheses for interpreting statistics. The evaluation of the user study show that the approach is valid and produces useful results.

Acknowledgements. This work was supported by the German Science Foundation (DFG) project FU 580/2 “Towards a Synthesis of Local and Global Pattern Induction (GLocSyn)”. The author would like to thank Sebastian Döweling, Aristotelis Hadjakos, and Axel Schulz for their valuable advice on designing the evaluation, and everybody who took the time to participate in the user study.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 207–216 (1993)
2. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems* 5(3), 1–22 (2009)
3. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. *Web Semantics - Science Services and Agents on the World Wide Web* 7(3), 154–165 (2009)
4. Bouckaert, R.R., Frank, E., Hall, M., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: WEKA — Experiences with a Java open-source project. *Journal of Machine Learning Research* 11, 2533–2541 (2010)

5. Callahan, E.S., Herring, S.C.: Cultural bias in wikipedia content on famous persons. *Journal of the American Society for Information Science and Technology* 62(10), 1899–1915 (2011)
6. Cohen, W.W.: Fast effective rule induction. In: *Twelfth International Conference on Machine Learning*, pp. 115–123. Morgan Kaufmann (1995)
7. Ell, B., Vrandečić, D., Simperl, E.: Labels in the Web of 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. 162–176. Springer, Heidelberg (2011)
8. Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L.A. (eds.): *Feature Extraction – Foundations and Applications*. Springer (2006)
9. Ihaka, R.: R: Past and future history. In: *Proceedings of the 30th Symposium on the Interface* (1998)
10. Kiefer, C., Bernstein, A., Locher, A.: Adding Data Mining Support to SPARQL Via Statistical Relational Learning Methods. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) *ESWC 2008. LNCS*, vol. 5021, pp. 478–492. Springer, Heidelberg (2008)
11. Kämpgen, B., Harth, A.: Transforming statistical linked data for use in olap systems. In: *7th International Conference on Semantic Systems, I-SEMANTICS 2011* (2011)
12. Mulwad, V., Finin, T., Syed, Z., Joshi, A.: Using linked data to interpret tables. In: *Proceedings of the First International Workshop on Consuming Linked Data, COLD 2010* (2010)
13. Novak, P.K., Vavpetič, A., Trajkovski, I., Lavrač, N.: Towards semantic data mining with g-segs. In: *Proceedings of the 11th International Multiconference Information Society, IS 2009* (2009)
14. Ott, R.L., Longnecker, M.: *Introduction to Statistical Methods and Data Analysis*. Brooks/Cole (2006)
15. Paulheim, H., Fürnkranz, J.: Unsupervised Feature Generation from Linked Open Data. In: *International Conference on Web Intelligence, Mining, and Semantics, WIMS 2012* (2012)
16. Piccinini, H., Casanova, M.A., Furtado, A.L., Nunes, B.P.: Verbalization of rdf triples with applications. In: *ISWC 2011 – Outrageous Ideas track* (2011)
17. 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 (2007)
18. W3C: SPARQL Query Language for RDF (2008), <http://www.w3.org/TR/rdf-sparql-query/>
19. Wrobel, S.: An algorithm for multi-relational discovery of subgroups. In: *Symposium on Pattern Discovery in Databases (PKDD 1997)* (1997)
20. Zapilko, B., Harth, A., Mathiak, B.: Enriching and analysing statistics with linked open data. In: *Conference on New Techniques and Technologies for Statistics, NTTs* (2011)

Green-Thumb Camera: LOD Application for Field IT

Takahiro Kawamura^{1,2} and Akihiko Ohsuga²

¹ Research & Development Center, Toshiba Corp.

² Graduate School of Information Systems,
University of Electro-Communications, Japan

Abstract. Home gardens and green interiors have recently been receiving increased attention owing to the rise of environmental consciousness and growing interest in macrobiotics. However, because the cultivation of greenery in a restricted urban space is not necessarily a simple matter, 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. Therefore, we propose an Android application, Green-Thumb Camera, to query a plant from LOD cloud to fit environmental conditions based on sensor information on a smartphone, and overlay its grown form in the space using AR.

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

1 Introduction

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 paper, 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 [25] 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.

The remainder of this paper is organized as follows. Section 2 describes our proposed service, focusing on plant recommendation and the AR function. Section 3 reports an experiment, and section 4 outlines related work, mainly on AR-based services. Section 5, the final section, presents conclusions and identifies future issues.

2 Proposal of Plant Recommendation Service

2.1 Problems and Approaches

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 [1], 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.

¹ 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. Example of plant display (top: before growth, bottom: after growth)

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

Sunlight

This factor indicates the illuminance suitable for growing each plant and has several levels such as shade, light shade, sunny [4,5].

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

smartphone 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. If it is less than 3000 lux, it is deemed to be a shade area. If it is more than 3000 lux but less than 10000 lux, it is deemed to be light shade, and if more than 10000 lux, it is deemed to be a sunny 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) [6], using the current month and area (described below), instead of the current temperature. The temperature for indoor plants from November to February is the average winter indoor temperature for each prefecture from WEATHERNEWS INC.(WN) [7]. 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 [8,9].

To get the current month, we simply used 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 [4]. 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.

Plant LOD and SPARQL Query. 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 [30]. 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

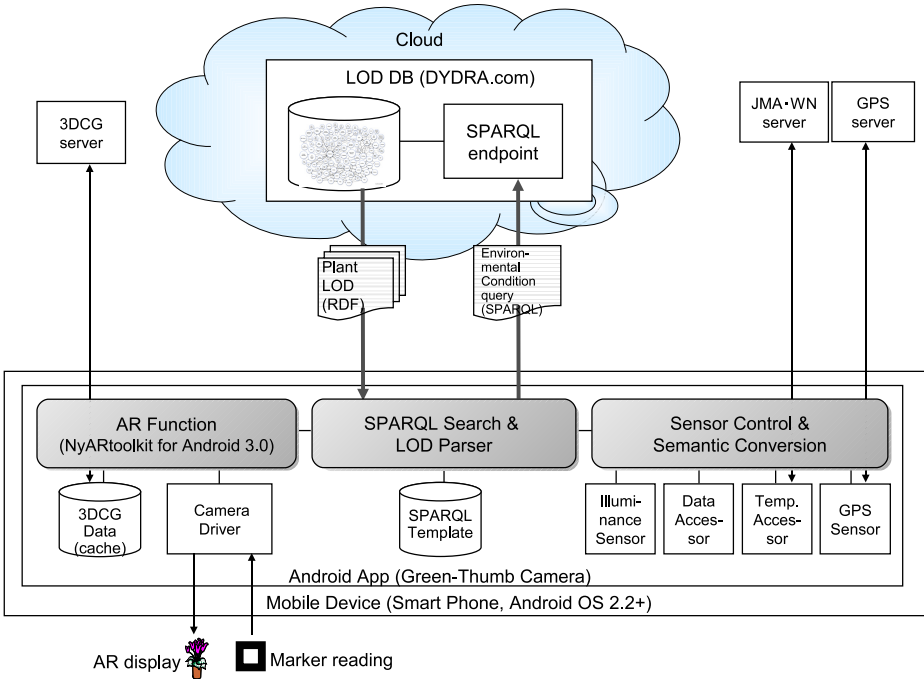


Fig. 3. Service architecture

intelligence on the net and adopted an approach of selecting a plant by querying with SPARQL.

In fact, we consider that SPARQL is intrinsically suited for information search 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.

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. We used a semi-automatic generation system for metadata from web pages that we had developed previously. Then, we created RDF data based on sentence structures and tables that frequently occur on gardening sites, and combined it with the existing Plant LOD (Fig. 3 left).

The Plant LOD is RDF data, in which each plant is an instance of “Plant” class of DBpedia [10] ontology. DBpedia has already defined 10,000+ plants as types of the Plant class and its subclasses such as “FloweringPlant”, “Moss” and

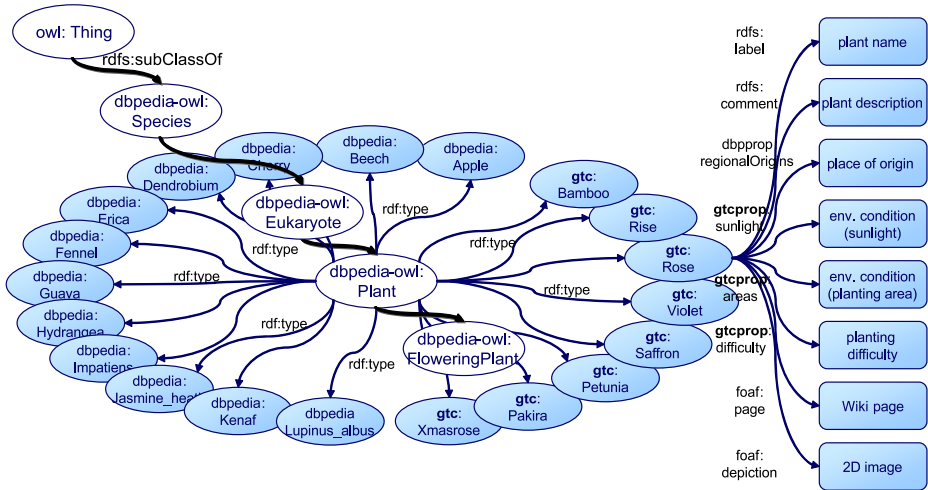


Fig. 4. Overview of Plant LOD

“Fern”. In addition, we created approx. 100 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 11 Properties to represent necessary attributes for plant cultivation, 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 }. Fig. 4 illustrates the overall architecture of the Plant LOD, where prefixes **gtc:** and **gtcp:** mean newly created instances and attributes. The Plant LOD is now stored in a cloud DB (DYDRA [11]) and a SPARQL endpoint is offered to the public.

The semi-automatic creation of LOD in this paper is greatly inspired by an invited talk of T. Mitchell at ISWC09 [29], and involves a boot strapping method based on ONTOMO [31] and a dependency parsing based on WOM Scouter [32]. But the plant names can be easily collected from a list on any gardening site and we have already defined the necessary attributes based on our service requirements. Therefore, what we would like to collect in this case is the value of the attribute for each plant. As the boot strapping method [12], we first generate specific patterns from web pages based on some keys, which are the names of the attributes, and then we apply the patterns to other web pages to extract the values of the attributes. This method is mainly used for the extraction of $\langle \textit{property}, \textit{value} \rangle$ pairs from structured part of a document such as table and list. However, we found there are many (amateur) gardening sites that explain the nature of the plant only in plain text. Therefore, we created an extraction method using the dependency parsing. It first follows the modification relation in a sentence from a seed term, which is the name of the plant or

the attribute, and then extract triples like $\langle \textit{plantname}, \textit{property}, \textit{value} \rangle$ or $\langle -, \textit{property}, \textit{value} \rangle$. Either way, a key or seed of extraction is retrieved from our predefined schema of Plant LOD to collate the existing LOD like DBpedia. Also, for correction of mistakes, we extracted the values of a plant from more than 100 web sites. If the values are identical, we sum up Google PageRanks of their source sites and determine the best possible value and the second-best. Finally, a user determines a correct value from the proposed ones. We conducted this semi-automatic extraction of the values for the 13 attributes of the 90 plants that we added, and then created the Plant LOD. In a recent experiment, the best possible values achieved an average precision of 85% and an average recall of 77%. We are now conducting more detailed evaluation, thus the results will be discussed in another paper.

The SPARQL query includes the above-mentioned environmental factors obtained from the sensors in 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 [13] 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 Function. This application requires a smartphone running Google Android OS 2.2+ and equipped with a camera, GPS, and an illuminance sensor. For the AR function, we used NyARToolkit for Android [17], which is an AR library for Android OS using a marker. It firstly detects the predefined marker (Fig. 5)

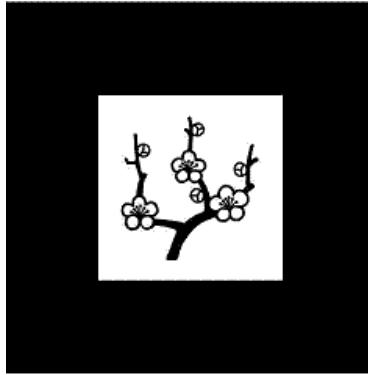


Fig. 5. AR marker (6cm × 6cm)

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.

2.3 Implementation of LOD/SPARQL for Android

In terms of implementation of SPARQL query and LOD analysis by Java on the Android OS, we developed a library to handle them.

For the creation of SPARQL query, we first prepared some query templates in SPARQL grammar. Then, we select a template as necessary, and replace environmental parameters in the template with the factors obtained from the sensors.

For the analysis of the returned LOD, we first designated the returned format as XML. Some SPARQL endpoints can return the result in JSON (JavaScript Object Notation) format whose grammar is simpler than that of XML. However, parsing a JSON document requires loading of the whole data stream, which consumes the local memory according to the size of the result content as well as XML DOM (Document Object Model) parser. Therefore, we used XML with XmlPullParser of android.util package, which is an event-driven parser like SAX (Simple API for XML), but faster than it.

TAT (Turn Around Time) of query to result is negligible compared to the following procedure to load the 3DCG files. Furthermore, battery consumption poses no problem, unlike in the case of repeating sensor invocations. Note that CPU is Qualcomm Snapdragon 1GHz and connection is WCDMA in our experiment, which is one generation ago.

For more advanced use of LOD/SPARQL in future, however, we are considering the use of ARQoid [16], which is a porting of Jena's ARQ SPARQL query engine to the Android OS, and androjena [15], which is a porting of Jena semantic web framework to the Android OS, although we need to examine the trade-off

of the functionality and performance overhead. Using these libraries, there is an Android Application called Sparql Droid [14]. It can load the local ontology in N3 format stored in the SD card, and query it (or the external SPARQL endpoints) with SPARQL, and it also allows for reasoning over a small ontology.

3 Experimental Test of Plant Recommendation

Fig. 6 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. To determine it's effectiveness, we would like to conduct some evaluation by a group of potential users in the near future.

4 Related Work

Recently, the remarkable progress of mobile devices has realized the AR function ubiquitously. Mobile devices and AR have a strong affinity because it becomes possible to overlay virtual information on reality everywhere. There are already several reports in the literature and commercial services have been proposed, which can be roughly classified into two categories depending on AR use: to annotate text information to the real object and/or materialize the virtual object in the real scene.

The former includes Sekai Camera [18], which sparked an AR boom in Japan, and Layar [19], VTT(Technical Research Centre of Finland) [20], and Takemura et al. [21]. Sekai Camera displays tags related to the real objects existing in a town, which show the users' comments and reviews. Layar annotates the text information for restaurants, convenience stores and spots on a landscape, and then provides their search function. Research at VTT concerned a system enabling a worker assembling industrial components to see the next parts and how to attach them through a camera. Takemura realized a system employing a wearable computer for annotating information on buildings.

The latter includes My.IKEA [22] and USPS Virtual Box Simulator [23]. My.IKEA realized simulation of furniture arrangement in the user's home through a camera by displaying 3DCG of the furniture on a corresponding marker that comes with a catalog. Virtual Box Simulator is a system to show 3DCG boxes for courier services for determining the suitable box size for an object to be dispatched. The service that we propose in this paper also adopts






CUT	PICTURE	ACTION	DIALOGUE
1		Find a place to put a new plant	“Oh, it has died here!”
2		Put the marker there	
3		See the marker through the camera view	“Show me what you can do!”
4		(Camera view displaying AR) Flick the camera view to left	“Rose...I don't like it much”
5			“Tulip, I like this one”

Fig. 6. Result of plant recommendation

the approach of materializing virtual objects in real scenes and displays 3DCG of non-existent objects as well. However, while the other systems materialize the predefined objects statically bound to the markers, our service materializes more adaptive objects by using the recommendation function according to the real situation estimated by the sensors.

Moreover, we introduce three kinds of research on combining AR with another technique. Regarding the combination with the recommendation function, Guven et al. [24] show 3DCG avatars of real reviewers for a product by reading a marker on the product, and then provide useful information on that product through conversation with the avatars. Our service also shows adaptive information in context with the AR. However, while the AR of this research is only used to show the avatar, AR of our service shows the recommended object itself and overlays it on the real scene to check the aesthetic balance with the surrounding. Therefore, it would be a more practical use of AR.

Regarding the combination with software agents, Nagao et al. proposed agent augmented reality [25] a decade ago and introduced the applications of shopping support and a traveler's guide system.

Furthermore, regarding the combination with plants, there is research by Nishida et al. [26]. They used a 3DCG fairy personifying the plant and whose physical appearance represents the plant's physical condition, thus introducing a game flavor to plant cultivation. Our service also uses AR for the plant growth. However, while they focused on plant cultivation, our service is for the planting and selection of plants and for checking whether they will blend in with the scenery. In fact, there has been little ICT research on plants for non-expert users who enjoy gardening, although precision farming includes agricultural field analysis using sensors for the expert. The most practical service for non-expert users may still be a search engine for the plant names. Focusing on those non-expert users, we provide adaptive information in context by combining the semantic information from the sensor and LOD with AR.

Finally, apart from AR, we introduce two kinds of research regarding sensors and semantics. The first one is Semantic Sensor Network(SSN), in which sensor data is annotated with semantic metadata to support environmental monitoring and decision-making. SemSorGrid4Env [28] is applying it to flood emergency response planning. Our service architecture is similar to SSN. However, instead of searching and reasoning within the mashuped semantic sensor data, we assume the existence of LOD on the net, to which the sensor data is connected.

The second one is about social sensor research, which integrates the existing social networking services and physical-presence awareness like RFID data and twitter with GPS data to encourage users' collaboration and communication. Live Social Semantics(LSS) [27] applied it to some conferences and suggested new interests for the users. It resembles our service 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.

5 Conclusion and Future Work

In this paper, we proposed an ‘agent’ service, Green-Thumb Camera, which works on a smartphone equipped with sensors, LOD and AR to enable users who lack gardening expertise to select a plant fitting the environmental conditions.

In the near future, first, we intend to summarize the semi-automatic generation of Plant LOD mentioned in section 2.2. We would also like to apply this framework of environmental sensing → semantic conversion → LOD Cloud (→ AR display) in other fields that would benefit from greater IT support. This vision is expressed in the subtitle of this paper. In particular, we are considering the provision of support for the greening business, which addresses environmental concerns, and for agribusiness in regard to the growing food problem. If the former is the case, target plants would be “Lawn” or “Sedum” in most cases. If the latter is the case, they are “Wheat”, “Rise”, “Corn” and “Bean”. For all those plants, there are sufficient knowledge about their cultivation on the net, but utilization of those knowledge in the field requires laptop PCs and keyword seaches. However, using this framework of the sensor and LOD, we can create a new service for the smartphone, which automatically shows instructions suitable for the current condition of a plant (we did not use a built-in camera to take a photo in this paper, but analysis of the photo of leaf, for example, will enables us to estimate protein content of the plant). It would be appealing as a simplified precision farming without capital investment. In field research, exploitation of mobile and facility sensors is now prevailing, but applications are still vague although sensor information is overflowing. By serving as an intermediary interpreting the semantics of sensor information and connecting it to the collective intelligence on the net, we seek to exploit the tremendous potential of LOD.

References

1. Srinivasan, A.: Handbook of precision agriculture: principles and applications. Routledge (2006)
2. Berners-Lee, T.: Design Issues: Linked Data (2006), <http://www.w3.org/DesignIssues/LinkedData.html>
3. Linked Data - Connect Distributed Data across the Web, <http://linkeddata.org/>
4. Neo-Green Space Design. Seibundo Shinkosha Publishing (1996)
5. engeinavi (in Japanese), <http://www.engeinavi.jp/db/>
6. Japan Meteorological Agency, <http://www.jma.go.jp/jma/indexe.html>
7. weathernews, <http://weathernews.com/?language=en>
8. Makimo Plant (in Japanese), <http://www.makimo-plant.com/modules/maintenance/index.php>
9. Angyo: The Village of Garden Plants, <http://www.jurian.or.jp/en/index.html>
10. 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.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)

11. DYDRA, <http://dydra.com/>
12. 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)
13. W3C: SPARQL 1.1 Query Language Working Draft (May 12, 2011), <http://www.w3.org/TR/2011/WD-sparql11-query-20110512/>
14. Sparql Droid, <https://market.android.com/details?id=com.monead.semantic.android.sparql>
15. androjena, <http://code.google.com/p/androjena/>
16. ARQoid, <http://code.google.com/p/androjena/wiki/ARQoid>
17. NyARToolkit for Android, <http://sourceforge.jp/projects/nyartoolkit-and/>
18. Sekai Camera, <http://sekaicamera.com/>
19. Layar, <http://layar.jp/>
20. Woodward, C., et al.: Demonstration of assembly work using augmented reality. In: Proc. of 6th ACM International Conference on Image and Video Retrieval, CIVR (2007)
21. Tenmoku, R., Kanbara, M., Yokoya, N., Takemura, H.: Annotation overlay with a wearable computer using augmented reality. In: Proc. of 1st CREST Workshop on Advanced Computing and Communicating Techniques for Wearable Information Playing (2002)
22. My.IKEA, <http://www.youtube.com/watch?v=0javjTvzIMw>
23. USPS: Priority Mail - Virtual Box Simulator, <https://www.prioritymail.com/simulator.asp>
24. Guven, S., Oda, O., Podlaseck, M., Stavropoulos, H., Kolluri, S., Pingali, G.: Social Mobile Augmented Reality for Retail. In: Proc. of IEEE International Conference on Pervasive Computing and Communication, PerCom (2009)
25. Nagao, K.: Agent Augmented Reality: Agents Integrate the Real World with Cyberspace. In: Ishida, T. (ed.) Community Computing: Collaboration over Global Information Networks (1998)
26. Nishida, T., Owada, S.: MOEGI: Plant Fostering by the Assistance of Augmented Reality. In: Proc. of 14th Workshop on Interactive Systems and Software, WISS (2006)
27. 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)
28. 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)
29. 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)

30. 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)
31. 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)
32. 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)

Assembling Rule Mashups in the Semantic Web

Guillermo González-Moriyón¹, Luis Polo¹, Diego Berrueta¹,
Carlos Tejo-Alonso¹, and Miguel Iglesias²

¹ Fundación CTIC
Gijón, Asturias, Spain
`name.surname@fundacionctic.org`
<http://www.fundacionctic.org/>
² ArcelorMittal
Avilés, Asturias, Spain
`miguel.iglesias@arcelormittal.com`
<http://www.arcelormittal.com/>

Abstract. This paper introduces RIF Assembler, a tool that reuses knowledge to automatically construct rule-based systems. Our novel approach is based on (a) annotating domain rules with metadata and (b) expressing assembly instructions as metarules that manipulate the annotations. We leverage the power of RIF as a rule interchange format, and of RDF and OWL as languages for rule annotations. RIF Assembler has applications in many scenarios. This paper presents two of them in increasing order of sophistication: a simplistic example related to the health care industry, and an actual usage in the steel industry that involves the construction of a decision-support system driven by business process descriptions.

Keywords: rules, RIF, metarules, rule mashups.

1 Introduction and Motivation

Rule-based systems are widely used to implement business applications within organizations. Rules capture knowledge in a declarative form. Complex logic can be assembled by combining small and understandable rules.

A category of software products known as BRMS (Business Rule Management Systems) aims to facilitate the development and maintenance of rule-based systems. The market for these tools is dominated by IBM, Oracle and Red Hat. Although they are feature-full, BRMSs are not silver bullets. The management of large collections of rules within an organization is still challenging (rule-based applications may contain thousands of rules). Furthermore, although rules are small, understandable and declarative, their reuse between systems is not always straightforward. Rules are application-specific and bound to a particular technology.

RIF aims to solve the latter issue by providing a common interchange format for rules [2]. RIF is a W3C standard that opens the door to a new scenario

in which behavioral knowledge can be exchanged using web standards, as a complement to the exchange of static models (OWL ontologies) and factual data (RDF descriptions). Nevertheless, to deal with the issue of the rules being application-specific, and to create rule mashups, it becomes necessary to select, adapt and modify them so they can fit into new roles. These transformations permit reuse of the rules across many applications.

RIF Assembler is an application conceived to automate and simplify the construction of rule-based systems that gather already-available knowledge pieces. It permits assembly of new rulesets by picking and selecting from the catalogue of existing rules, as well as adapting them to serve their new purpose if necessary. Due to its verbosity and complex grammar, the normative XML syntax of RIF (RIF/XML) does not offer a convenient platform to carry out these operations. Instead, our approach relies on manipulating rules as RDF resources for easy querying and transformation. Therefore, a bidirectional mapping between RIF/XML trees and RDF graphs has been defined.

The paper is organized as follows. The next Section introduces a (partial) bidirectional mapping between RIF documents and RDF graphs. Section 3 describes how RIF Assembler works. Two usage scenarios are discussed in Section 4. Related work is examined in Section 5 and, finally, concluding remarks are made in Section 6.

2 Moving between RIF and RDF

RIF is a family of languages, called *dialects*, covering different kinds of rules: from logic-programming [10] to production rules [5]. The syntax and semantics of each dialect is rigorously and formally specified, sharing a common core of machinery. Among their shared features, RIF dialects include support for annotations.

In RIF, annotations allow metadata to be attached to almost every syntactic element of the language, from the RIF document itself (top element in the hierarchy) to the terminal symbols of the grammar [4]. No more than one annotation is allowed per element. An annotation has the form $(* \text{id } \varphi *)$, where *id* represents the identifier of the annotated syntactic element (a URI), and φ is a RIF formula that captures the metadata. In particular, φ is a frame (an expression of the form $s[p \rightarrow o]$) or a conjunction of frames (i.e., $\text{And}(s_1[p_1 \rightarrow o_1], \dots, s_n[p_n \rightarrow o_n])$).

The RIF machinery for annotations is very flexible and permits great syntactic freedom. For instance, the identifier (*id*) is not required in general. However, when absent, it is not possible to attach metadata to the element nor can cross-references be made between elements of a RIF document. Thus, our advice to authors of RIF documents is to assign identifiers to at least groups and rules in order to facilitate reuse.

¹ RIF normative interchange syntax is an XML grammar, although for the sake of readability, in this paper we will use the more human-readable RIF Presentation Syntax (PS), defined in the informative part of the specification.

```

Group (
  (* ex:bf-rule ex:bf-rule [
    dc:title    -> "Breast feeding contraindication alert for Nafarelin"
    dc:relation -> <http://www.drugbank.ca/drugs/DB00666>
    dc:source   -> <http://www.sometrustedworthysite.com/>
  ]
  *)
  ?Person[ex:avoid-drug -> <http://www.drugbank.ca/drugs/DB00666>]
  :-
  ?Person # foaf:Person
  ?Person[foaf:gender -> "female"
    ex:status   -> ex:breastFeeding]
)

```

Fig. 1. A RIF rule with annotations

Figure 1 contains an example of a RIF rule in the domain of health care. Note that the metadata make use of existing ontologies and vocabularies, such as Dublin Core, FOAF and domain ontologies.

2.1 From RIF Documents to RDF Graphs

The following paragraphs define a translation from RIF documents to RDF graphs. The translations of the annotations and the representation of the hierarchical structure of rules and groups (groups are also known as rulesets) are treated separately.

Table 1 describes a mapping (π) between RIF metadata expressions (φ) and RDF triples. For the sake of simplicity, base terms in φ are limited to constants (i.e., variables and other terms, such as positional terms, are not permitted). This constraint makes the translation straightforward because both ends use URIs and literals for constants.

It is worth noting the divergence between annotations translated by π and the RDF syntax for RIF suggested by W3C [8]. In our case, semantically-equivalent triples for φ annotations are provided according to [4] (i.e., there is a direct correspondence between a frame and a triple), while [8] describes an RDF-serialization for its frame-based syntax. Although the latter is more expressive and permits representation of complete RIF documents, it is more difficult to query using SPARQL. As RIF Assembler aims to keep things simple, it uses the simpler equivalence of annotations, as suggested by the RIF-in-RDF note.

The mapping π can be used to comprehensively collect all the annotations of multiple RIF documents as a single RDF graph. This makes it possible to execute SPARQL queries against the rules metadata, as well as fostering information reuse based on “linked data” principles. Moreover the graph structure of RDF is a more natural structure than XML trees to represent relationships between rules (e.g., rule A replaces rule B) and rules belonging to multiple rulesets.

Table 1. Interpretation of RIF annotations as RDF graphs (N3 syntax)

Annotation φ	$\pi(\varphi)$
$s [p \rightarrow o]$	$\{ s \ p \ o \}$
$s [p_1 \rightarrow o_1 \ \dots \ p_n \rightarrow o_n]$	$\{ s \ p_1 \ o_1 \ ; \ \dots \ ; \ p_n \ o_n \}$
$\text{And}(F_1, \dots, F_n)$	$\{ \pi(F_1) \} \cup \dots \cup \{ \pi(F_n) \} .$

Table 2. RDF interpretation of RIF hierarchies

RIF PS non-terminal symbol	$\pi(\cdot)$
$\langle \text{Group} \rangle$	$\{ \langle \text{group}_{id} \rangle \ \text{rdf:type} \ \text{rulz:Ruleset} \}$
$\langle \text{RULE} \rangle$	$\{ \langle \text{rule}_{id} \rangle \ \text{rdf:type} \ \text{rulz:Rule} \}$
$\langle \text{Group}' \rangle$ within $\langle \text{Group} \rangle$	$\{ \langle \text{group}_{id} \rangle \ \text{rulz:subset} \ \langle \text{group}'_{id} \rangle \}$
$\langle \text{RULE} \rangle$ within $\langle \text{Group} \rangle$	$\{ \langle \text{rule}_{id} \rangle \ \text{rulz:inRuleset} \ \langle \text{group}_{id} \rangle \}$

In order to represent the structure of RIF documents in RDF graphs, we make use of the *Rulz* vocabulary² to type the resources (rules and groups) and to capture their hierarchy. The mapping between RIF entities and *Rulz* concepts is described in Table 2. The expression $\langle \cdot_{id} \rangle$ denotes the URI that identifies the RIF entity (that is, the *id*). If the entity has no identifier, a fresh blank node is generated and associated with the entity.

2.2 From RDF Graphs to RIF Documents

To translate from RDF back to RIF, the π^{-1} mapping is applied. The hierarchy described by some RDF graphs cannot be mapped by π^{-1} because the data structures are not isomorphic — a graph (RDF) is more general than a tree (XML). To ensure the transformation is feasible, a constraint is introduced: each of the connected subgraphs defined by the edges labelled *rulz:subset* and *rulz:inRuleset*⁻¹ must have a tree-shaped structure. In other words: a ruleset can have no more than one super-ruleset, no cycles can occur within a subset hierarchy and a rule can be part of no more than one ruleset.

The π^{-1} mapping may produce a single tree or many of them. If there are multiple trees, i.e., there are disconnected rulesets and rules, a new root node $\langle \text{Group} \rangle$ is generated to subsume all the structures under a single tree. Moreover, if the output of the mapping is a single rule that is not part of any group, a root node $\langle \text{Group} \rangle$ is also generated, as required by the RIF/XML syntax.

As indicated at the beginning of this section, RIF does not permit metadata to be attached to an entity that lacks an identifier. Note that this restriction

² <http://vocab.derri.ie/rulz>

does not exist in RDF because blank nodes are allowed as the subject of triples. Consequently, if blank nodes are present in the RDF graph, the π^{-1} mapping mints identifiers (URIs) for rules and rulesets to complete the transformation.

3 RIF Assembler

RIF Assembler receives one or more RIF documents as input and produces a single RIF file. The instructions to select the domain rules that will form part of the output and drive their transformation are also provided by rules. These assembly instructions are metarules (i.e., second-order rules). The tool can also read OWL ontologies and RDF datasets, which are taken into account in the metarule reasoning process, but are not part of the output of the system.

RIF Assembler does not create rules from scratch. Any rule in the output must be present in the input (note that the reverse is not necessarily true), although it may be subject to changes during the process. The transformations are limited to its metadata and its location within the structure of the ruleset. More specifically, it is not possible to modify the formulation (and therefore the meaning) of the rule.

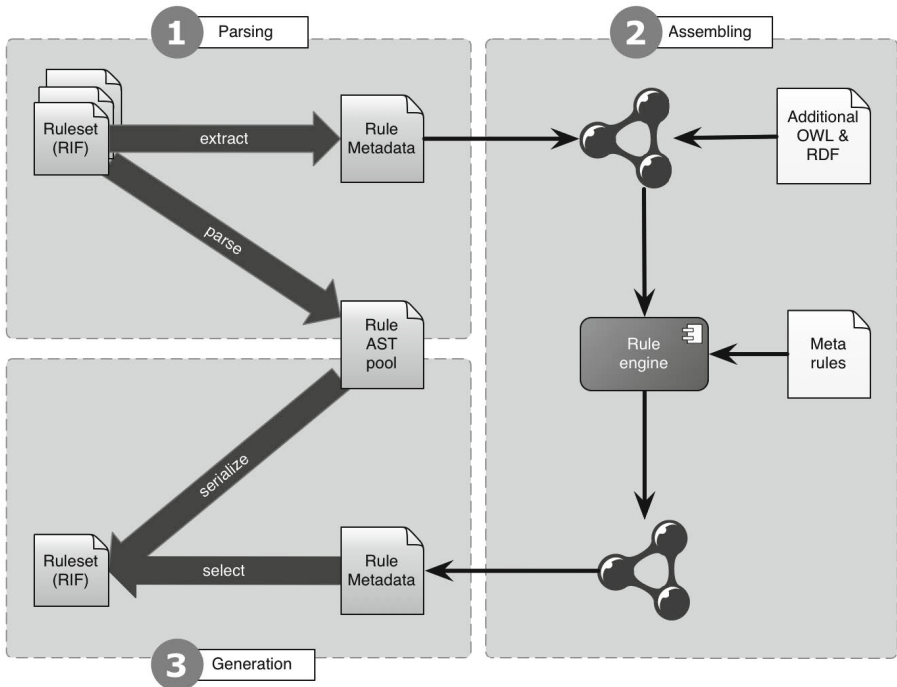


Fig. 2. Architecture of RIF Assembler

RIF Assembler carries out the following steps, as illustrated in Figure 2:

RIF parsing: The input RIF documents are parsed to populate two data structures³. Individual Abstract Syntax Trees (AST) are generated for each rule, and incorporated into a pool. As a consequence, rules are disengaged from the source documents. At the same time, the rule and group metadata, as well as the hierarchical organization of the original documents, are converted to RDF as explained in the previous section. This information is put into an RDF store and merged with other domain knowledge sources, such as OWL ontologies and RDF datasets.

Assembling: In the previous step rules and groups are abstracted from their sources and syntax. This makes it possible to handle them as RDF resources and to manipulate them by simply querying and updating the RDF graph. Metarules specify the conditions and restrictions to be satisfied in order to produce the tailored system. More precisely, metarules are fired to create, modify and delete rule and group metadata; to select and delete rules; or to rearrange the hierarchy by creating/deleting groups and changing rule membership. It is worth noting that domain rules and metarules live in separate universes; at no point do they mix with each other.

RIF generation: The last step of RIF Assembler generates a single AST from the individual ASTs available in the rule pool. The RDF graph is queried to find out which rules and groups are to be included in the output and how they nest. The annotations are also obtained from the RDF graph. The unified AST is then serialized as a RIF/XML document.

This process has been implemented in a web application built upon the Jena Framework⁴. To implement the metarules, the forward chaining engine of the Jena general purpose rule-based reasoner (also known as Jena Rules) is used⁵. Input and output RIF documents use the RIF/XML syntax instead.

A live instance of the application is available at <http://ontorule-project.eu/rifle-web-assembler/>. The user interface is shown in Figure 3 and is divided in three areas:

1. A toolbar to execute the main operations, namely: (a) to export the graph to an RDF/XML file; (b) to execute SPARQL queries; (c) to export the assembled rules to a RIF document; (d) to compute a partial evaluation of the assembly and (e) to reset the application. The partial evaluation is a feature that supports multistage assembly processes.
2. A list of the input documents including: domain rules in RIF/XML, domain knowledge in RDF and OWL files, and metarules in Jena rules. The documents can be uploaded from local files, URIs or by direct input (typing or pasting the text in a form).

³ To parse RIF/XML, we use the RIFle library, available at <http://rifle.sf.net>.

⁴ <http://incubator.apache.org/jena/>

⁵ <http://incubator.apache.org/jena/documentation/inference/>

1 Export to RDF Query with SPARQL Export to RIF Load Assembled RIF Clear model

2 Input Documents

RIF	step1-rules-pool.rif	http://ontorule-project.eu/resources/assembler/step1-rules-pool.rif	View document
OWL	process-ontology-and-facts.owl	http://ontorule-project.eu/resources/assembler/process-ontology-and-facts.owl	View document
JR	step1-assembling-rulesets.jenarules	http://ontorule-project.eu/resources/assembler/step1-assembling-rulesets.jenarules	View document

Upload by:

3 Statistics

Input RIF rulesets	1
Input RIF rules	4
Input RIF triples	43
Input metarules	2
Input Extra triples	109
Assembled triples	167
Assembled RIF rulesets	4
Assembled RIF rules	3

Fig. 3. User interface of RIF Assembler tool

3. The panel at the right displays statistics about the information loaded in the system and the resulting model after applying the metarules. For instance, in Figure 3, 4 domain rules have been loaded from one RIF document, but only 3 rules remain after the execution of the metarules.

Finally, RIF Assembler makes use of Parrot, a RIF and OWL documentation tool [16], in order to display the combinations of ontologies and rules in the input files and the final assembled ruleset.

4 Usage Scenarios

Two usage scenarios are presented in this section. The first one is a simplistic, imaginary example in the health care domain, and illustrates the concept of rule mashups. The second one is a realistic and more sophisticated scenario related to knowledge reuse within ArcelorMittal, the world’s largest steel producer. More details about the latter scenario can be found at [7].

4.1 Health Care Scenario

Nowadays, drugs are shipped with Patient Information Leaflets (PIL) which contain essential information about the product. The structure of a PIL, e.g., “list of excipients”, “contra-indications”, and “use during pregnancy and lactation”, is defined by regulations, for instance, European Directive 2001/83/EC.

One can imagine that, in the near future, pharmaceutical companies will publish this information on the web (open data). Although some information, such as

the list of excipients, may be modeled and made available as RDF graphs using vocabularies such as SNOMED⁶, other information, such as contra-indications and interactions with other medicinal products, may require using RIF rules. As an example, Figure 11 contains a rule that alerts breast-feeding women not to take a particular drug.

In this hypothetical scenario, RIF Assembler may be used to process the rules harvested from the web. As not all the sources are equally trustworthy, RIF Assembler may execute metarules that decide which rules are reliable, taking into account, for instance, provenance information contained in annotations. An example of these metarules would be: “keep all the rules that come from a source whitelisted by the World Health Organization”. The output of the assembly process would be a ruleset potentially usable for making decisions on treatments and prescriptions. Such a system would support professionals in medicine and be the foundation for personal software assistants (virtual doctors).

4.2 Steel Industry Scenario

This real-world scenario focuses on the quality control system of a galvanization line of the ArcelorMittal steel mill. After a steel coil has been galvanized, the product quality is assessed by examining data measured by sensors all along the production line. Three alternative outcomes are possible: if the coil meets the quality requirements from the order, it is shipped to the customer; if the coil presents some non-critical defects, it is sent to repair; finally, if the product quality is critically low, it is sent for scrapping.

As one of the world’s leaders in the steel industry, ArcelorMittal operates several factories all over the globe. Ideally, the rules that implement company business policies should be usable wherever a factory carries out the galvanization process. Actually, the situation is more complex; factories are not identical clones. Steel mills are equipped with diverse machinery in terms of precision, age, maintenance, operation-cost and lifespan. Moreover, although rule-based systems are widely adopted within the company, different rule engines and rulesets are used at each location, for historic reasons as well as local specifics. The company maintains a pool of shared knowledge that is populated by the artifacts already deployed within the company. Thanks to RIF and RIF Assembler, rules can be drawn from this pool and packaged into rulesets tailored for a given factory. More precisely, RIF Assembler, driven by the metarules, builds a system that assesses the quality of galvanized coils for a particular facility, namely, the factory located in Avilés, Spain.

In a preliminary step, rules in the pool were annotated with provenance information (e.g., the factory they come from) and regarding the part of the business process they carry out (or *realize*, in the vocabulary of this scenario). To this end, we have created a domain ontology that models industrial processes, such as galvanization, and their associated quality processes. In essence, a business process is split into a sequence of tasks. The `implements` property captures the

⁶ <http://www.snomed.org/>

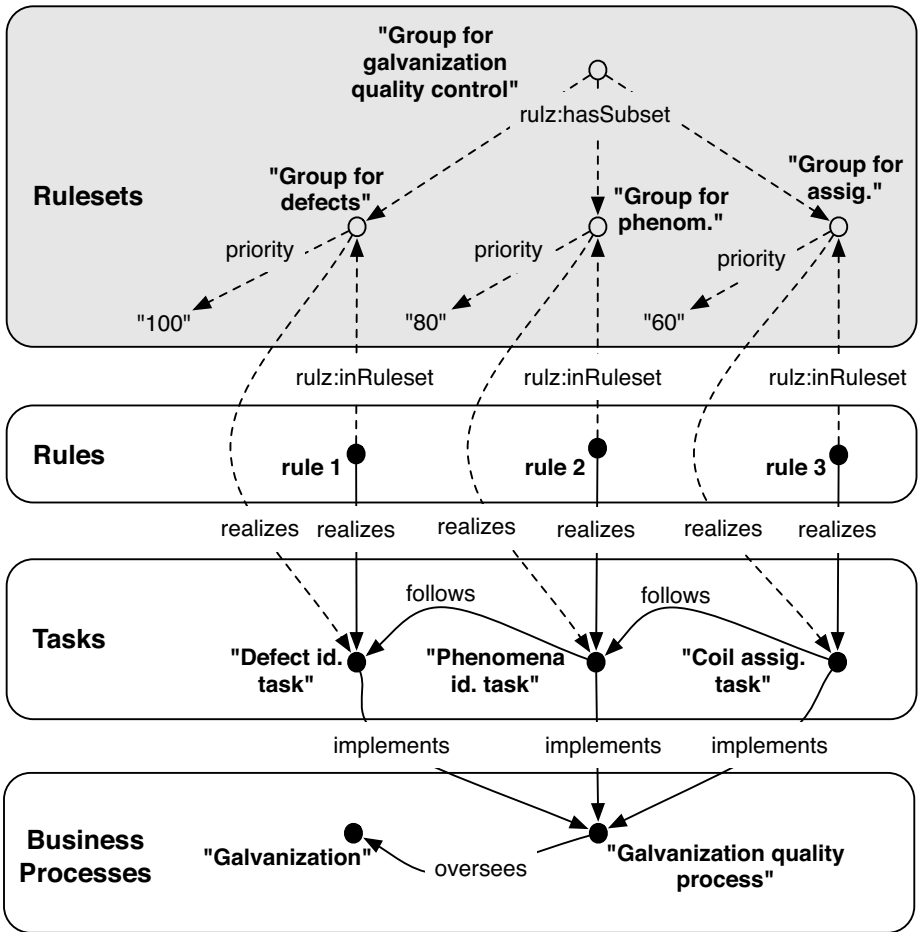


Fig. 4. RDF graph representing the rules and rulesets of the steel industry scenario

relation between tasks and business processes. The `follows` property defines total order within sets of tasks. The `realizes` property is used in rules and ruleset annotations to connect them to the tasks (see Figure 4). Using this ontology, we can express the fact that “The group to which Rule 1 belongs realizes the defect identification task that implements the galvanization quality process”.

Rules are drawn from the shared pool and their metadata are augmented and refined with specific knowledge borrowed from other ArcelorMittal facilities. Metarules instruct RIF Assembler on how to manipulate and organize the input rules. These assembly instructions are dictated by business experts in the domain who are aware of the specifics throughout the production line. In the case of the Avilés galvanization line, the metarules carry out three activities:

1. Creating a generic quality system from the rule pool. Metarules such as the one in Figure 5 select the rules that are relevant to the galvanization quality

process and put them in a group as depicted in Figure 4. Rules from the pool that are not relevant are simply discarded. Another metarule assigns priorities to each group based on the precedence relation between tasks.

2. Augmenting the system with additional rules from similar factories. Some metarules determine which business policies in use in other facilities may also be applied in Avilés, even if the associated rules are not part of the generic quality system. These rules are simply added to the final ruleset.
3. Refining the system by attending to the specifics of the galvanization process in Avilés. In some cases, rules from another factory may also be added, replacing rules from the generic quality system. This is the case with rules related to electrogalvanization, which is a refinement of the generic galvanization process and is available only at certain factories.

Although not used in this scenario, RIF Assembler may also exploit OWL inference, particularly the OWL-RL profile [12], to augment the expressivity of metarules. This opens the door to handling ontologies with complex hierarchies, such as those that describe business processes beyond the simple case addressed in this scenario. Another practical application of reasoning within RIF Assembler is to combine rules that are annotated with respect to different ontologies, and that consequently require alignment.

Figure 4 depicts the RDF graph at an intermediate stage of the assembly process. The solid lines represent annotations of rules and statements from the ontology that is provided as input. The dashed lines indicate inferences derived by the metarules. All the ruleset resources and their hierarchy were created by the metarules.

```
[R1:
  (?rule rdf:type rulz:Rule)
  (?rule bp:realizes ?task)
  (?rule rulz:inRuleset ?group)
  (?rule bp:factory bp:Pool)
  (?task bp:implements bp:QualityGalvanizationProcess)
  (?task rdfs:label ?taskName)
  strConcat("Autogenerated ruleset for task ",?taskName,?rulesetName)
  makeTemp(?newGroup)
->
  remove(2)
  (?newGroup rdf:type rulz:Ruleset)
  (?newGroup rdfs:label ?rulesetName)
  (?newGroup bp:realizes ?task)
  (?newGroup bp:scope bp:GalvanizationQualityProcess)
  (?newGroup bp:factory bp:FactoryInAviles)
  (?rule rulz:inRuleset ?newGroup)
]
```

Fig. 5. Metarule that creates new groups for rules from the pool that realize a relevant business task

5 Related Work

Business Rule Management Systems (BRMS) evolve from (production) rule engines to cope with other requirements beyond the execution of rules. One of the key features of modern BRMSs is the rule repository [9], where artifacts and their metadata are stored. Rule metadata are particularly important in the last step of the BRMS lifecycle: maintenance of the rule-based application [13]. This final step involves knowledge reuse and adaptation to changes in the application context. Leading BRMS products feature some mechanism to extract rules, i.e., generate rulesets that contain subsets of the complete knowledge base, by specifying conditions applied to rule metadata [3]. This is the case of IBM WebSphere ILOG JRules, which permits the execution of queries against rule metadata. Improving on this feature was one of the motivations for our work on knowledge reusability. In particular, RIF Assembler extends the query functionality with the execution of metarules, permitting not only selection, but also modification of the ruleset structure.

The Object Management Group⁷(OMG) introduced the Model-driven Architecture (MDA) paradigm [15], which aims to model real systems using standards such as UML, MOF or XMI. The models defined with these standards remain abstract, and actual implementations can be obtained via automatic code generation. Although the topic of knowledge reuse is shared with RIF Assembler, different drivers motivate these approaches: MDA is model-centric while RIF Assembler is rule-centric.

SPIN [11] is a W3C Member Submission that proposes an RDF syntax for SPARQL queries. Some of these queries, namely CONSTRUCT and SPARUL queries, may express rules [14]. Therefore this work and SPIN share the idea that rules can be represented by RDF resources. This permits the construction of hybrid models that combine the model (ontology) and the queries, and where queries can modify the model itself. We chose to build on RIF and not on SPARQL/SPIN, because the former covers a wider range of rule languages. Thus, RIF Assembler can be used with any BRMS that supports the RIF standard, while using SPIN would require translation from different rule languages to SPIN.

XSPARQL provides a language to transform between RDF and XML [1]. It is conceivable to use it to generate RIF/XML from SPARQL queries. In this sense, it can go beyond what it is currently possible with RIF Assembler. However there is a price to pay for this flexibility: while RIF Assembler only requires rule writing skills, which is an ability that is presumed in the target user community, XSPARQL requires technical knowledge of the RIF/XML syntax, the RDF model and the SPARQL query language.

There are similarities between our work and metaprogramming, i.e., programs that generate other programs [17]. RIF Assembler can be seen as metaprogramming where both the final program and the metaprogram are expressed in rules (RIF and Jena Rules, respectively). However, RIF Assembler does not alter the formulation of the rules, and therefore it is limited with respect to the

⁷ <http://www.omg.org/>

expressivity of the output programs (rulesets). In the future, we plan to fully translate the rules to RDF in order to enable manipulation of the rule syntactic structure. Moreover, at this stage we do not fully exploit the ability of rules to check the consistency of the output.

6 Conclusions

The vision of RIF Assembler relies on an appealing idea, namely, that rules can be reified as RDF resources and treated as first-class web citizens. By doing so, the doors are opened to rules linking to arbitrary resources in the web of data, and vice versa. Many applications may exploit this idea, such as rule search engines and rule-based personal assistants in the areas of ambient intelligence and eHealth.

In this paper, two scenarios give insight into the potential of RIF Assembler. We show that, assuming that annotated rules are available, it is possible to derive mashup rulesets by simply writing down assembly instructions as metarules. These metarules can be inspired by domain experts, dramatically simplifying the construction of families of decision-support systems with respect to previous, manual approaches.

Knowledge reuse, in particular the reuse of rules, is of critical importance to any organization. We envision that the functionality of RIF Assembler may eventually be an integral part of future BRMS products. As the availability of rules increases on the web and in corporate environments, fostered by the adoption of RIF, reuse will become easier. However, RIF Assembler goes beyond the pure exchange of rules. It proposes that rules can be mixed and manipulated independently of their source. For instance, given two rule-based systems *A* and *B*, respectively developed with IBM JRules and JBoss Drools (both Java-based environments), their rules can be exported to RIF. This permits the use of RIF Assembler to select subsets of *A* and *B* to create a new system (described in RIF), *C*, that can be translated to JRules, Drools or any other execution environment, such as the C-based CLIPS. RIF Assembler supports scenarios where reuse implies more than the mere portability of previously-built solutions, but also rearrangement of knowledge in order to meet new requirements and contexts of use, as in the ArcelorMittal scenario.

RIF Assembler does not yet analyze the contents and semantics of the rules. Therefore, it is difficult to detect and handle contradictions between the rules. Similarly, RIF Assembler does not provide automated consistency checks of the assembled ruleset. It is up to the user to decide and implement metarules to deal with rulesets that do not merge seamlessly. Nevertheless, the tool provides some help in this task. For instance, it makes it easy to replace a troublesome rule with a better alternative. The authors will further improve in this direction following the findings made by the ONTORULE project on static rule consistency checking [6].

Acknowledgements. This work has been partially supported by the European Commission under ONTORULE Project (FP7-ICT-2008-3, project reference 231875).

References

1. Akhtar, W., Kopecký, J., Krennwallner, T., Polleres, A.: XSPARQL: Traveling between the XML and RDF Worlds – and Avoiding the XSLT Pilgrimage. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 432–447. Springer, Heidelberg (2008)
2. Boley, H., Kifer, M.: RIF overview. W3C Working Group Note, W3C (June 2010), <http://www.w3.org/TR/rif-overview/>
3. Boyer, J., Mili, H.: Agile Business Rule Development - Process, Architecture, and JRules Examples. Springer (2011)
4. de Bruijn, J.: RIF RDF and OWL Compatibility. Recommendation, W3C (June 2010), <http://www.w3.org/TR/rif-rdf-owl/>
5. de Sainte Marie, C., Hallmark, G., Paschke, A.: RIF Production Rule Dialect. Recommendation, W3C (June 2010), <http://www.w3.org/TR/rif-prd/>
6. Fink, M.: D2.6 consistency maintenance final report. Deliverable, ONTORULE project (2011)
7. González-Moriyón, G., Polo, L., Berrueta, D., Tejo-Alonso, C.: D5.5 final steel industry public demonstrators. Deliverable, ONTORULE project (2011)
8. Hawke, S., Polleres, A.: RIF In RDF. Working Group Note, W3C (May 2011)
9. Herbst, H., Myrach, T.: A repository system for business rules. In: Mark (ed.) Database Application Semantics, pp. 119–138. Chapman & Hall, London (1997)
10. Kifer, M., Boley, H.: RIF Basic Logic Dialect. Recommendation, W3C (June 2010), <http://www.w3.org/TR/rif-blld/>
11. Knublauch, H., Hendler, J.A., Idehen, K.: SPIN - Overview and Motivation. Member Submission, W3C (February 2011)
12. Motik, B., Fokoue, A., Horrocks, I., Wu, Z., Lutz, C., Grau, B.C.: OWL 2 web ontology language profiles. W3C recommendation, W3C (October 2009), <http://www.w3.org/TR/2009/REC-owl2-profiles-20091027/>
13. Nelson, M., Rariden, R., Sen, R.: A lifecycle approach towards business rules management. In: Proceedings of the 41st Annual Hawaii International Conference on System Sciences, pp. 113–113 (January 2008)
14. Polleres, A.: From sparql to rules (and back). In: Proceedings of the 16th International Conference on World Wide Web WWW 2007, p. 787 (2007)
15. Poole, J.D.: OMG Model-Driven Architecture Home Page (2001), <http://www.omg.org/mda/index.htm>
16. Tejo-Alonso, C., Berrueta, D., Polo, L., Fernández, S.: Metadata for Web Ontologies and Rules: Current Practices and Perspectives. In: García-Barriocanal, E., Cebeci, Z., Okur, M.C., Öztürk, A. (eds.) MTSR 2011. CCIS, vol. 240, pp. 56–67. Springer, Heidelberg (2011)
17. Visser, E.: Meta-programming with Concrete Object Syntax. In: Batory, D., Conzel, C., Taha, W. (eds.) GPCE 2002. LNCS, vol. 2487, pp. 299–315. Springer, Heidelberg (2002)

Product Customization as Linked Data

Edouard Chevalier and François-Paul Servant

Renault SA

13 avenue Paul Langevin

92359 Plessis Robinson, France

{edouard.chevalier, francois-paul.servant}@renault.com

Abstract. Ranges of customizable products are huge and complex, because of the number of features and options a customer can choose from, and the many constraints that exist between them. It could hinder the publishing of customizable product data on the web of e-business data, because constraints are not tractable by agents lacking reasoning capabilities. But the configuration process, which helps a customer to make her choice, one step at a time, is a traversal of a graph of partially defined products - that is, Linked Data. Reasoning being hosted on the server, its complexity is hidden from clients. This results in a generic configuration API, in use at Renault. As configurations can be completed to valid commercial offers, the corresponding ontology fits nicely with GoodRelations. Benefits in e-business related use cases are presented: sharing configurations between media, devices and applications, range comparison based on customer's interests, ads, SEO.

Keywords: configuration, customizable product, Linked Data, GoodRelations, automotive.

1 Introduction and Motivation

Publishing product data to improve e-business performance and visibility on the web through Search Engine Optimization (SEO) is gaining momentum, thanks to vocabularies such as GoodRelations [1], and the Schema.org [2] initiative.

An increasing number of manufacturers and vendors have been trying to make their websites usable by agents and client applications, and accurately searchable by search engines. This has produced interesting results with products such as books or music, for instance: search results listing actual products - not web pages - and including links to commercial offers with price, ratings, etc.

But suppose you are looking for, say, a small car, with a gasoline engine, a sun-roof, air conditioning, and an adapter to connect your MP3 player: how nice if you could just type that in your search engine. Ah, and you're concerned with price, and you would like to compare CO2 emissions; well, your search engine probably won't help there. You might find some dedicated sites to provide comparisons between cars, but nothing precise enough to allow the exact kind

¹ <http://schema.org/>

of comparison you want: price and CO2 emissions of small gasoline cars with sun-roof, etc.

Hardly surprising: the data such applications require is not available yet. Are manufacturers reluctant to open up their data about their ranges? If they are, they will have to change as more daring competitors enter the game and begin publishing their data: the cost of not appearing at all in the results of searches made by potential customers would be overwhelming, particularly if such search results easily convert to purchase orders.

On the other hand, it should be noted that cars are more complicated to handle than books: books are searched on the basis of a very small set of properties (title, author,...); they are well identified (e.g. through ISBN); and comparisons between commercial offers only involve completely defined products. Whereas cars are customizable, a crucial aspect of the problem: rather than fully specified products, what you compare are sets of them, that is, partially defined products.

In industries practicing “Build to Order” of fully customizable products, ranges are huge, because of the number of features and options a customer can choose from: more than 10^{20} different cars are for sale at Renault, and 30 to 40 decisions are needed to completely differentiate one from them. Those ranges are not only huge, they are also complex, because of the many constraints between features which invalidate some of their combinations: if every combination of distinctive features and options were possible, there would be 10^{25} different Renault cars, not our mere 10^{20} - meaning you have only one chance upon 100,000 to define an existing Renault car, if you choose its specifications without taking the constraints into account.

Specifying those product ranges requires the use of a vocabulary able to represent the constraints. This can be done by means of Semantic Web languages [2], but using this data in practical applications requires sophisticated automatic reasoning to handle the constraints. Publishing such range definitions on the web clearly won't bring many practical results soon, as one cannot expect strong reasoning capabilities from client agents.

Though difficult to specify and hard to manipulate, these product ranges are nevertheless described rather effectively, for human users, by means of dedicated web applications called configurators. A configurator helps a user interactively define a product step by step, each step describing a valid partially defined product (PDP), with a start price and a list of remaining choices given all previous selections. Each of these choices links to another PDP until completion. Thus, the configuration process traverses a graph whose nodes are PDPs. Now identify each PDP with a URI returning the list of the PDPs it is linked to, among other relevant information: what you have is a description of the range as Linked Data. This is how a configuration engine can publish descriptions of complex products on the web of data, which agents without reasoning capabilities can effectively understand.

This should be of some benefit to any configurator application, whatever its actual implementation. To name but one - the easy sharing of PDPs between

applications, devices, and media such as social networks is an effective way by itself to increase visibility on the web.

Renault began to publish data about its range in this way, after implementing a Linked Data based configuration API on top of its configuration engine, a deductively complete reasoner allowing configuration in free order. Any valid combination of choices can therefore be handled as a PDP, and be published on the web of data. Agents can easily crawl this data.

This document is structured as follows. We begin with an overview of the configuration process. In section 3, we show how it can be modeled as Linked Data which provides for an easy to use configuration API. We describe its implementation and use at Renault in section 4. A GoodRelations compliant configuration ontology is proposed in section 5. Finally we discuss the benefits of the solution: a clean system architecture, reduced development costs for client applications, improved sharing of configuration-related information. This opens the way to novel applications, including e-business related ones.

2 Product Range Specification and Configuration

2.1 Product Range Specification

Because the set of different products that a customer can specify and order is too huge to be enumerated, ranges of customizable products are defined in intention.

The specification of a family of similar products (typically those of the same “model”) is based on a “lexicon”, i.e., a set of variables representing the relevant descriptive attributes: body type, type of fuel, color, etc. In a completely defined product, any of these variables is assigned one value and one only. Such a value is called a “specification” in ISO-10303 STEP AP 214 terminology, a term that we’ll use throughout this paper. In the Renault range, the variables are discrete: any of them, e.g. the type of fuel, has a finite list of possible values, e.g. gasoline, diesel, electric, gasoline-electric hybrid.

Then a set of constraints restricts the possible combinations of specifications.

The Product Range Specification (PRS) is therefore a Constraint Satisfaction Problem (CSP), and the many PRS related questions that have to be answered in the day to day operation of business are computationally hard (SAT being NP-complete). Renault has developed tools, based on a compiled representation of this CSP. The computationally hard part of the problem is fully solved in an offline phase, guaranteeing bounded and fast response times for most of the queries: for configuration queries, time is linear on the size of the compiled representation, which happens to remain small enough [3].

2.2 Presenting a Range of Customizable Products to Customers

A configurator application is the main way of presenting a complex range of customizable products to customers. It is a decision support tool that guides users to desirable - and valid - product configurations. Most web sites of automotive

constructors give access to a configurator application, often through a link entitled “build your own car” - although it will not actually be built, but chosen from a huge set.

Most of the configurator applications help a user interactively define a product step by step, each step describing a valid partially defined product, in the sense that it can be completed, without changing any of the current selections, into an existing fully specified product, which can be ordered. We’ll call “Configuration” any such valid, partially defined product: in other words, any state of the configuration process.

Note: A configurator application may conceivably have to handle “invalid configurations”, that is, combinations of specifications that are impossible. This can happen, for instance, if the user is allowed to begin the configuration process by choosing features without any control of their compatibility; or if she is allowed to choose a feature incompatible with her previous selections. In this case, it is the responsibility of the configurator application to restore the consistency of the configuration, necessarily by excluding some of the previous user selections. For us, the word “Configuration” excludes such invalid combinations.

2.3 Features of a Configuration Engine

We list here the features that we think are necessary in a good configurator. Not all of them may be available in the implementations we see on the web, either because the software supporting the configuration process (the configuration engine) is not able to provide them, or because of a poor application design, which tends to stick to the old way of selling products, typically imposing a predefined order on the user to make her choices, which simplified the handling of the configuration process, at the expense of user comfort.

The main point is that the configuration engine should guarantee completeness of inference, that is, every consequence logically entailed by a given state in the configuration process gets actually inferred when in that state, and not later [3]. This is absolutely necessary if users are to freely choose from specifications compatible with their previous selections, and to be barred from making choices no valid configuration satisfies; in other words, if the whole range is to be made accessible to them, without their ever having to backtrack from dead-ends.

Here a list of desirable features in a functional perspective:

- free choice order
- pricing information, possibility to filter by a max price
- negative choices (configuring by excluding specification)
- permissiveness and conflict resolution: users are allowed to change their minds about previous selections, or to select a specification excluded by them (in the latter case, the system should advise on which choices to change in order to restore a valid configuration).
- completion: providing a completely defined product matching the configuration.

Configuration of the Model. Frequently, each of the distinct models in a given product range is described with its own lexicon. This imposes the choice of a model as the first step in the detailed configuration of the product. This required first step is a nuisance - a customer may be hesitating between two similar models, and would need more information to make her choice. As a partial remedy, we can devise a set of variables shared by all the models, e. g. the body and engine type, the CO2 emission level, etc.

Another way to alleviate this problem is by allowing textual searches in the whole union set of specifications over the different models, with the configuration engine checking whether the conjunction of the found specifications matches existing configurations (see how-to in [3.3](#)).

2.4 Related Work

Product configuration is an open and active field of research, with an important community and lots of publications. But this paper is not about reasoning and the way it is implemented. In fact, one of the main points is precisely about hiding the complexity of reasoning from clients.

The exact context of this work is the borderline where product configuration meets e-Commerce applications of Linked Data. The main contribution in the same field that we know of is Volkswagen's "Car Option Ontology"². Their approach is different: they publish the constraints, in a proprietary vocabulary. We think that such data cannot be effectively used by simple agents, because they do not have reasoning capabilities. Instead, we host the reasoning on the server, and free clients from the burden, ensuring maximal usability of the data that we provide.

3 Configuration API

The configuration process can be modeled as Linked Data. This provides the basis for a simple, yet generic, Configuration API.

3.1 Configuration as Linked Data

In a typical configuration application, the user is presented with successive choices in a way that she cannot choose incompatible specifications. Each configuration, that is each successive state of the configuration process, is characterized by the specifications selected so far. A configuration engine is responsible for ensuring that only valid choices are presented at each step.

Such an application can therefore be implemented as a GUI over a REST service which takes the description of a valid configuration (the list of the specifications already selected) as input parameter, something like :

`configService?chosenSpec=spec1&chosenSpec=spec2&...` (1)

² <http://purl.org/coo/ns>

and returns the next list of specifications to be chosen from, all guaranteed to be compatible with the input. Choosing one of them is then just a matter of adding it to the list of the “chosenSpec” query parameters and of getting the updated state of the configuration process.

Note that a query such as [\(II\)](#) identifies a configuration, and can be used as a URI for the configuration in question; or, more precisely, redirect to an actual URI of it; therefore, we can improve the service by making it return the URI of the linked configuration along with each compatible specification: the representation of the configuration resource then contains a list of couples (compatible specification, linked Configuration).

Such a service makes it easy to implement a configurator application: accessing a configuration URI returns the data needed to build the corresponding web page: basically a list of links to the next configurations. Every configurator application on the web could be (re-)implemented this way: it is just a matter of wrapping the configuration engine in a REST service that provides the data needed to generate the HTML

When implementing such an application, play with HTTP content negotiation, in quite classical Linked Data style, to respond either with data or with a HTML page to a given configuration URI; either a page built from the data, or the unadulterated data themselves. In the HTML page, include the data as RDFa or microdata markup; stating in particular that the page describes the Configuration.

3.2 Querying

The Linked Data based API allows to crawl the range, starting from the root of the service (the “empty configuration”). Only valid configurations are returned.

It is useful to also provide a way to query the dataset. The template of the service [\(II\)](#) can be used as a simple querying API. Mind however that any combination of specifications may not be valid. The service should detect such invalid conjunctions and return a 404 Not found HTTP error. Only configuration engines that support free order can provide such functionality in every circumstance.

It would be tempting to query the dataset using SPARQL syntax:

```
SELECT ?conf WHERE { ?conf :chosenSpec :spec1 , :spec2 . }
```

but according to SPARQL semantics, this should return all the configurations with spec1 and spec2 - possibly several billions of them. Instead we would not expect a list of configurations here, but one only - or zero if spec1 and spec2 are not compatible. It is feasible to implement the intended semantics with SPARQL, but the syntax is a bit cumbersome, therefore far less attractive. We therefore didn't implement a SPARQL endpoint.

3.3 Use Case: Implementing Text Search

With these two services, we can implement text based searching in the configuration space. Here is a sketch of a solution, assuming the configuration engine

supports free choice order. The configuration corresponding to a car model links to the specifications compatible with that model; now, using a text search engine tool such as Lucene, index the (model, specification) pairs with the text form of the specification as the index key. Then, searching for “air conditioning, sun roof, MP3” (say) will get you a list of (model, specification) pairs; making the configuration service conjoin the car model and relevant specifications will get you configurations matching your text search. This adapts to the case where the configuration engine only supports free order only after some main choices have been made; for instance, if choices for car model, engine and level of equipment are required before allowing all options to be chosen in free order.

4 Renault Implementation

Traditionally, we have been providing access to the functionalities of our configuration engine through a java API. Recent plans for important changes in the Renault web site sparked an opportunity to provide a Linked Data based access.

The definition of the current commercial offer is managed by upstream systems, then compiled into the binary data used by the configuration engine (size: <100MB). It is published by means of a REST service, such as described above: Linked Data is materialized on the fly when PDPs are queried (30 KB per PDP). When the definition of the range is updated, part of the knowledge base used by the configuration engine is replaced. URIs of PDPs include the release number of the knowledge base, so all previous URIs are “deprecated” - but they still can be queried by clients: an HTTP 301 redirects to the new URI, if the PDP still exists in the range. A 404 is returned otherwise. In the latter case, the service can be re-queried to get a “similar” product.

The implementation of the service uses Jersey³ (the reference implementation of JAX-RS⁴). As of this writing (February 2012) only JSON data are returned, and only for German and Italian markets.⁵ All functionalities of our configuration engine are made accessible through the JSON data, including querying in free order, maximum price, conflict resolution, completion, etc., (optional query parameters being added to the configuration URIs to implement some of them).

A cursory look at the data⁵ may convey the impression a configuration URI does not contain the list of chosen specifications which defines the configuration. It does, though; only encoded in a short form. For we anticipated configurations would be shared on Twitter; using an URL shortener or an internal index might bring down performance as vast numbers of configuration URIs are generated: 100-300 to represent but one configuration. Indeed, many links are included since we provide free order of choices.

Regarding performances, the HTTP response time for accessing one configuration is around 20-30 milliseconds.

³ <http://jersey.java.net/>

⁴ <http://jcp.org/en/jsr/detail?id=311>

⁵ <http://co.rplug.renault.{de,it}/docs>

5 Configuration Ontology

This simple ontology⁶ describes the classes and properties involved in the modeling of the configuration process as Linked Data.

As a partially defined product whose completion to a valid product always exists, a configuration can be seamlessly described in the GoodRelations ontology framework and can participate in the web of data for e-business. This ontology is generic, that is, applicable to any kind of customizable product: it does not depend on the set of variables and specifications with which a given product is defined.

Examples. In the following, we use examples about a very simple range of cars, all of the same model called Model1. The lexicon contains four variables:

- Fuel Type: {Gasoline, Diesel}
- Temperature Control: {Heating, AirCond}
- Radio Type: {NoAudio, SimpleRadio, RadioMP3}
- Roof: {NormalRoof, SunRoof}.

The product range specification is defined in Tab. 1, by the list of specifications available on the three levels of equipment. The total number of different completely defined cars is 8.

Table 1. Product Range Specification example

	Fuel Type	Temperature Control	Radio Type	Roof
Low-end	{Gasoline,Diesel}	{Heating}	{NoRadio}	{NormalRoof}
Mid-range	{Gasoline,Diesel}	{AirCond}	{SimpleRadio}	{NormalRoof}
High-end	{Gasoline,Diesel}	{AirCond}	{RadioMP3}	{NormalRoof,SunRoof}

Notations. The RDF examples are written in Turtle syntax, using the prefix “co” for this configuration ontology, “gr” for GoodRelations, “vso” for the Vehicle Sales Ontology and “r” for the specifications.

5.1 Main Classes

co:Specification. Specifications are first class objects, identified by URIs. This is very natural in most cases, as the specifications correspond to “real world objects”: a fuel type, a radio system, etc. This can also be handy in cases where literal values would appear to be enough at first sight; e. g., where variables are given values from physical ranges, such as widths, or CO2 emission levels. In such cases, modeling them requires more information than their mere basic type; e.g. a unit.

In GoodRelations parlance, a specification is an instance of gr:Qualitative-Value (vso:FeatureValue in the case of vehicles).

⁶ <http://purl.org/configurationontology>

co:Configuration. This is the main class, of course. A Configuration is a state in the configuration process. It is defined by a list of choices: cf. the `co:definingChoice` property. As explained in section 3 implementing the configuration process as Linked Data is based on listing the specifications compatible with a given Configuration, along with the configurations they link to. This is modeled through the `co:ConfigurationLink` class, and the `co:possible` property.

5.2 Definition of a Configuration

A Configuration is defined by the choices the user made (and the definition of the range): primarily, the selection of specifications. Other kind of choices, not directly involving specifications, may be allowed by the configuration engine: for instance, a user can set a maximal price (“a car that costs less than 10.000 euros”), or a maximal delivery time (“a car that I can get within one month”).

co:definingChoice. Parent to all properties specifying the choices that define the Configuration: a Configuration is defined by the list of triples it is the subject of, and which have a `co:definingChoice` as their predicates.

co:chosenSpec. A SubProperty of `co:definingChoice` listing the specifications selected by the user:

```
ex:Conf1 a co:Configuration ;
        co:chosenSpec r:Model1, r:SimpleRadio .
```

Now say you want a radio, but you do not care what kind it is. Because a configuration engine may support choices such as `r:SimpleRadio` OR `r:RadioMP3`, if two or more of the `co:chosenSpec` of a Configuration correspond to the same variable, by convention they are to be interpreted as ORed (even XORED, by the way).

```
ex:Conf2 a co:Configuration ;
        co:chosenSpec r:Model1, r:SimpleRadio, r:RadioMP3 .
```

This means that the car has either a `r:SimpleRadio`, or a `r:RadioMP3`, not both.

Choice order. Choices are made one at a time and in a given order, which may matter. Of course it doesn’t impact the characteristics of the product in any way, but it can be used by the application, for instance to display a textual description of the configuration. This could be achieved with an additional `co:choiceSeq` property having `rdf:Seq` as its range.

co:maxPriceChoice. A subProperty of `co:definingChoice`, an upper limit set on the price of the configuration.

5.3 Description of a Configuration

Given the `co:definingChoice(s)` of a Configuration, some specifications are implied (included in any completely defined product matching the configuration),

some are impossible (they can no more be chosen), others are simply compatible: they still can be chosen among several alternatives. Given the co:defining-Choice(s) of a Configuration, some specifications are implied, i. e., included in any completely defined product matching the configuration, some are impossible, i. e. they can no more be chosen, others are simply compatible: they can still be chosen from among several alternatives.

co:impliedSpec. Given the constraints between the specifications of our range, r:AirCond is implied on ex:Conf1:

```
ex:Conf1 co:impliedSpec r:AirCond.
```

co:possible and co:ConfigurationLink. On ex:Conf1, we still can choose the fuel type:

```
ex:Conf1 co:possible
  [ a co:ConfigurationLink ;
    co:specToBeAdded r:Diesel ;
    co:linkedConf ex:Conf1PlusDiesel .],
  [ a co:ConfigurationLink ;
    co:specToBeAdded r:Gasoline ;
    co:linkedConf ex:Conf1PlusGasoline .].
```

Here is one of the linked configurations:

```
ex:Conf1PlusDiesel a co:Configuration ;
  co:chosenSpec r:Model1, r:SimpleRadio, r:Diesel.
```

HTML display of a co:ConfigurationLink: it corresponds to an hypertext link, whose href is the value of the co:linkedConf property. As for the text of this link, the rdfs:label of the co:specToBeAdded value is quite adequate. It can be directly included in the RDF as a rdfs:label of the co: ConfigurationLink:

```
ex:Conf1 co:possible [ a co:ConfigurationLink ;
  co:specToBeAdded r:Diesel : rdfs:label "Diesel";
  co:linkedConf ex:Conf1PlusDiesel .]
```

Proposing the selection of several specifications at once. Let us note that this model supports the selection of several specifications at once. This can be useful from a marketing point of view, as an emphasis on certain packs of specifications, or on certain full featured configurations:

```
ex:Conf3
  co:chosenSpec r:Model1;
  co:possible [ a co:ConfigurationLink ;
    rdfs:label "Over-equipped configuration!" ;
    co:specToBeAdded r:AirCond, r:RadioMP3, r:SunRoof ;
    co:linkedConf ex:overEquippedConf .].
```

co:alternative. A user may want to change one of its previous selections. This property lists those of the `co:chosenSpec`, which can be changed: it links the configuration to a similar one, with one of the `co:chosenSpec` removed or changed. This property may not be used when the chosen specification in question happens to be implied by the other choices. For instance, on `ex:Conf1PlusDiesel`, the `r:Diesel` can be replaced by `r:Gasoline`:

```
ex:Conf1PlusDiesel co:alternative [
  a co:ConfigurationLink ;
  co:specToBeRemoved r:Diesel ;
  co:specToBeAdded r:Gasoline ;
  co:linkedConf ex:Conf1PlusGasoline .].
```

co:impossible. When a specification is not compatible with a configuration, the configuration engine can nevertheless provide a way to select it - of course, at the cost of discarding some of the previous selections; there is a conflict, to be resolved by removing or changing some of the `co:definingChoice(s)`.

```
ex:Conf1 co:impossible [
  a co:ConfigurationLink ;
  co:specToBeAdded r:SunRoof ;
  co:linkedConf ex:Conf1WithResolvedConflict .].
```

co:defaultSpec. Specification included by default in a Completely Defined Product matching this configuration.

co:lexicon. Used in particular to link a Configuration to the corresponding Lexicon. A way to get the variables definition, an information an application can use, for instance to make explicit the fact that specifications corresponding to a given variable are alternatives ones, e.g. to display a menu with radio buttons to choose the fuel type from.

Price. A configuration has a starting price, corresponding to the price of the cheapest product matching this configuration. We use the `gr:hasPriceSpecification` to state the starting price of a Configuration:

```
ex:Conf1 gr:hasPriceSpecification [
  a gr:UnitPriceSpecification ; gr:hasCurrency "EUR" ;
  gr:hasMinCurrencyValue "9000.00"^^xsd:float. ].
```

Starting prices of the linked configurations can be embedded within the RDF data returned when dereferencing the configuration:

```
ex:Conf1 co:possible [ a co:ConfigurationLink ;
  co:specToBeAdded r:Diesel ;
  co:linkedConf ex:Conf1PlusDiesel .].
ex:Conf1PlusDiesel gr:hasPriceSpecification [
  a gr:UnitPriceSpecification ; gr:hasCurrency "EUR" ;
  gr:hasMinCurrencyValue "10000.00"^^xsd:float. ].
```


Completion. Any configuration can be completed. The `co:minPriceCompleted-Conf` property links to a completely defined product matching this configuration, at the same price.

5.4 Integration with GoodRelations

A configuration mainly describes a Partially Defined Product. As such, in GoodRelations terms, a `co:Configuration` is a `gr:ProductOrServiceModel`:

an intangible entity that specifies some characteristics of a group of similar, usually mass-produced products, in the sense of a prototype.

The suffix “Model” may seem misleading when used for a Configuration, as it suggests something such as “Ford T”, and not “Ford T with Air Conditioning and MP3 connection plug” (itself not a completely defined product - you still can choose, well, the color: it is a “prototype of similar products”).

On the other hand, a configuration has a price. It may be seen as a commercial offer, or the expression of a customer’s wish list. It can therefore be considered as a `gr:Offering` as well. Giving, as we do, `gr:hasPriceSpecification` the start price of a `co:Configuration` makes it a *de facto* `gr:Offering`. Also, the range depends on the vendor, a typical characteristic of an offer; e.g. two PC vendors both sell, say, PC intel core i7 2500K, 4GB RAM: this is a configuration; however they propose different disk capacities.

So, a Configuration can be considered as both a `gr:ProductOrServiceModel` and a `gr:Offering`.

5.5 Vocabularies for Specifications

This ontology is generic: it does not depend on the variables and specifications used to define a product, and it allows a publisher to use its own terms as specifications. This is an important point, as the whole purpose of the configuration process is to come out with an order for a completely defined product, which implies its definition in the manufacturing company’s terms. On the other hand, there are shared vocabularies on the web for products. No technical obstacle prevents us from adding triples using terms coming from such vocabularies to the description of a Configuration. Example using the Vehicle Sales Ontology:

```
ex:Conf5 a co:Configuration ;
    co:chosenSpec r:Model1, r:Gasoline ;
    vso:fuelType dbpedia:Gasoline .
```

We won’t go further into this question.

5.6 Indexing Configurations

The configuration ontology gives us the means to precisely describe ranges of customizable products, making it easy to crawl them. In section [3.3](#), we saw how

an agent can implement a text based searching mechanism with small indexes, and with calls to the configuration service. What about search engines, then? We expect them to index our products as a matter of course.

The harsh reality, though, is that ranges are huge. We can proudly announce the availability of our 10^{20} descriptions of completely defined products on the web of data, and of even more partially defined ones, yet this is far more than what the most obstinate robot can cope with. So, we cannot but give thought to the fact that indexing will be partial.

Basically, configuration will be indexed by specifications. The semantics of the properties used to describe a Configuration should be carefully taken into account when deciding on which specifications indexing will be based. For instance, if the values of the “co:possible” property were used to index configurations, queries searching for products containing several specifications could return matches that actually do not include their conjunction: spec1 and spec2 can both be individually compatible with a given configuration, while spec1 and spec2 together is impossible. Or, they could get displayed at a lower price than the true one: the start price of a configuration generally increases when options are added. The only way to return accurate results would be to query the configuration service at runtime; while this is a simple thing for a specialized agent to do, search engines will not. As an other example, indexing configurations with chosen and implied specifications only would require to build a very large index, to get matches for searches involving many specifications. The best solution probably uses the union set of the values of co:chosenSpec, co:impliedSpec and co:defaultSpec.

Of course we do not know how search engines will proceed. We enable them to crawl the dataset, either starting just from its root (the “empty configuration”), or from any configuration, and following links whose semantics is precisely defined in the co:ConfigurationLink class. We provide them with enough information to customize their strategies. For instance, they can choose which links they follow. Not all specifications are of equal interest: the sun roof, the MP3 connector, etc. are probably more important - for a customer as well as for a search engine - than, say, the color of the ashtray.

On the other hand, the “sitemap” file of the web site is the place for the publisher to list configurations the indexing robots should consider first. A still unanswered question is: which configurations should be included in the sitemap file to get the most of it from a marketing point of view? Clearly, the choice should be driven by marketing data: for instance, which specifications and configurations should be “pushed” toward the customer?

6 Benefits

6.1 Improved Architecture

As noted in section [4](#), the access we historically provided to the functionalities of our configuration engine was through a java API. Switching to a REST based API brought its own benefits. Before this change, our configuration engine and

associated PRS data were duplicated in a number of different applications: web sites, salesman's assistant, etc. We improved our architecture by having it now really web based. Data and configuration engine are centralized on one server, accessed by all applications needing configuration functionalities. Updates are easier, server resources are shared, and the web architecture ensures scalability.

6.2 Reduced Development Costs of Client Applications

The development of several new client applications is on its way, and the costs are much lower than with our previous Java API: the GUI developer does not have to understand the concepts underlying configuration, nor (for the larger part) to learn an API. Basically, she just has to display the links found in the data.

6.3 Benefits of Universal Configuration Identifiers

Configurations truly deserve their status of first class objects. They represent Partially Defined Products. They also capture the exact expression of the customer's wish list, constrained by the definition of the range: a very important point of concern from a marketing point of view! Global identifiers for configurations may be put to a number of uses, most of which increase the visibility of the commercial offer. To name a few:

Tagging web content, defining links. Configurations may be used in describing web pages, can be the subject of clickable ads, etc.

Easily sharing configurations between applications, devices, media. Identifying configurations with URIs allows for their easy sharing between corporate systems: web site, salesman's assistant, ordering system, etc., as well as outside ones such as social networks. This results in improved fluidity of the e-business processes. As a proof of concept, we developed prototype software showing how a potential customer can begin a configuration by clicking on an ad or decoding a QR code in a billboard; modify it on her smartphone or PC; exchange it with members of her family; share it on FaceBook; have it transferred to the salesman's assistant when she finally goes to a shop, and have it converted to an order. By the way it emerged that integration with Facebook OpenGraph has applications of interest to marketing people.

6.4 e-business Use Case Example: Targeted ads

Agents knowledgeable about the buying habits and preferences of consumers can use this data to generate ads matching their possible wishes better. For instance, if a user, known to be young and accustomed to buying and downloading music, issues a query about cars, display an ad for a small car with an MP3 adaptor.

7 Conclusion

Data about customizable products can be published effectively as Linked Data. We described the corresponding service and ontology. Most, if not all, configurator applications on the web could be modified with relative ease, to publish data that way. It gets us accurate descriptions of complex ranges of products, which can be crawled and understood by simple agents: all reasoning takes place inside the service publishing the data, its complexity hidden from the clients. For search engines, the number of configurations is challenging - we added more than 10^{20} of them to the web of data - but the linked nature of the dataset should be sufficient to use it effectively. As for specialized agents, we expect them to offer new functionalities with configuration data, such as comparing ranges based on customer's interests; and in the end, to answer the question: what are the prices and CO2 emission levels of small cars with gasoline engines, sun-roofs, air conditioning, and MP3 adapters?

References

1. Hepp, M.: GoodRelations: An Ontology for Describing Products and Services Offers on the Web. In: Gangemi, A., Euzenat, J. (eds.) EKAW 2008. LNCS (LNAI), vol. 5268, pp. 329–346. Springer, Heidelberg (2008)
2. Badra, F., Servant, F.P., Passant, A.: A Semantic Web Representation of a Product Range Specification based on Constraint Satisfaction Problem in the Automotive Industry. In: OSEMA Workshop ESWC (2011), <http://ceur-ws.org/Vol-748/paper4.pdf>
3. Pargamin, B.: Vehicle Sales Configuration: the Cluster Tree Approach. In: ECAI Workshop on Configuration (2002)

From Web 1.0 to Social Semantic Web: Lessons Learnt from a Migration to a Medical Semantic Wiki

Thomas Meilender^{1,2}, Jean Lieber², Fabien Palomares¹, and Nicolas Jay²

¹ A2ZI, 3 route de Boncourt, 55200 Commercy, France
{thomas.meilender, fabien.palomares}@a2zi.fr

² Université de Lorraine, LORIA, UMR 7503, Vandœuvre-lès-Nancy, F-54506, France
CNRS, LORIA, UMR 7503, Vandœuvre-lès-Nancy, F-54506, France
Inria, Villers-lès-Nancy, F-54600, France
{thomas.meilender, jean.lieber, nicolas.jay}@loria.fr

Abstract. Oncolor is an association whose mission is to publish and share medical guidelines in oncology. As many scientific information websites built in the early times of the Internet, its website deals with unstructured data that cannot be automatically queried and is getting more and more difficult to maintain over time. The online contents access and the editing process can be improved by using web 2.0 and semantic web technologies, which allow to build collaboratively structured information bases in semantic portals. The work described in this paper aims at reporting a migration from a static HTML website to a semantic wiki in the medical domain. This approach has raised various issues that had to be addressed, such as the introduction of structured data in the unstructured imported guidelines or the linkage of content to external medical resources. An evaluation of the result by final users is also provided, and proposed solutions are discussed.

Keywords: semantic wikis, decision knowledge, medical information systems.

1 Introduction

During the two last decades, the Internet has totally changed the way information is published and shared in most of scientific areas, including medicine. First websites in web 1.0 were made of static pages and hyperlinks allowing limited interactions between editors and readers. Then, information sharing has evolved with the rising of web 2.0 by allowing users to contribute to the contents. Numerous studies have shown the position impact of such evolutions on medical information systems [1,23]. Participative web applications can be implemented and used in a collaborative way to build large databases. Finally, semantic web has appeared. Semantic web aims at creating and sharing formalized information in order to make it available for both humans and machines. Social semantic web

is considered as the merging of web 2.0 and the semantic web, i.e. a web where shared formal information is edited collaboratively.

The Kasimir research project started in 1997. It aims at providing tools to assist decision making by practitioners and, more generally, decision knowledge management in oncology. The project is conducted in partnership with Oncolor, an association gathering physicians from Lorraine (a region of France) involved in oncology. On its static website, Oncolor publishes more than 140 medical guidelines written in HTML in a web 1.0 fashion. This base is built through a consensus between medical experts and is continually updated according to the oncology state of the art and to local context evolutions. In order to facilitate the creation, maintenance and publication of guidelines, Oncolor has expressed the need for more efficient and collaborative tools. Moreover, it would be a great benefit if the knowledge contained in guidelines was formalised and made available for semantic systems, particularly for Kasimir, since knowledge acquisition is a bottleneck for building knowledge systems.

In this paper, an application of a semantic wiki approach for medical guideline edition is reported¹. The expected benefits are twofold: first, online collaborative work is simplified by the use of wikis and second, semantic technologies allow the creation of additional services by making use of external medical resources such as terminologies, online ontologies, and medical publication websites. However, despite the effort of the semantic wiki community to simplify its systems, it is still hard for medical expert to create semantic annotations. This issue involves the need of taking into account structured and unstructured content but also, when this is possible, to include dedicated tools for formalising data. In these cases, implementation and development of semantic wiki extensions are required.

The rest of the paper is structured as follows: Section 2 describes the application context. The migration of static Oncolor website to a collaborative system is presented in Section 3, while Section 4 relates the addition of semantic annotations and services. After a report on our evaluation study in Section 5, some related work is introduced in Section 6. Section 7 is a discussion about the benefits of the system, as well as ongoing and future work.

2 Context

2.1 Application Context

Oncolor Website and Oncology Guidelines. One of Oncolor's objectives is to create and to keep up to date oncology guidelines. Clinical guidelines are sets of recommendations on treatments and care of people with specific diseases. They aim at improving treatment quality and patient support by standardising cares. They are based on clinical evidence, clinical trials and consensus between medical experts from different specialties such as oncology, surgery, etc.

More than 140 guidelines have been edited to give recommendations about treatment of many different cancers as well as typical situations such as pain

¹ <http://oncowiki.a2zi.fr>

treatment or dental care. Since guidelines are intended for both medical staff and patients, editors have exploited various kinds of formats in order to be both precise and didactic. Most guidelines follow the same structure. The first part introduces the guideline with few sentences that explain which circumstances imply the use of the guideline and the treatments that will be proposed. The next part is a textual description of clinical and paraclinical investigations that can lead to the starting point of the guideline. This starting point is often a staging step allowing to classify the patient according to international classifications. These classifications are presented as simple tables. Depending on classifications results, decision trees guide the reader to the next step that details the medical recommendation available in various formats, such as medical publications in PDF or hypertext links to distant resources. Finally, guidelines conclude with advice about medical supervision and sometimes with a lexicon of specific terms.

As in all medical information systems, data quality in oncology is critical. Each guideline should be reviewed every second year by experts. Two kinds of editors can be identified in the reviewing process:

- Medical experts contribute with their technical knowledge. They are gathered in committees under the supervision of coordinators that make sure the guidelines are complete and the consistent. Most medical experts have poor computer skills, limited to word processing and Internet browsing.
- Oncolor staff manages communication between the committee members and creates the final guideline layout. They also check that guidelines are up to date and propose new ways to facilitate their diffusion, while public health physicians check the consistency of the information base. Most of Oncolor employees do not have more computer skills than medical experts, except for a computer graphic designer. Particularly, Oncolor does not have a webmaster in its staff.

Guidelines are made available on the Oncolor website [2], which also contains various information about local healthcare services and provides links to dedicated tools. This site also stores other Oncolor projects, including a thesaurus of pharmacological products which is closely related to oncology guidelines. It contains information about drugs used in cancer treatment.

Created in the mid 1990s, this website was completely made using a commercial WYSIWYG HTML editor. The resulting HTML code is not readable, due to successive technology evolutions. The first created pages were done using only HTML and then, in the past 15 years, CSS, Javascript and XHTML were introduced. Few pages also use ASP. All these evolutions have led to the construction of weird pages where only the visual aspect is important and in which document structure is hard to identify. Over the years, updating the website is becoming more and more complex for Oncolor staff. All the pages edited on the Oncolor website must be validated to follow the principles of HONcode certification [1] which guarantee the quality and the independence of the content.

In this context, Oncolor has been asked to integrate a collaborative tool to simplify the guideline creation and maintenance process. Moreover, it would be of great benefits for Oncolor to keep track of all changes in the guidelines. That

is why the system has to propose a versioning file system and some social tools to allow communication between experts during updating process.

The Kasimir Research Project. Started in 1997, Kasimir is a multidisciplinary project also involving industrial (A2ZI) and academic (LORIA, CNAM Laboratory of Ergonomics) partners. Kasimir aims at providing software to assist decision making by practitioners and more generally decision-making knowledge management in oncology. The Kasimir project's recent work mainly focuses on semantic web as a background for formalizing, sharing, and exploiting pieces of knowledge [9]. The last version of the KATEXOWL toolkit and, particularly, the framework EDHIBOU [4], use semantic web technologies such as SparQL and OWL for storing and exploiting pieces of knowledge. It can automatically generate simple user interfaces for decision support thanks to user-friendly forms that guide practitioners around the knowledge base.

To fill its scientific contribution, Kasimir needs to use more widely its tools by taking advantage of real world data sources. However, few guidelines are currently available for EDHIBOU: they need to be formalised, i.e. transformed into a knowledge base using a formalism that can be handled by an inference engine. Until now, this complex step required two experts: a medical domain expert writing guidelines and validating the final results, and a knowledge engineer formalising them. It seems that if medical domain experts could formalise the guidelines themselves in a machine-understandable way, the process would be simplified. Even if this goal seems very difficult to reach for now, it would be a good evolution if formalisation tools could help experts make simple semantic annotations.

2.2 Scientific Context

Medical Resources. To build efficient tools, it is important to take into account numerical digital resources already available. Among them, large websites reference scientific communications in the domain of medicine, such as the well-known Pubmed [3]. Pubmed provides an easily configurable search engine that can be called through distant requests. Publications are indexed using a specific controlled vocabulary, Medical Subject Headings (MeSH [20]). MeSH contains more than 25,000 descriptors, most of these accompanied by a short description or definition, some links to related descriptors, and a list of synonyms or very similar terms. In the French context, Cismef [10] uses a French traduction of MeSH to index medical online resources with a French vocabulary.

Beyond the already-cited MeSH, many controlled vocabularies have been used to structure medical applications [8]. Among resources available in French, the International Statistical Classification of Diseases and Related Health Problems (ICD-10) is probably one of the simplest. ICD-10 is a medical classification that provides codes to classify diagnoses and causes of death and is organised as a simple hierarchy. ICD-10 is widely used in medical information systems, but semantic applications generally use other vocabularies due to its lack of semantic depth. Considered the most comprehensive, SNOMED is a multiaxial,

hierarchical classification system including coverage of diseases, clinical findings, therapies, procedures and outcomes. About 270,000 concepts are described by unique identifiers with several labels and can be used to describe complex situation by using semantic relations and modifiers. It is interesting to note that MeSH, ICD-10, SNOMED and other ontologies such as Galen are integrated in the terminology integration system Unified Medical Language System (UMLS).

Moreover, many semantic web systems provide freely questionable online information. For example, BioPortal [5] is a repository of biomedical ontologies whose functionalities include the ability to browse, search, and visualise ontologies. More specialised, DrugBank [25] provides an annotated database of drugs and drug target information. Many other resources are available, such as Bio2RDF [6], which allows an access to Pubmed with linked data, or LinkedCT [12] which indexes clinical trials. The information resources cited above and many more can be interlinked by using DBpedia [7].

Wikis and Semantic Wikis: The Migration Process. Traditional wikis are usually based on a set of editable pages, organised into categories and connected by hyperlinks. They became the symbol of interactivity promoted through web 2.0. One of the founding principles of wikis, which is also the principal vector of their popularity, is their ease of use even by persons that lack considerable computer skills. Wikis are created and maintained through specific content management systems, the wiki engines, while *wikitexts* enable structuring, layout, and links between articles. At this point, an idea has emerged: to exploit stored pieces of knowledge automatically.

Indeed, a limit use to the wikis is illustrated by the querying of the data contained in their pages. The search is usually done through word recognition by strings, without considering their meaning. For example, the system cannot answer a query like: “Give me the list of all currently reigning kings.” The solution used in Wikipedia is a manual generation of lists. However, the manual generation of all the lists answering queries users may raise is, at the very least, tedious, if not impossible. This has motivated the introduction of a semantic layer to wikis. Moreover, it would be interesting if information contained in wikis were available through external services.

Semantic wikis were born from the application of wiki principles in the semantic web context. A semantic wiki is similar to a traditional one in the sense that it is a website where contents are edited in a collaborative way by users and are organised into editable and searchable pages. However, semantic wikis are not limited to natural language text. They characterise the resources and the links between them. This information is formalised and thus becomes usable by a machine, through processes of artificial reasoning. Thus, semantic wikis can be viewed as wikis that are improved by the use of semantic technologies as well as collaborative tools for editing formalised knowledge.

Semantic wikis corresponds to both Oncolor and Kasimir needs: guidelines can be written in a collaborative way and semantic technologies allow to formalise and extract structured content.

3 From Web 1.0 to Web 2.0

3.1 Choosing a Semantic Wiki Engine

The first part of the migration was choosing the most adapted semantic wiki engines. Whereas many semantic wiki engines have emerged for the last 10 years, only four open source projects seem active at this time: AceWiki [17], KiWI [22], Ontowiki [13], and Semantic Mediawiki [16]. AceWiki uses ACE (*Attempto Controlled English*), a sub-language of English that can be translated directly into first order logic. However, Oncolor guidelines are already written in French and the development of a controlled language for French medical guidelines that covers all the contents would be tedious. Ontowiki and KiWI focus on RDF triple edition by proposing dedicated interfaces such as dynamic forms. Their approaches are very strict and do not seem reconcilable with importation of unstructured contents. Moreover, no large scale implementation of these engines can be found and, their development and user communities are limited. So, less extensions are available and the support is weak.

Semantic Mediawiki (SMW) seems to be the best solution. SMW is an extension of Mediawiki, the engine used by Wikipedia. For the sake of simplicity for users, it integrates the RDF triples editing in its wikitext. In this way, it enables the creation of typed links that can also be used for indicating the attributes of the page. Another interesting point of SMW is its popularity: there is a large community of developers around it, and this community produces many extensions, such as editing forms, the integration of an inference engine, etc. For instance, the Halo extension² proposes forms, an auto-completion system, the integration of a SPARQL endpoint and much more. The only limitation for our migration is that SMW does not provide extensions that allow to draw the trees that are frequently used in the guidelines, but we have developed a decision tree editor, as will be discussed further. Tutorials and community support make the installation of SMW simple. Less than one hour is needed to install it for anybody with average computer skills.

3.2 Importing Guideline Content

Once the semantic wiki had been installed, a specific skin that corresponds to Oncolor graphics standards has been built to customise the application. The next part of the work was to import guidelines in the wiki. However, in order to correspond to wiki syntax, content had to be formatted into wikitext. For each guideline, the HTML content was extracted and HTML pages were merged when guidelines did contain more than one page. The table of contents was automatically extracted and marked up when possible. However, the state of the HTML code made impossible to systemically identify document structure. It can be noted that the migration would have been simpler if CSS had been used from the start. Then, unnecessary content such as browsing elements and

² <http://www.projecthalo.com/>



Fig. 1. An excerpt of guideline in the wiki

JavaScript functions was removed. A parser was also used to transform HTML into wikitext when simple tags were detected (images, tables, etc.). Moreover, by using a parser and context analysis, specific fields were identified. The objective was to identify interesting information about a guideline such as the date of its last update or keywords. Moreover, by examining website folder structure, an anatomical classification of the guideline has been identify. This classification was reused as a base for guideline categorisation in the wiki.

Despite of all our efforts, the layout of the imported guidelines had to be checked then. Due to the critical nature of the information, this checking was done by Oncolor staff. On average, a person needed half a day to check each guideline.

Additionally, the Oncolor thesaurus of pharmacology was imported. As its content is closely related to guidelines, it was important to let it available in the same information system. One page per described drug was created. In this case, the simplicity of the HTML pages made the migration easier.

To migrate guidelines, Mediawiki import capacities were used. They allow to import wikitext content from text files. In the wiki, some templates were built to highlight the fields previously identified. An excerpt of a resulting page is shown in Figure 1. All the guidelines are presently in the wiki.

3.3 User Right Management

In the usual philosophy of wikis, everybody can edit pages, even anonymously. Although the importance of the information availability for the public, medical data are critical and the guidelines must be approved by Oncolor experts to be in public access. Moreover, if an expert modifies a guideline, the modification

has to be checked by the coordinator in charge of the guideline. During the revision period, modifications are numerous and each of them implies a complete review of the guideline and its layout. To allow private modifications, a special namespace has been created, that can be viewed as a workspace for the experts. Final versions are shown on the main namespace, and each guideline has an equivalent in the new namespace where experts can add their contributions. When a guideline is considered as correct and complete by the coordinator in the workspace, the page is simply copied to the final location in the main namespace.

According to this revision process, three kinds of users have been identified:

- anonymous users, that can read pages of the main namespace,
- medical experts, that can read pages of the main namespace and edit pages in the workspace,
- administrators, that can edit all pages, even wiki system pages.

3.4 KCATOS, a Decision Tree Editor

Decision trees were imported from the previous website as bitmap pictures. At this point, guideline updates can also be simplified by proposing an online editor. KCATOS is a Mediawiki extension that allows the collaborative drawing of decision trees. KCATOS decision tree language is a graphical representation based on a small set of geometrical figures connected by directed edges. This representation was directly inspired by the graphics standards of Oncolor. Indeed, guidelines use visual representations that can mostly be viewed as trees. An advantage to use these graphics standards is that Oncolor experts already know them. We want to preserve Oncolor's graphic semantics in order to facilitate the understanding of guidelines by physicians.

From a semantic point of view, each kind of node has its own meaning; e.g. rounded rectangles represent medical situations, etc.

4 Introducing and Exploiting Formalised Knowledge

4.1 Extracting Decision Knowledge from Decision Trees

Most of the time, decision trees can be considered as structures from which a meaning can be extracted. In order to avoid ambiguities and to guarantee guideline consistency, classical syntactical rules of decision trees are used. A syntactic module can be used to check if the edited tree respects the rules. Thus, KCATOS can propose an export algorithm that allows to transform decision trees into OWL.

KCATOS's export algorithm defines two classes: **Situation** and **Recommendation**. The first one represents some patient information while the second one represents the description of the decision proposed by the system. These classes are linked by the property **hasRecommendation**. This means that for each situation there is a recommendation that is associated to it.

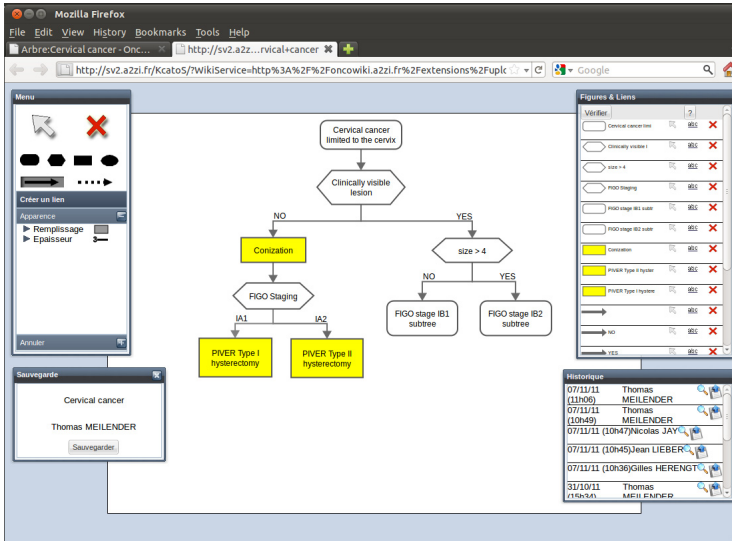


Fig. 2. The KCATOS decision tree editor interface

A tree is read using depth-first search. Each node is transformed using rules which take into account the shape and its ancestors.

The export algorithm creates many concepts and properties. Including all of them in the semantic wiki would decrease the ease of navigation because it would lead to the creation of numerous pages. In order to avoid these page creations, translated trees are stored in a specific file and linked to the wiki. Thus, created ontologies are made available for other semantic web applications. From a technical point of view, OWL API [15] is used to perform the export.

4.2 Using Semantic Tools of Wiki

Extracting the whole semantics of a guideline is a tedious job that has to be done by a medical expert with skills in knowledge engineering. As Oncolor does not have this kind of specialist in its staff, formalising the guidelines would be a great investment. Moreover, it is still difficult for non-specialists to understand the benefits that semantics could bring to medical information system. That is why the key idea of the project is to insert step-by-step useful semantic annotations into the guidelines in order to increase Oncolor interest in the semantic web technologies. The first way to introduce semantics is to exploit identified fields extracted during the guideline migration. To improve their visualisation and their update, SMW templates and queries mechanisms were used.

SMW proposes many ways to edit semantic annotations. The more basic way to create annotations is wikitext, which can be improve thanks to templates. Templates are generic pre-developed page layouts that can be embedded in several wiki pages. They can also manage variables that are instantiated in the

corresponding page. For instance, a template is used to generate the box in the top right corner of the page shown in Figure 1. The template used to create this box is generic enough to be applied to all guideline pages, and its use allows flexible modifications. As template use is simple (and can be further simplified by associating forms to them), they provide a simple way to create annotation fields that can be filled by any users without specific skills.

Then semantic annotations can be exploited by SMW inline query engine. Using a simple query language, semantic search can be done directly in a page and results are displayed as tables, lists, etc. Combined to templates, semantic queries are a simple way to create dynamic content relying on semantic annotations.

```

{{#ask: [[Category:Guideline]] [[last Update::<{{#time:d F Y|2 years ago}}]]
|?last Update
| sort = last Update
| format=template
| [...]
}}

```

(a) Excerpt of inline query that requests the guidelines that are out-of-date (translated from French).

➔ Référentiels nécessitant une mise à jour

AFFICHAGES		Ceci est un liste générée dynamiquement proposant les référentiels dont la dernière mise à jour r	
Page	Référentiel	Mise à jour	
Discussion	Vagin	10 octobre 2001	
Modifier	Vulve	10 octobre 2001	
Historique	Tumeurs nerveuses du médiastin	20 octobre 2001	
Renommer	Tumeur épithéliale du thymus	20 octobre 2001	
Suivre	Tumeurs du médiastin (diagnostic)	20 octobre 2001	
Reactualiser	Cancer à calcitonine thyroïde - NEM2	20 octobre 2001	
	Nodule thyroïdien : CAT	13 février 2002	

(b) Results of the query.

Fig. 3. An excerpt of inline query that requests the guidelines that are out-of-date, and the wiki page that contains the result

A use of templates and inline queries is shown by the management of the dates in the guidelines. Every guideline has at least one date that indicates the date of the last validated update. This date is entered in a template in which it is associated with a property which links the date to the guideline. Then, a maintenance page is created to highlight the guidelines that are out-of-date. The query is shown in Figure 3(a) while its result, that can be seen in Figure 3(b), is displayed as a table thanks to specific templates. Moreover, another query is added in the template present on each guideline which shows a warning if the guideline has to be updated.

Templates are also used to link guidelines to external publication resources. To create the link, the first step was to define a common vocabulary between guidelines and publication website. Then, templates were designed to allow easily semantic annotations in guideline using MeSH vocabulary. Cismef, which indexes a large amount of medical publication in French, already indexes Oncolor's guidelines using terms from the MeSH thesaurus. These terms were imported in the wiki as a base that can be freely edited. As PubMed also uses this thesaurus to index this document, requests to PubMed and Cismef can be automatically built using templates and inline queries. Each request is dedicated to the guideline it belongs to and provides publications that are indexed by the same terms. Thus, it provides a bibliography tool useful for staff and provides further information to the reader.

4.3 Querying Resources of Web of Data

To show another view of the semantic web, we tried to investigate on external structured data sources that could bring additional information to the wiki. So, an extension was created to query external sources using SPARQL. In this part, pharmacology thesaurus was used. The idea was to explore external resources by building SPARQL requests based on the name of the drug studied in a current page. The target of the searches was Drugbank, specialised in drug description, and DBpedia, a generalist knowledge base. Thus, for most of the drugs, we get additional information in the semantic web. An example is shown in Figure 4. However, most information are in English and we deplore the lack of available French information source. This module is no longer online pending the Oncolor board is approval of the use of external data sources and the validation of the ones that can be exploited.

5 Evaluation

To carry out the evaluation, the opinions of the users have been investigated. People asked were the four main contributors from Oncolor staff: two public health physicians, a computer graphic designer, and a medical secretary.

The first interesting point is that, before the beginning, the only thing they knew about wikis was Wikipedia and none had ever contributed to a wiki. Despite this, three contributors thought that less than one day of self-training is needed to learn wikitext and to be an efficient contributor. The only difficulties are related to particular layouts (tables and references) and wiki advanced functions dealing with user management and page history. The only reluctance to migrate to a wiki was guideline quality. They agreed a concern with that the old system was time-consuming, but it had the advantage to produce high quality guidelines. Experiments were led to update Oncolor's old website and semantic wiki with the same modifications. They show that the quality did not suffer of the change and that the efficiency of updating has been increased by the semantic wiki.

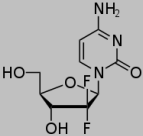
Gemcitabine	
	
Description (DrugBank)	Gemcitabine is a nucleoside analog used as chemotherapy. It is marketed as Gemzar [®] 200 mg by Eli Lilly and Company. As with fluorouracil and other analogues of pyrimidines, the drug replaces one of the building blocks of nucleic acids, in this case cytidine, during DNA replication. The process arrests tumor growth, as new nucleosides cannot be attached to the "faulty" nucleoside, resulting in apoptosis (cellular "suicide"). Gemcitabine is used in various carcinomas: non-small cell lung cancer, pancreatic cancer, bladder cancer and breast cancer. It is being investigated for use in oesophageal cancer, and is used experimentally in lymphomas and various other tumor types.
Mechanism of action	Gemcitabine inhibits thymidylate synthetase, leading to inhibition of DNA synthesis and cell death. Gemcitabine is a prodrug so activity occurs as a result of intracellular conversion to two active metabolites, gemcitabine diphosphate and gemcitabine triphosphate by deoxycytidine kinase. Gemcitabine diphosphate inhibits ribonucleotide reductase, the enzyme responsible for catalyzing synthesis of deoxynucleoside triphosphates required for DNA synthesis. Gemcitabine triphosphate (difluorodeoxycytidine triphosphate) competes with endogenous deoxynucleoside triphosphates for incorporation into DNA.
Toxicity	Myelosuppression, paresthesias, and severe rash were the principal toxicities. LD ₅₀ =500 mg/kg (orally in mice and rats)
Pharmacology	Gemcitabine is an antineoplastic anti-metabolite. Anti-metabolites masquerade as purine or pyrimidine - which become the building blocks of DNA. They prevent these substances becoming incorporated in to DNA during the "S" phase (or DNA synthesis phase of the cell cycle), stopping normal development and division. Gemcitabine blocks an enzyme which converts the cytosine nucleotide into the deoxy derivative. In addition, DNA synthesis is further inhibited because Gemcitabine blocks the incorporation of the thymidine nucleotide into the DNA strand.
Indication	For the first-line treatment of patients with metastatic breast cancer, locally advanced (Stage IIIA or IIIB), or metastatic (Stage IV) non-small cell lung cancer and as first-line treatment for patients with adenocarcinoma of the pancreas.
Sources	http://www4.wiwiw4-berlin.de/drugbank/resource/drugs/DB00441 http://kbpedia.org/resource/Gemcitabine

Fig. 4. Example of data that can be imported from DBpedia and Drugbank about *Gemcitabine* using SPARQL queries

Our panel cited the main advantages they see in using a wiki. They have agreed that wikis are collaborative tools that allow more reactivity and more flexibility in the update process. It has also been said that wikis improve conditions of employment by allowing distant work, which was impossible with the previous system. Moreover, they recognised that the wiki increases the quality of the editing process and of the guideline themselves by allowing the standardisation of the guideline and by simplifying the work on its layout.

In our system, the preferred contribution is the query to medical publication websites Pubmed and Cismef which propose automatically a bibliography related to a guideline. The previous system did not permit that kind of function that has been judged very useful. It is really important for the project that Oncolor staff appreciated this contribution that is relying on semantic web technologies. Moreover, all participants declared that they are interested in using MeSH annotations and want to lead further this experimentation.

6 Related Work

It already exists many medical wikis (e.g. medical portal of Wikipedia, <http://wikisr.openmedicine.ca>, <http://askdrwiki.com>, www.ganfud.org, etc.) but only few of them use semantic web technologies. OpenDrugWiki [18], which also uses SMW, is a wiki used as a back-office system for editing, merging from different sources, and reviewing information about drugs.

The closest semantic wiki to the one introduced in this paper is probably CliP-MoKi [21]. CliP-MoKi is a SMW-based tool for the collaborative encoding in a distributed environment of cancer treatment protocols. The wiki mainly relies

on semantic forms and focuses totally on structured content while our project aims at migrating already existing unstructured data.

Semantic wikis have already been experimented in various domains. Particularly, the building of a semantic portal for the AIFB Institute described in [14] shows how important the technical settings are for increasing wiki performances and how difficult it is to find the right balance between structured and unstructured data. This last issue has also been tackled in [24].

7 Lessons Learnt and Future Work

In this paper, a migration from a web 1.0 website containing medical data to a semantic wiki has been described. The first step was the migration of data from an HTML website to a collaborative solution, Semantic Mediawiki. The second step consisted in adding a semantic layer to show the benefits that semantic web technologies could bring.

Among the difficulties we have met, the analysis of the HTML version of the guidelines was hard because of the use of invalid code. This is the result of the use of different HTML editors that follow the evolution of the standard over a decade. It appears that a correct use of HTML and CSS would have simplified the migration, particularly the identification of tables of content and specific fields. Moreover, medical information is critical and its migration implies a long work of verification by medical experts. According to Oncolor members, about 70 days of work were necessary to check and correct all the guidelines.

Once the semantic wiki has been installed, the use of traditional wiki tools for edition was easily learnt by Oncolor staff. However, we have noticed that the creation and the use of semantic annotations remain difficult for non knowledge expert although semantic wikis seem to be a simple approach. For example, SMW inline query language is hard to handle for non computer specialists and template construction also requires computer skills. Some tools have yet to be implemented to improve this aspect in the philosophy of semantic forms and the Halo project.

Another problem was to find the right balance between structured and unstructured data. The advantage of structured data is the typing that enables to easily reuse data in the semantic web context. However, structured data are still difficult to edit and exploit, as shown in the context of semantic wikis. Moreover, most of existing information sources are unstructured, and tedious work would be necessary to transform them. This job would be expensive and time-consuming so its benefits have to be shown first to non semantic web experts. Our methodology was to add semantic annotations step-by-step to improve the semantic wiki quality. Until now, our work has consisted in showing the improvements so that future developments will be upon Oncolor request.

Introducing structured information yields benefits when it is done in accordance with already existing resources. In the medical domain, numerous thesauri and information sources have been created, and it is hard for no medical specialists to determine which ones can be used. This choice has to be made according to

the goal of the application with the approval of medical specialists. For instance, it was hard to determine which thesaurus will be used to index guidelines. We finally have chosen MeSH upon Oncolor request, although SNOMED or UMLS seem more complete and CIM-10 seems more simple. The reason was that the link to medical publication websites is useful for editors and provides additional information for the readers.

Finally, the use of data from semantic web is a major concern in the medical domain, due to the critical nature of the data. Using external resources seems to cause a kind of reluctance in clinicians. Each source has to be first approved by medical authorities before it can be exploited by a medical system. Particularly, all sources must at least follow the principle of the HONcode certification.

Currently, our work focuses on minor technical adaptation of the wiki to Oncolor needs. Our next task will be to increase gradually the semantic annotation's presence. The long-term goal is to obtain a structured knowledge base that contains all the information provided by oncology guidelines. For such a project to be successful, several issues have to be taken into account. The project must be able to rely on several medical experts to structure and check information. From this point of view, Oncolor will have a crucial role of support to play and so, their satisfaction is really important. Moreover, to complete the formalisation, resources that are more expressive than MeSH will be needed. SNOMED or UMLS seem to be better options. Finally, the scale of this final ontology will require significant improvement in ontology engineering tools, particularly for the edition and the maintenance.

References

1. Honcode, <http://www.hon.ch/> (last consulted: December 2011)
2. Oncolor website, <http://www.oncolor.fr> (last consulted: December 2011)
3. Pubmed, <http://www.ncbi.nlm.nih.gov/pubmed/> (last consulted: December 2011)
4. Badra, F., d'Aquin, M., Lieber, J., Meilender, T.: EdHibou: a customizable interface for decision support in a semantic portal. In: Proc. of the Poster and Demo. Session at the 7th International Semantic Web Conference (ISWC 2008), Karlsruhe, Germany, October 28 (2008)
5. Bail, S., Horridge, M., Parsia, B., Sattler, U.: The Justificatory Structure of the NCBO BioPortal 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. 67–82. Springer, Heidelberg (2011)
6. Belleau, F., Nolin, M.-A., Tourigny, N., Rigault, P., Morissette, J.: Bio2rdf: Towards a mashup to build bioinformatics knowledge systems. *Journal of Biomedical Informatics* 41(5), 706–716 (2008)
7. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: Dbpedia - a crystallization point for the web of data. *Journal of Web Semantics: Science, Services and Agents on the WWW* 7(3), 154–165 (2009)
8. Bodenreider, O.: Biomedical ontologies in action: Role in knowledge management, data integration and decision support. In: IMIA Yearbook Medical Informatics, pp. 67–79 (2008)

9. D'Aquin, M., Brachais, S., Lieber, J., Napoli, A.: Decision Support and Knowledge Management in Oncology using Hierarchical Classification. In: Kaiser, K., Miksch, S., Tu, S.W. (eds.) *Proc. of the Symp. on Computerized Guidelines and Protocols, CGP 2004*, Prague, Czech Republic. *Studies in Health Technology and Informatics*, vol. 101, pp. 16–30. IOS Press (2004)
10. Darmoni, S.J., Thirion, B., Leroyt, J.P., Douyère, M., Lacoste, B., Godard, C., Rigolle, I., Brisou, M., Videau, S., Goupyt, E., Piott, J., Quéré, M., Ouazir, S., Abdulrab, H.: A search tool based on 'encapsulated' MeSH thesaurus to retrieve quality health resources on the internet. *Medical Informatics and The Internet in Medicine* 26(3), 165–178 (2001)
11. Giustini, D.: How Web 2.0 is changing medicine. *BMJ* 333, 1283–1284 (2006)
12. Hassanzadeh, O., Kementsietsidis, A., Lim, L., Miller, R.J., Wang, M.: Linkedct: A linked data space for clinical trials. *CoRR*, abs/0908.0567 (2009)
13. Heino, N., Dietzold, S., Martin, M., Auer, S.: Developing Semantic Web Applications with the OntoWiki Framework. In: Pellegrini, T., Auer, S., Tochtermann, K., Schaffert, S. (eds.) *Networked Knowledge - Networked Media*. *SCI*, vol. 221, pp. 61–77. Springer, Heidelberg (2009)
14. Herzig, D.M., Ell, B.: Semantic MediaWiki in Operation: Experiences with Building a Semantic Portal. 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. 114–128. Springer, Heidelberg (2010)
15. Horridge, M., Bechhofer, S.: The owl api: A java api for owl ontologies. *Semantic Web* 2(1), 11–21 (2011)
16. Krötzsch, M., Vrandečić, D., Völkel, M., Haller, H., Studer, R.: Semantic wikipedia. *Journal of Web Semantics* 5, 251–261 (2007)
17. Kuhn, T.: How controlled english can improve semantic wikis. In: *Proc. of the 4th Workshop on Semantic Wikis, European Semantic Web Conference 2009*. *CEUR Workshop Proceedings* (2009)
18. Köstlbacher, A., Maurus, J., Hammwöhner, R., Haas, A., Haen, E., Hiemke, C.: Opendrugwiki – using a semantic wiki for consolidating, editing and reviewing of existing heterogeneous drug data. In: *5th Workshop on Semantic Wikis Linking Data and People, SemWiki 2010* (May 2010)
19. Lange, C., Schaffert, S., Skaf-Molli, H., Völkel, M. (eds.): *4th Semantic Wiki Workshop (SemWiki 2009) at the 6th European Semantic Web Conference (ESWC 2009)*, Hersonissos, Greece, June 1. *CEUR Workshop Proc.*, vol. 464. *CEUR-WS.org* (2009)
20. Nelson, S.J.: Medical terminologies that work: The example of mesh. In: *Proc. of the 10th Int. Symposium on Pervasive Systems, Algorithms, and Networks*, December 14–16, pp. 380–384 (2009)
21. Rospocher, M., Eccher, C., Ghidini, C., Hasan, R., Seyfang, A., Ferro, A., Miksch, S.: Collaborative Encoding of Asbru Clinical Protocols, pp. 1–8 (2010)
22. Schaffert, S., Eder, J., Grünwald, S., Kurz, T., Radulescu, M., Sint, R., Stroka, S.: Kiwi - a platform for semantic social software. In: Lange, et al. [19]
23. Schreiber, W., Giustini, D.: Pathology in the era of Web 2.0. *American Journal of Clinical Pathology* 132, 824–828 (2009)
24. Sint, R., Stroka, S., Schaffert, S., Ferstl, R.: Combining unstructured, fully structured and semi-structured information in semantic wikis. In: Lange, et al. [19]
25. Wishart, D.S., Knox, C., Guo, A., Cheng, D., Shrivastava, S., Tzur, D., Gautam, B., Hassanali, M.: Drugbank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic Acids Research* 36(Database-Issue), 901–906 (2008)

Semantics Visualization for Fostering Search Result Comprehension

Christian Stab, Kawa Nazemi, Matthias Breyer, Dirk Burkhardt,
and Jörn Kohlhammer

Fraunhofer Institute for Computer Graphics Research IGD
64283 Darmstadt, Germany

{christian.stab,kawa.nazemi,matthias.breyer,dirk.burkhardt,
joern.kohlhammer}@igd.fraunhofer.de

Abstract. Current search engines present search results in an ordered list even if semantic technologies are used for analyzing user queries and the document contents. The semantic information that is used during the search result generation mostly remains hidden from the user although it significantly supports users in understanding why search results are considered as relevant for their individual query. The approach presented in this paper utilizes visualization techniques for offering visual feedback about the reasons the results were retrieved. It represents the semantic neighborhood of search results, the relations between results and query terms as well as the relevance of search results and the semantic interpretation of query terms for fostering search result comprehension. It also provides visual feedback for query enhancement. Therefore, not only the search results are visualized but also further information that occurs during the search processing is used to improve the visual presentation and to offer more transparency in search result generation. The results of an evaluation in a real application scenario show that the presented approach considerably supports users in assessment and decision-making tasks and alleviates information seeking in digital semantic knowledge bases.

Keywords: Semantic Search, Information Visualization, SemaVis, Search User Interface, Visual Query Enhancement.

1 Introduction

The optimal use of information and knowledge plays a major role in global competition and forms the basis for competitiveness of industrial companies. Thereby, semantic technologies provide adequate linking tools for heterogeneous data sources as well as the generation of a broader context that facilitates information access and enables data exchange between different systems [1]. With the ongoing establishment of semantic technologies like the *Resource Description Framework (RDF)*¹, the *Web Ontology Language (OWL)*² and semantic-oriented query languages

¹ <http://www.w3.org/RDF/>

² <http://www.w3.org/TR/owl-features/>

like SPARQL³ these developments are not only limited to specific domains but also adopted in daily search processes of web-based search engines [2]. In both, domain-specific applications and web-based search engines, the results of search processing are usually presented in sorted lists. In most cases the ordering of list entries represents the relevance of the results for the individual search of the user according to various criteria [3]. So the most relevant result is placed in the first row followed by less important ones. Using this kind of result presentation, the semantic information of the documents that is used during search result generation and the analysis of search terms, remains in most cases hidden from the user, though this information considerably supports users in information-seeking tasks and selection of appropriate documents for further examination.

According to Hearst [4] efficient and informative feedback is critically important for designing search user interfaces. This includes in particular feedback about query formulation and about reasons the particular results were retrieved. However, relevance indicators besides list ordering such as numerical scores or special icons are less frequently used because the meaning of the relevance score is opaque to the user [5] in these presentations. This is because the majority of existing relevance indicators only presents a single relevance per search result that summarizes all criteria instead of offering a more fine-grained insight to search result processing.

In order to offer users an adequate tool that provides nevertheless the possibility to assess the relevance of retrieved search results, we developed a novel approach that utilizes information visualization techniques and semantic information that emerges during search result generation. The major contributions and benefits of our approach are:

- *Support for relevance assessment:* The presented approach supports users in assessing the relevance of search results and offers more transparency in the search result generation process.
- *Query-Result-Relation visualization:* The visual representation of relations between query terms and search results as well as the retrieved semantic meaning of query terms offers a fine-grained visual overview of search result relevancies and facilitates the information seeking and decision making process.
- *Visual feedback for query-enhancement:* The illustration of additional attributes and possible terms related to a given search request allows users to narrow search results and to refine the individual search process.

The rest of the paper is organized as follows: In the next section we introduce our approach for presenting search results in semantic domains and give a detailed description of all parts and features. Then we introduce the application scenario of the visualization and give an overview of its domain. We present the evaluation that we have performed to compare our approach to already existing solutions followed by a related work section, a discussion and a prospect of future work. As a detailed description of the whole search process with all technical aspects is beyond the scope of this paper, we only briefly describe the semantic background processing and focus on the aspects of the visualization component and the advantages of semantics for visualizing search results.

³ <http://www.w3.org/TR/rdf-sparql-protocol/>

2 Visualizing Search Results in Semantic Domains

The visualization approach is part of the SemaVis-Framework⁴, an adaptive visualization framework that contains different aspect-oriented visualizations [6, 7, 8] for presenting semantic information. The framework also provides different data providers, modular presentation models and a script-based configuration language that facilitate the development and integration of novel visualization techniques in semantically modeled environments. In the following we will give an overview of the visualization component before introducing the details in succeeding sections.

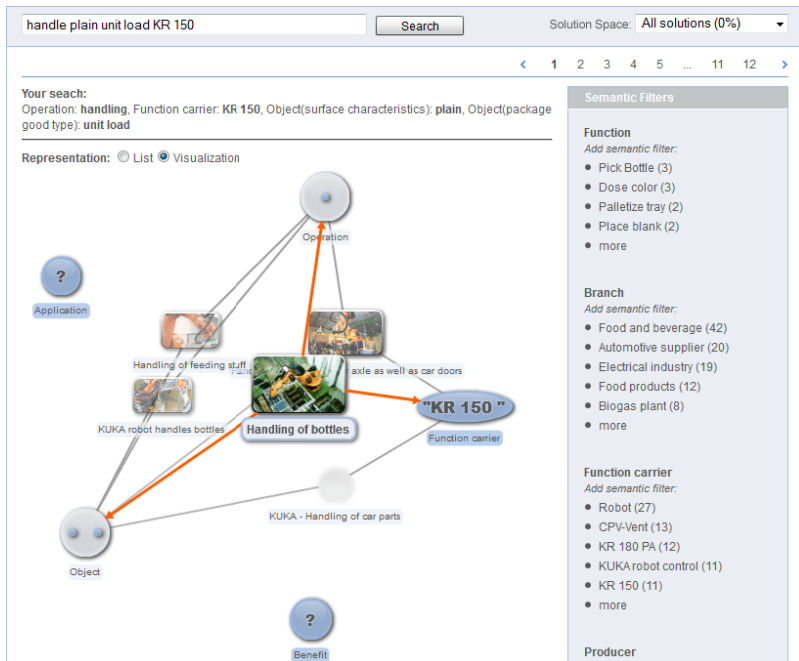


Fig. 1. Semantics visualization of search results showing query-result-relations, retrieved semantic meanings of query terms and result relevancy for an exemplary search

The visualization component is mainly based on a force-based layout algorithm that positions nodes in a two-dimensional space by assigning different forces to the edges and nodes of the graph. By adapting these forces in an iterative simulation, the physical system reaches a mechanical equilibrium resulting in an aesthetical layout of the graph. In contrast to commonly used force-based algorithms that assign the same force to every node and edge respectively, our approach utilizes a weighted model

⁴ <http://www.semavis.com>

based on similarities emerged during the semantic retrieval process. Another difference is that the visualization distinguishes between different node types:

- *Attribute nodes* are not positioned by the force-based algorithm. Instead, they are placed by a second layout algorithm in a circular form during the initialization of the visualization and users are allowed to freely move them on the surface (Fig. 1 shows an example).
- *Result nodes* represent the hits found for the given search query. They are suspended between the attribute nodes and are positioned by the force-based layout algorithm according to their similarities and relations to the attribute nodes.

Figure 1 shows an example with five attribute nodes and five result nodes. The weighting of the attraction and repulsion of the nodes and edges according to the retrieved similarity values arranges the result nodes closer to more similar attributes. So the best hit that matches to all attributes is placed near the center of the visualization.

2.1 Visualizing Query-Result-Relations

Giving adequate feedback about the reasons the results were retrieved is one of the major challenges for designing adequate search user interfaces. This is especially important for semantic search engines, in which the meaning of query terms is interpreted by means of semantically modeled entities, because the interpretation might be highly ambiguous. For example the query term *ford* might be interpreted as the name-attribute of a car manufacturer, as the surname-attribute of the famous inventor or the title-attribute of an activity for crossing rivers. Each of these interpretations will deliver a completely different result set. So it is not sufficient to only present the relations between query terms and results, but it is also necessary to point out the semantic interpretation of the given query terms to allow an unambiguous assessment of retrieved results.

To meet these demands and to provide an adequate tool that allows users to unambiguously determine the most relevant result for their individual search, our approach visualizes both query-result-relations and the interpreted semantic meaning of query terms. Therefore, each term of the given query is presented in an attribute node of the visualization. The interpreted semantic meaning emerged during search processing is visible in the label of the attribute node. So for every possible interpretation a new node is created that represents the query term and its semantic meaning. The relations between search results and the instantiated⁵ attribute nodes are depicted as directed and weighted edges between attribute nodes and result nodes. As mentioned above, the weighting of an edge is derived from the retrieved similarity between the result and the attribute node, whereby the results are placed nearer to more relevant query terms and attribute nodes respectively.

⁵ *'instantiated'* in this context means that a query term is assigned to a specific attribute.

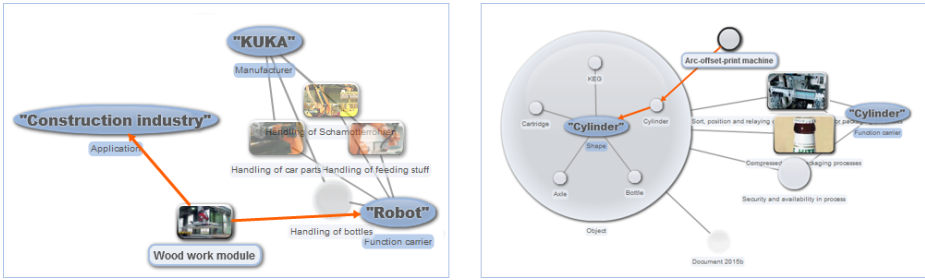


Fig. 2. Left: The visualization of query-result-relations reveals that only one of the five results is semantically related to the queried application area. Right: The visual representation of the identified semantic meanings of query terms avoids mistakes and ambiguity in result assessment tasks.

Figure 2 shows the result visualization of the query ‘*kuka robots in construction industry*’, where the term *kuka* is identified as manufacturer, the term *robot* as function carrier and *construction industry* as application area. The visualization reveals that only one of the results is related to the queried application area whereas other results are related to the given manufacturer (Figure 2 left). The second example shows the visualization of the results for the query ‘*cylinder*’. The given term is on the one hand identified as shape of an object and on the other hand as a specific function carrier. By visualizing the connections between search results and related interpretations of the query term, users can easily recognize the results that match their initial search intention.

2.2 Semantic Neighborhood and Hierarchical Attributes

Search results in semantic domains are not only retrieved by analyzing the content of resources but also by considering the semantic information and the semantic structure respectively. For example a resource that matches to only one of the given query terms is higher rated in the result list when the remaining terms match to semantically related resource. In some cases, semantic search processing enables the retrieval of highly relevant resources even if the given query terms are not contained in the resources. Especially in such cases where semantic structures are responsible for result generation, it can be a very time-consuming and tedious task to identify the right results for the individual search process. So it is important to provide an adequate presentation that allows users to unambiguously assess the retrieved results and enables them to comprehend why specific results are considered as relevant. To offer this kind of feedback the proposed approach presents related resources that are responsible for result retrieval and their related attributes in *expandable attribute nodes*. Thus each of these nodes contains resources from the semantic neighborhood of retrieved results that are of some relevance for the result generation. The labels of these expandable attribute nodes are derived from the conjoint concept in the semantic structure to indicate their meaning.

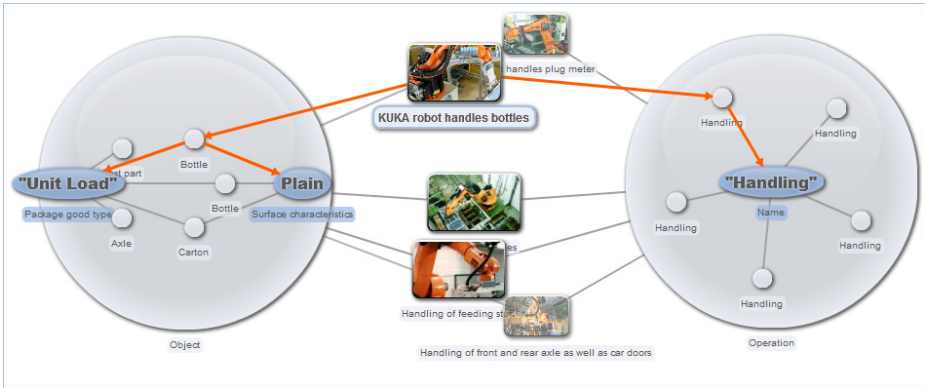


Fig. 3. The visualization of the semantic neighborhood combined with query-result-relations reveals why search results are retrieved during semantic search processing

Figure 3 shows an example of two hierarchically structured attribute nodes. The expandable attribute node on the left side contains five different objects (bottle, carton, etc.) and two elementary attribute nodes that are instantiated with given query terms. Each of the five object nodes is suspended between the inner attribute nodes to display the relations and relevancies to the current search. Finally, the relations between search result and query terms are represented by highlighting the path from a result over the related resource up to the instantiated attribute nodes. So the visualization indicates the ‘indirection’ in search result processing and reveals parts of the semantic neighborhood that are responsible for search result retrieval.

2.3 Mapping Results’ Relevance to Visual Properties

Beside the visualization of relations between query terms and retrieved results the proposed approach utilizes different similarity values that emerge during the result retrieval process for improving the visualization:

- *Partial similarities* are a measure between attribute nodes and results that represent the relevance of a retrieved resource to a given query term (encapsulated in an attribute node).
- *Result similarities* are aggregated values of all partial similarities that correspond to the overall relevance of a retrieved result.

In order to make the optimum use of these values, each similarity is mapped to specific visual properties like length, color and size that can be preattentively perceived [9]. On the one hand the size and color intensity of result nodes are adjusted according to the result similarity. Thereby the resource that has the highest overall similarity for a specific search query is presented most conspicuous whereas resources with minor similarities are visualized less notable (Figure 1). On the other hand partial similarities are used to adapt the weights of edges between results and attribute

nodes. This results in different lengths of the visible connections and indicates the relevance between specific query terms and search results.

2.4 Visual Feedback for Query Enhancement

Several studies revealed that it is a common search strategy for the user to first issue a general query, then review a few results, and if the desired information is not found, to reformulate or to enhance the query [4, 10, 11]. Transferred to the presented visualization, this refinement strategy corresponds to substitutions or reassignments of attribute values because these are directly related to the terms of the current search query. On the one hand the instantiation of further attributes defines a more specific search condition and on the other hand the removal of attribute values results in wider-ranged search spaces. In contrast to commonly used search user interfaces, the influence of changing search conditions is immediately visible in the visualization. The representation of query-result-relations reveals which of the current search results fulfill new conditions (Figure 4) and provides an immediate visual feedback for the query refinement task.

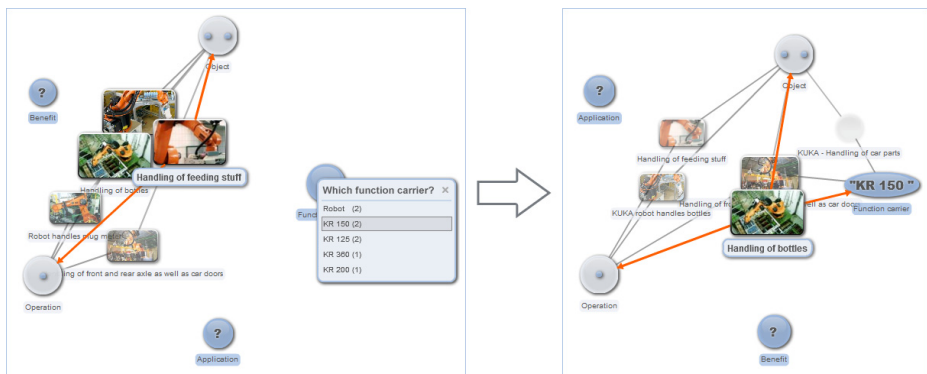


Fig. 4. The visual recommendation of additional attributes and possible terms for query enhancement offers a visual tool for narrowing search results

To ensure that users are aware of additional attributes the visualization recommends attributes that are not instantiated by the given query but related with the current result set. These recommendations are visualized as additional attribute nodes and labeled with a question mark to encourage users to instantiate them for narrowing their search. The size of the recommended attribute nodes is mapped to their influence to the current result set. So attribute nodes whose instantiation will cause major changes of the result set are represented larger than attribute nodes whose instantiation will only affect smaller parts. By selecting a specific recommendation, users are able to select different values for instantiating the attribute node and narrowing their retrieved results (Figure 4).

3 Application Scenario

The developed visualization approach is applied in the field of mechanical engineering and automation technology where highly complex processes and diverse user groups are involved. The processes in these domains range from initial development over construction and production steps up to sales and consumer services. So the same resource is treated in different contexts and various software systems. To ensure that each employee in these complex processes has access to the appropriate information at the right time, semantic technologies are used to link different knowledge bases and to provide a novel way to access information.

In this context the objective of the visualization is to provide a homogeneous access to the combined knowledge base for a variety of users. The focus of the investigation is predominantly on providing more transparency in search processing and to offer a tool that enables users to unambiguously assess the results of individual search processes. Additionally the aspect of recommending further dimensions for improving and narrowing the result set plays a major role in the investigation of the introduced visualization tool. In the current state the visualization is fully integrated in the search platform⁶.

4 Evaluation

For evaluating our approach we performed a user study in which we compared the visualization with a common list presentation (Figure 5). The study is mainly focused on answering the question whether our visualization approach can support the user in assessing search results and if our approach satisfies the needs of searchers. For verification of our assumption we investigated the task completion time and formulated the following hypothesis:

- *H1: There is a difference in task completion time between the list presentation and the visualization in assessing search results.*

Additionally to the task completion time we measured the user satisfaction as a subjective evaluation criterion.

4.1 Experimental Design

According to the hypothesis that contains one independent variable with two different conditions (list presentation and visualization) the design of our experiment is based on a basic design [12]. Additionally, we decided to use a within-group design for our experiment where each participant accomplishes the given tasks in each

⁶ Demonstration is available at <http://athena.igd.fraunhofer.de/Processus/semavis.html> Note that the knowledgebase of the online demonstrator is currently only available in German and contains only selected resources. Possible queries for demonstration are 'kuka roboter bauindustrie', 'glattes stückgut handhaben' and 'glas transportieren'.

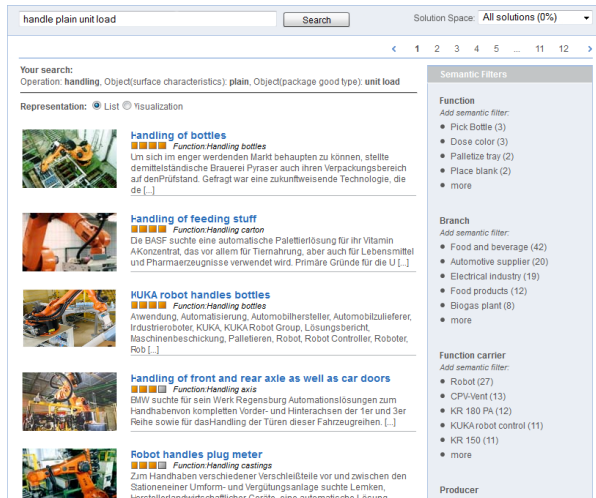


Fig. 5. List presentation of search results used for the evaluation

condition (in this case the different user interfaces). In contrast to between-group designed experiments, in within-group designs less participants are needed and individual differences between the participants are isolated more effectively [12]. Possible learn-effects when switching between conditions are controlled by a systematic randomization of condition- and task-ordering. Furthermore participants were advised to disregard the knowledge from previous conditions and to explicitly show the solution of tasks by means of elements in the user interface.

Altogether the experiment contains three tasks that had to be accomplished from every participant with both conditions (list presentation and visualization). Because the focus of the evaluation is the comparison of two different user interfaces and not the investigation of the whole search process, we were able to pre-assign the query terms for every task. So every participant retrieves the same results for every task and thus also the same visual representation and the evaluation outcome is not influenced by other factors.

In the first task participants had to identify the relations between each search result and the terms of the given query. The second task was of the same type as the first task with the difference that the result contains hierarchical structured attributes instead of only flat attributes. In the third task participants had to identify the most relevant item for a specific search situation. To ensure that the solution could be found in each condition, we performed several pretests. We also ensured that each participant gets the same visual presentation for each task and condition. The time limit for each task was set to three minutes. If a wrong answer was given or a participant could not solve a task, the completion time of the task was also set to three minutes.

4.2 Procedure

Altogether 17 participants, mainly graduates and students attended the evaluation. The average participant was between 24 and 29 years old. The participants were mainly involved in computer science ($M = 4.65$; $SD = 0.6$)⁷ and had no previous knowledge of the engineering domain. After a general introduction to the user study and an explanation of the procedure and tasks, participants got a brief introduction to both systems in systematically randomized ordering. Both systems were queried with a reference query and participants had the chance to ask questions about the systems. After each task participants had to rate their overall satisfaction with the system on a scale from 1 to 9 and three additional questions concerning their subjective opinion of the system on a Likert scale from 1 (strongly disagree) to 5 (strongly agree). After participants had completed all tasks, they had to answer a brief demographic questionnaire.

4.3 Results

Figure 6 shows the average task completion times for each of the three tasks and both conditions. The direct comparison of the average task completion times reveals that participants performed better with our visualization approach ($avg(t) = 51.3$ sec; $SD = 25.8$) compared to the list presentation ($avg(t) = 88.1$ sec; $SD = 30.1$). A paired-samples t-test also suggests that there is a significant difference in the task completion time between the group who used the list presentation and the group who used our visualization approach ($t(50)=7.8028$, $p<0.05$).

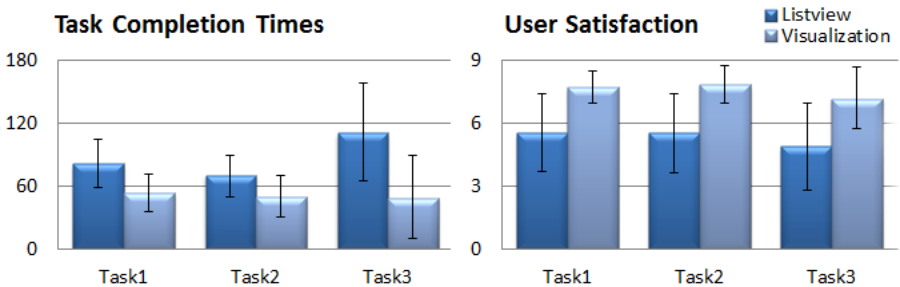


Fig. 6. Left: Task completion times. Right: Average user satisfaction

Hence the null hypothesis is refuted and the alternative hypotheses confirmed. The comparison of means also indicates that users performed significantly faster with the visualization approach compared to the list presentation. So we can proceed from the assumption that visualizing search results taking semantic information into account has a positive effect on the efficiency when assessing search result relevance.

⁷ Measured on a five point scale (5 = very much experience; 1 = very little experience) in the demographic part of the questionnaire.

The evaluation of satisfaction ratings indicates that participants feel more comfortable with our visualization approach instead of the commonly used list presentation. The list visualization obtained an average rating of 5.31 with a standard deviation of 1.91 whereas the visualization obtained an average rating of 7.57 and a standard deviation of 1.10. Additionally, the result of the question “Would you use the system in the future for similar searching tasks?” confirms the assumption that users prefer the visualization to the list presentation (list: $M = 3.14$; $SD = 0.87$; visualization: $M = 4.25$; $SD = 0.77$)⁸.

5 Related Work

Although the objective of semantic technologies was not focused on presenting semantics to end-users, there are several other approaches that benefit thereby. SemaPlorer [13] is an interactive application that allows users to visualize the search results of multiple semantics data sources. The user interface of SemaPlorer also provides a geographic visualization and a media view for visualizing geospatially annotated data and picture galleries respectively. However, this approach is mainly focused on combining search results from different heterogeneous knowledge bases and faceting the search by predefined facets. The Relfinder interface [14] supports users in interactively discovering relations between resources in semantic knowledge spaces. Users can prompt two or more resources and the relations between them are shown in a graph-based visualization. Although this approach demonstrates the benefit of communicating semantic knowledge to users, it is strictly limited to relation discovery between two or more resources.

There are also different approaches for using information visualization techniques for search user interfaces. To name only a few, the Microsoft Academic Search interface [15] incorporates geographic, graph-based and temporal visualization techniques for exploring publications or authors and offers also an stacked area chart for analyzing trends in the field of computer science. SkylineSearch [16] is a search interface that supports life science researchers in performing scientific literature search. It leverages semantic annotations to visualize search results in a scatterplot plotting relevance against publication date. Even though semantic annotations are used for search processing and estimating relevance values, semantic knowledge is not directly presented to the user. The WebSearchViz [17] is an approach for visualizing web search results based on the metaphor of the solar system. It offers users the possibility to observe the semantic relevance between a query and a web search result by the spatial proximity and distance between objects. However the system does not visualize semantic interpretations of search results or semantic structures.

Another commonly used and useful approach for visualizing result relevancy is the term highlighting technique [18] where the terms of the given query are highlighted in the surrogates of search result lists. For example the BioText System [19] represents

⁸ Measured on a five point Likert scale.

beside extracted figures from relevant articles, query terms highlighted in the title and boldfaced in the text excerpts for communicating reasons the particular results were retrieved. Even though term highlighting can be useful for improving search result list presentations, it does not reveal the semantic interpretation of search results and prevent users from scanning the whole result list for getting an overview.

6 Discussion

The introduced approach was applied and evaluated in the field of mechanical engineering and automation technology. Although this domain contains highly complex processes and different kinds of heterogeneous users, domain experts were able to semantically design it and build a comprehensive model that enables different stakeholders the access to heterogeneous resources. In such well-defined domains, aspects like data diversity, user roles and processes are in some way controllable and the data access methods can be accurately aligned to specific tasks of the stakeholder. The results of the evaluation showed that the proposed visualization approach performed very well in the present domain. Nevertheless, further investigations are needed to prove if the proposed approach is also transferable to other domains and if it can be seamlessly integrated in semantic web search engines.

Currently, most search user interfaces are based on result list presentations and usually show the titles and surrogates of the results. Cause of the public's great familiarity with this commonly used search result presentation, there is a certain degree of risk with the introduction of a novel approach in user interfaces. Even if novel approaches provide a variety of extended features and easier information access, the success of each innovation in user interfaces is measured by the acceptance of the users. Although the results of the evaluation show that the introduced visualization approach performed well in a controlled experimental environment and users are convinced of its benefits, there is still the need to prove if visualization techniques will be applicable in web search engines. However, current trends show an increased use of information visualization techniques in search user interfaces.

7 Conclusion and Future Work

In this paper we introduced a novel approach for visualizing search results in semantic knowledge bases. The results of the evaluation showed that the utilization of semantic information in search results visualization successfully fosters search result comprehension and supports user in assessing retrieved resources. Also the approach performed well for presenting different semantic interpretations of query terms and query-result-relations respectively. The visual recommendation of novel dimensions and immediate visual feedback for query refinement additionally fosters the common search strategies of users and offers more transparency in search result processing.

For future work we plan the extension of query refinement features. In particular we plan to implement the removal and change of attribute values that is not included

in the current version. Furthermore the multiple instantiation of attributes may be a useful extension of the introduced concept.

Acknowledgements. This work has been carried out within the Core-Technology Cluster (Innovative User Interfaces and Visualizations) of the THESEUS research program, partially funded by the German Federal Ministry of Economics and Technology. We thank H. J. Hesse, R. Traphöner and C. Dein (THESEUS PROCESSUS, Attensity Europe GmbH) for the inspiring discussions, the provision of the data and the support during the development of the data connection. We are also grateful to all participants that spend their time participating in the evaluation.

References

1. Shadbolt, N., Berners-Lee, T., Hall, W.: The Semantic Web Revisited. *IEEE Intelligent Systems* 21(3), 96–101 (2006)
2. Fernandez, M., Lopez, V., Sabou, M., Uren, V., Vallet, D., Motta, E., Castells, P.: Semantic Search Meets the Web. In: 2008 IEEE International Conference on Semantic Computing, pp. 253–260 (2008)
3. Cutrell, E., Robbins, D., Dumais, S., Sarin, R.: Fast, flexible filtering with phlat. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 261–270 (2006)
4. Hearst, M.A.: *Search User Interfaces*. Cambridge University Press (2009)
5. White, R.W., Bilenko, M., Cucerzan, S.: Studying the Use of Popular Destinations to Enhance Web Search Interaction. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 159–166 (2007)
6. Stab, C., Breyer, M., Nazemi, K., Burkhardt, D., Hofmann, C., Fellner, D.W.: SemaSun: Visualization of Semantic Knowledge based on an improved Sunburst Visualization Metaphor. In: Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2010, pp. 911–919. AACE, Chesapeake (2010)
7. Stab, C., Nazemi, K., Fellner, D.W.: SemaTime - Timeline Visualization of Time-Dependent Relations and Semantics. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Chung, R., Hammound, R., Hussain, M., Kar-Han, T., Crawfis, R., Thalmann, D., Kao, D., Avila, L. (eds.) ISVC 2010, Part III. LNCS, vol. 6455, pp. 514–523. Springer, Heidelberg (2010)
8. Nazemi, K., Breyer, M., Hornung, C.: SeMap: A Concept for the Visualization of Semantics as Maps. In: Stephanidis, C. (ed.) UAHCI 2009, Part III. LNCS, vol. 5616, pp. 83–91. Springer, Heidelberg (2009)
9. Ward, M., Grinstein, G., Keim, D.: *Interactive Data Visualization: Foundations, Techniques, and Applications*. A. K. Peters, Ltd., Natick (2010)
10. Jansen, B.J., Spink, A., Pedersen, J.O.: A Temporal Comparison of Altavista Web Searching. *Journal of the American Society for Information Science and Technology* 56(6), 559–570 (2005)
11. Jansen, B.J., Spink, A., Koshman, S.: Web Searcher Interaction with the Dogpile.com Metasearch Engine. *Journal of the American Society for Information Science and Technology* 58(5), 744–755 (2007)

12. Lazar, J., Feng, J.H., Hochheiser, H.: *Research Methods in Human - Computer Interaction*. John Wiley & Sons (2010)
13. Schenk, S., Saathoff, C., Staab, S., Scherp, A.: SemaPlorer - Interactive Semantic Exploration. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web* 7(4), 298–304 (2009)
14. Heim, P., Lohmann, S., Stegemann, T.: Interactive Relationship Discovery via the Semantic Web. 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. 303–317. Springer, Heidelberg (2010)
15. Microsoft Academic Search, <http://academic.research.microsoft.com>
16. Stoyanovich, J., Lodha, M., Mee, W., Ross, K.A.: SkylineSearch: semantic ranking and result visualization for pubmed. In: *Proceedings of the 2011 International Conference on Management of Data, SIGMOD 2011*, pp. 1247–1250 (2011)
17. Nguyen, T., Zhang, J.: A Novel Visualization Model for Web Search Results. *IEEE Transactions on Visualization and Computer Graphics* 12(5), 981–988 (2006)
18. Aula, A.: Enhancing the readability of search result summaries. In: *Proceedings of HCI 2004*, pp. 6–10 (2004)
19. Hearst, M.A., Divoli, H., Guturu, A., Ksikes, P., Nakov, M.A., Wooldridge, J.Y.: BioText Search Engine: beyond abstract search. *Bioinformatics* 23(16), 2196 (2007)

Evaluating Scientific Hypotheses Using the SPARQL Inferencing Notation

Alison Callahan¹ and Michel Dumontier^{1,2,3,*}

¹Department of Biology, ²School of Computer Science, ³Institute of Biochemistry,
Carleton University, Ottawa, Canada
acallaha@connect.carleton.ca,
michel_dumontier@carleton.ca

Abstract. Evaluating a hypothesis and its claims against experimental data is an essential scientific activity. However, this task is increasingly challenging given the ever growing volume of publications and data sets. Towards addressing this challenge, we previously developed HyQue, a system for hypothesis formulation and evaluation. HyQue uses domain-specific rulesets to evaluate hypotheses based on well understood scientific principles. However, because scientists may apply differing scientific premises when exploring a hypothesis, flexibility is required in both crafting and executing rulesets to evaluate hypotheses. Here, we report on an extension of HyQue that incorporates rules specified using the SPARQL Inferencing Notation (SPIN). Hypotheses, background knowledge, queries, results and now rulesets are represented and executed using Semantic Web technologies, enabling users to explicitly trace a hypothesis to its evaluation as Linked Data, including the data and rules used by HyQue. We demonstrate the use of HyQue to evaluate hypotheses concerning the yeast galactosegene system.

Keywords: hypothesis evaluation, semantic web, linked data, SPARQL.

1 Introduction

Developing and evaluating hypotheses in the context of experimental research results is an essential activity for the life scientist, but one which is increasingly difficult to carry out manually given the ever growing volume of publications and data sets[1]. Indeed, biologists perceive that the predominant challenge in research is to “locate, integrate and access” the vast amounts of biological data resulting from small- and large-scale experiments[2]. Life sciences resources for the Semantic Web, such as Bio2RDF[3] and the growing number of bio-ontologies offer the potential to develop systems that consume these resources and computationally reason over the knowledge they contain to infer new facts[4-6]and answer complex questions[7].

With the diversity of research claims that exist in such large resources, there is also the potential for statements to contradict one another. Formally exploring the out-

* Corresponding author.

comes of relying on different sets of research claims to assess a hypothesis is necessary to not only confer confidence in the hypothesis evaluation methodology (whether manual or automatic), but also to provide evidence for the likelihood of one interpretation of results compared to another. Previous research efforts that have aimed at formally evaluating scientific data in the context of hypotheses include HYPGENE[8, 9], HinCyc[10], GenePath[11] and Adam the Robot Scientist[12, 13]. Each of these projects use application-specific representations for data and the rules used to assess this data, making their extension to new domains, as well as their comparison and performance evaluation difficult.

Towards addressing the challenge of integrating experimental knowledge with biological hypotheses, we previously developed HyQue[14, 15]. HyQue uses Semantic Web standard languages (RDF/OWL) to represent hypotheses and data, SPARQL queries to retrieve data, and domain-specific rulesets to evaluate hypotheses against this data. While HyQue uses rulesets based on well understood scientific principles[16, 17], finer grained evaluations would require the exclusion or inclusion of additional rules. Problematically, HyQue's domain-specific evaluation rules were hard-coded, which made it implausible for users to construct custom rule sets for hypothesis evaluation.

In this paper, we describe an extension of HyQue that uses evaluation rules specified using the SPARQL Inferencing Notation (SPIN) in place of hardcoded rules. SPIN is a W3C member submission¹ rule language whose scope and expressivity are defined by SPARQL. Thus, SPIN rules are SPARQL queries which can not only be used to assert new facts, but also used to infer OWL class membership for non-hierarchical class membership axioms². Moreover, SPIN rules can be serialized into RDF, and hence can become part of a system that maintains provenance concerning calculations and inferences.

In this new version of HyQue, hypotheses, background knowledge, queries, results and now evaluation rulesets are represented and executed using Semantic Web technologies. Domain specific rules for evaluating experimental data in the context of a hypothesis are now maintained independently of the system rules that are used to calculate overall hypothesis evaluation scores. We demonstrate these features by evaluating hypotheses about the galactose gene system in yeast[16]. HyQue enables users to explicitly trace a hypothesis to its evaluation, including the data and rules used. In addition to making the hypothesis evaluation methodology transparent and reproducible (essential qualities for good e-science), this allows scientists to discover experimental data that support a given hypothesis as well as explore new and potentially uncharacterized links between multiple research outcomes. A unique strength of HyQue is that its design is not dependent upon a specific biological domain, and the assumptions encoded in its hypothesis evaluation rules are changeable and maintained separately from the evaluation system. As our understanding of biological systems evolves and improves through research, the way HyQue evaluates hypotheses, as well as the facts and data it uses, can evolve as well.

¹ <http://www.w3.org/Submission/2011/SUBM-spin-overview-20110222/>

² <http://www.w3.org/Submission/2011/SUBM-spin-modeling-20110222/>

2 Methods

2.1 Overview

HyQue evaluates hypotheses (and assigns an evaluation score) by executing SPIN rules over the pertinent knowledge extracted from a HyQue Knowledge Base (HKB). A hypothesis is formulated as a logical expression in which elements of the hypothesis correspond to biological entities of interest. HyQue maps the hypothesis, expressed using terminology from the HyQue ontology³, to the relevant SPIN rules, which execute SPARQL queries to retrieve data from the HKB. Finally, HyQue executes additional SPIN rules over the extracted data to obtain a quantitative measure of hypothesis support. Figure 1 provides a graphical overview of HyQue.

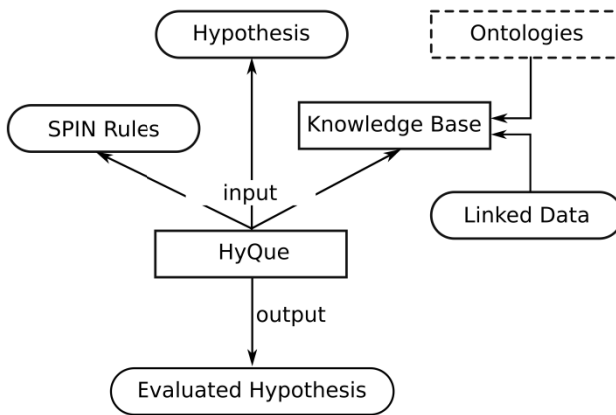


Fig. 1. HyQue uses SPIN rules to evaluate a hypothesis over RDF linked data and OWL ontologies. The dashed rectangle represents OWL ontologies. Rounded rectangles are RDF resources.

2.2 HyQue Hypothesis Model

A HyQue *hypothesis* may be composed of one or more *propositions* that *specify events* related to each other by AND/OR operators. Events must have an agent (an entity executing an action) and a target (the object of the action), and can optionally have a physical location, a physical operator (e.g. ‘binding’), a logical operator (e.g. ‘repression’ or ‘activation’) and a perturbation context (in the case of genes and proteins). HyQue maps these events to SPARQL queries through a SPIN rule, and subsequently executes them over the HyQue Knowledge Base. HyQue currently supports the following kinds of events [14, 15]:

³ The HyQue ontology, linked data, and SPIN rules are available at the project website: <http://hyque.semanticscience.org>

1. protein-protein binding
2. protein-nucleic acid binding
3. molecular activation
4. molecular inhibition
5. gene induction
6. gene repression
7. transport

2.3 HyQue Knowledge Base (HKB)

A HyQue Knowledge Base (HKB) consists of RDF data, RDFS-based class hierarchies and/or OWL ontologies. For demonstration purposes, our HKB consists of an RDF version of the galactose (GAL) gene network in yeast [17], an extended version of the Bio2RDF compatible yOWL knowledge base [7, 15] and the following bio-ontologies (for the listed entities):

- **Gene Ontology (GO):** cellular components, events (*e.g.* 'nucleus', 'positive regulation of gene expression')
- **Evidence Codes Ontology (ECO):** the type of evidence supporting an event (*e.g.* 'electronic annotation', 'direct assay')
- **Sequence Ontology (SO):** event participants (*e.g.* 'gene')
- **Chemical Entities of Biological Interest (CHEBI) Ontology:** event participants (*e.g.* 'protein', 'galactose')

All Linked Data (encoded using RDF) and ontologies (encoded using OWL) that comprise the HKB are available at the project website.

2.4 The HyQue Scoring System

HyQue uses rules to calculate a numerical score for a hypothesis based on the degree of support the hypothesis has from statements in the HKB. HyQue first attempts to identify statements about experimentally verified events in the HKB that have a high degree of matching to a hypothesized event, and then assesses these statements using domain specific rules to assign a score to the hypothesized event. If there is a statement about an experimentally reported GAL gene/protein interaction in the HKB that exactly matches a hypothesized event, then that event will be assigned a maximum score when it is evaluated by HyQue. In contrast, if a hypothesized event describes an interaction between a protein A and a protein B but there is a statement in the HKB asserting that protein A does *not* interact with protein B, then the hypothesis will be assigned a low score based on the negation of the hypothesized event by experimental data. Different HyQue rules add or subtract different numerical values based on whether the relevant experimental data has properties that provide support for a hypothesized event. For instance, if an event is hypothesized to occur in a specific cellular compartment *e.g.* nucleus, but the HKB only contains a statement that such an event takes place in a different cellular component *e.g.* cytoplasm, then a rule could be formulated

such that the hypothesis, while not directly supported by experimental evidence, will be penalized less than if the event had been asserted to not take place at all.

Based on such scoring rules, each event type has a maximum possible score. When a hypothesized event is evaluated by HyQue, it is assigned a normalized score calculated by the sum of the output of the relevant rule(s) divided by the maximum possible score. In this way, if an event has full experimental support, it will have an overall score of 1, while if only some properties of the hypothesized event are supported by statements in the HKB it will have a score between 0 and 1.

Overall proposition and hypothesis scores are calculated by additional rules based on the operators that relate events. If a proposition specifies ‘event A’ OR ‘event B’ OR ‘event C’ then the maximum event score will be assigned as the proposition score, while if the ‘AND’ operator was used, the mean event score will be assigned as the proposition score. Using the mean reflects the relative contribution of each event score while still maintaining a normalized value between 0 and 1. Similar rules are used to calculate an overall hypothesis score based on proposition scores.

HyQue uses SPIN to execute rules that reflect this scoring system.

2.5 HyQue SPIN Rules

HyQue uses two types of rules to evaluate hypotheses: *domain specific rules* that depend on the subject of the hypothesis (in this case, gene regulation) and *system rules* that define how to combine the output of domain specific rules in order to determine an overall hypothesis evaluation score. These rules are defined separately using SPIN and can be changed independently of each other.

HyQue system rules describe how to calculate event, proposition and overall hypothesis scores based on the structure and content of the hypothesis. For example, the following rule (modified with single quoted labels for illustrative purposes) generates four statements that assert the relationship between a HyQue hypothesis (any instance of the class `hyque:HYPOTHESIS_0000000`) and its evaluation.

```
CONSTRUCT {
  ?this 'has attribute' ?hypothesisEval .
  ?hypothesisEval a 'evaluation'.
  ?hypothesisEval 'obtained from' ?propositionEval .
  ?hypothesisEval 'has value' ?hypothesisEvalScore .
} WHERE {
  ?this 'has component part' ?proposition .
  ?proposition 'has attribute' ?propositionEval .
  BIND(:calculateHypothesisScore(?this) AS ?hypothesisEvalScore) .
  BIND(URI(fn:concat(afn:namespace(?this), afn:localname(?this), "_",
    "evaluation")) AS ?hypothesisEval) .
}
```

This SPIN rule states that a HyQue hypothesis (`hyque:HYPOTHESIS_0000000`) will be related to a new attribute of type ‘evaluation’ (`hyque:HYPOTHESIS_0000005`) by the ‘has attribute’ (`hyque:HYPOTHESIS_0000008`) object property. The numeric value of this evaluation is specified using the ‘has value’ (`hyque:HYPOTHESIS_0000013`) datatype property. Since the evaluation of the hypothesis comes from evaluating the propositional parts, these are related with the ‘is obtained from’ (`hyque:HYPOTHESIS_0000007`) object

property. The SPARQL variable ‘?this’ has a special meaning for SPIN rules, and refers to any instance of the class the rule is linked to. SPIN rules are linked to classes in the HyQue ontology using the `spin:rule` predicate.

This hypothesis rule uses another rule, `calculateHypothesisScore`, to calculate the hypothesis score, and the output of executing this rule is bound to the variable `?hypothesisEvalScore`. Note that the hypothesis rule is constrained to a HyQue hypothesis that ‘has component part’ (`hyque:HYPOTHESIS_0000010`) some ‘proposition’ (`hyque:HYPOTHESIS_0000001`) that ‘has attribute’ a proposition evaluation. In this way HyQue rules are chained together – when one rule is executed, all the rules it depends on are executed until no new statements are created. In this case, because a hypothesis evaluation score requires a proposition evaluation score, when the hypothesis evaluation rule is executed, the HyQue SPIN rule for calculating a proposition score is executed as well. Each proposition evaluation is asserted to be ‘obtained from’ the event evaluations corresponding to the event(s) specified by (`hyque:HYPOTHESIS_0000012`) the proposition. Each event evaluation is also asserted to be ‘obtained from’ the scores determined for each event property (the agent, target, location *etc.*) and the statements in the HKB the scores are based on.

Domain specific rules for HyQue pertain to the domain of interest. An example of a domain specific rule is `calculateActivateEventScore` corresponding to the following SPARQL query:

```
SELECT ?activateEventScore
WHERE {
  BIND (:calculateActivateAgentTypeScore(?arg1)
        AS ?agentTypeScore) .
  BIND (:calculateActivateTargetTypeScore(?arg1)
        AS ?targetTypeScore) .
  BIND (:calculateActivateLogicalOperatorScore(?arg1)
        AS ?logicalOperatorScore) .
  BIND (:penalizeNegation(?arg1) AS ?negationScore) .
  BIND (3 AS ?maxScore) .
  BIND (((?agentTypeScore + ?targetTypeScore) +
        ?logicalOperatorScore) + ?negationScore) /
        ?maxScore) AS ?activateEventScore) .
}
```

In this rule, a numeric score (`?activateEventScore`) is calculated from the sum of a set of outputs from other sub-rules divided by the maximum score possible (in this case, 3). This rule uses a special variable `?arg1`, which corresponds to any entities linked using the SPIN `sp:arg1` predicate. This special variable is selected by specifying a `spin:constraint` on the rule, which states that any variable passed to the rule when it is called can be referred to within the rule to by ‘?arg1’. For example, if the rule were called by including `calculateActivateEventScore(?data)` in a SPARQL query WHERE statement, `?data` will be the variable referenced by `?arg1` in the rule definition.

The sub-rule `calculateActivateLogicalOperatorScore` determines a score for the type of logical operator specified in a HyQue hypothesis based on domain specific knowledge about the GAL gene network. This rule corresponds to the following SPARQL query:

```

SELECT ?score
WHERE {
  ?arg1 'has logical operator' ?logical_operator .
  BIND (IF((?logical_operator = 'positive regulation of molecular
function'), 1, -1) AS ?score) .
}

```

Thus, if the logical operator specified in a hypothesis event is of type ‘positive regulation of molecular function’ (GO:0044093) the rule will return 1, and otherwise the rule will return -1. The `calculateActivateEventScore` rule is composed of several sub-rules of this format. HyQue uses similar rules for each of the seven event types listed in section 2.2 to evaluate hypotheses.

SPIN rules were composed using the free edition of TopBraid Composer 3.5. HyQue executes SPIN rules using the open source SPIN API 1.2.0 and Jena 2.6.4.

2.6 Executing HyQue SPIN Rules over the HKB

To execute the HyQue SPIN rules over an input hypothesis using data from the HKB, a Java program was written with the open source SPIN API (version 1.2.0) and the Jena API (version 2.6.4). Users can submit a hypothesis to the program via a servlet available at <http://hyque.semanticscience.org>. The servlet returns the RDF-based hypothesis evaluation.

3 Results

HyQue currently uses a total of 63 SPIN rules to evaluate hypotheses. 18 of these are system rules, and the remaining 45 are domain specific rules that calculate evaluation scores based on well understood principles of the GAL gene network in yeast as described in section 2.5. These rules have been used to evaluate 5 representative hypotheses about the GAL domain, one of which is presented in detail in section 3.1.

3.1 Evaluating a Hypothesis about GAL Gene Induction and Protein Inhibition

The following is a natural language description of a hypothesis about the GAL gene network that has been evaluated by HyQue. Individual events are indicated by the letter ‘e’, followed by a number to uniquely identify them. Events are related by the AND operator in this hypothesis, while the two sets of events (typed as propositions in the HyQue hypothesis ontology) are related by the OR operator.

(Gal4p induces the expression of GAL1	<i>e1</i>
AND Gal3p induces the expression of GAL2	<i>e2</i>
AND Gal4p induces the expression of GAL7)	<i>e3</i>
OR	
(Gal4p induces the expression of GAL7	<i>e4</i>
AND Gal80p induces the expression of GAL7	<i>e5</i>
AND Gal80p does not inhibit the activity of Gal4p	
when GAL3 is over-expressed)	<i>e6</i>

Two domain specific SPIN rules were executed to evaluate this hypothesis: `calculateInduceEventScore` for *e1-e5* and `calculateInhibitEventScore` for *e6*, in conjunction with system rules to calculate overall proposition and hypothesis scores based on the event scores.

By identifying and evaluating statements in the HKB that experimentally support *e1*, the `calculateInduceEventScore` rule assigns *e1* a score of 4 out of a maximum score of 5 (see Table 1). This corresponds to a normalized score of 0.8. Similarly, events 2-5 also receive a score of 0.8. The `calculateInhibitEventScore` rule assigns event 6 a score of 1 based on comparable scoring rules. Therefore, the proposition specifying *e4*, *e5* and *e6* receives a higher score (0.87 – the mean of the individual event scores) than the proposition specifying *e1*, *e2* and *e3* (with a mean score of 0.8). Because the two propositions were related by the OR operator, the hypothesis is assigned an overall score that is the maximum of the two proposition scores, in this case, a value of 0.87.

Table 1. SPIN rules executed to evaluate a hypothetical GAL gene induction event, their outcomes, and contribution to an overall hypothesis score assigned by HyQue

SPIN Rule	Rule output	Score
<code>penalizeNegation</code>	Event is not negated	0
<code>calculateInduceAgentTypeScore</code>	Actor is a 'protein' (CHEBI:36080)	+1
<code>calculateInduceTargetTypeScore</code>	Target is a 'gene' (SO:0000236)	+1
<code>calculateInduceLogical OperatorScore</code>	Logical operator is 'induce' (GO:0010628)	+1
<code>calculateInduceAgentFunction Score</code>	Actor does not have 'transcription factor activity' (GO:0003700)	0
<code>calculateInduceLocationScore</code>	Location is 'nucleus' (GO:0005634)	+1

The complete HyQue evaluations of this hypothesis as well as that of four additional hypotheses are available as RDF at the project website.

3.2 Changing a Domain Specific Rule Affects Hypothesis Evaluation

The `calculateInhibitEventScore` used to evaluate event 6 in section 3.1 in its current form does not take into account the physical location of the event participants. In other words, the score does not depend on data describing where the event participants are known (or not) to be located in the cell. However, some experimental evidence suggests that physical location in the context of an inhibition event plays an important role. Specifically, the inhibition of Gal4p activity by Gal80p is known to take place in the nucleus, yet this inhibition is interrupted when Gal80p is bound by Gal3p, which is typically found in the cytoplasm[18].

The effect of changing the `calculateInhibitEventScore` rule to require that all event participants be located in the nucleus to achieve a maximum score (a reasonable assumption given published findings[19]) on the hypothesis in section 3.1 would be that the score for *e6* is reduced. This is because adding an additional

sub-rule (let us call it `calculateInhibitEventParticipantLocationScore`) would increase the maximum score, while experimental data in the HKB is not available to satisfy the conditions of this new sub-rule – there is not experimental data available about the location of the Gal4p or Gal80p proteins in the cell. More specifically, let us say that the maximum score possible for `calculateInhibitEventScore` with the new sub-rule is now 4, and that event 6 is therefore assigned a score of 0.75 (3/4) based on the output of this rule. This changes the overall hypothesis score in that the first proposition (specifying events 1-3) now has a higher mean score (0.8, *versus* 0.78 for the second proposition as calculated using the new rule), and thus this is assigned as the overall hypothesis score.

This example demonstrates how using a different domain specific rule affects an overall hypothesis evaluation, and how the effect can be traced to both the rule(s) used and the data the rules are executed over.

4 Discussion

Using SPIN rules to evaluate HyQue hypotheses has several advantages. While HyQue “version 1.0” used SPARQL queries to obtain relevant statements from the HKB, the scoring rules used to evaluate those statements were hard-coded in system code. HyQue’s SPIN evaluation rules can be represented as RDF, which allows the potential for users to query for HyQue rules that meet specific conditions, as well as potentially link to and aggregate those rules. In addition, users can create their own SPIN rules to meet specific evaluation criteria and augment existing HyQue rules to include them. In this way, different scientists may use the same data to evaluate the same hypotheses and arrive at unique evaluations depending on the domain principles encoded by the SPIN rules they use, as demonstrated in section 3.2. Encoding evaluation criteria as SPIN rules also ensures that the source of an evaluation can be explicitly stated, both in terms of the rules executed and the data the rules were executed over. This is crucial for formalizing the outcomes of scientific reasoning such that research conclusions can be confidently stated.

Separating HyQue system rules from the GAL domain specific rules highlights the two aspects of the HyQue scoring system. Specifically, HyQue currently encodes certain assumptions about how events in hypotheses may be related to one another, and how these relations are used to determine an overall hypothesis score, as well as domain specific assumptions about how to evaluate data in the context of knowledge about the GAL gene network. However, because assumptions about hypothesis structure are encapsulated by HyQue system rules, they may be changed or augmented without affecting the GAL domain specific rules, and *vice versa*. HyQue system rules can be extended over time to facilitate the evaluation of hypotheses that have fundamentally different structures than those currently presented as demonstrations. We envision a future iteration of HyQue where users can submit unique system and domain specific rules to use for evaluating hypotheses and in this way further research in their field by exploring novel interpretations of experimental data and hypotheses.

Similarly, it may be possible in future for HyQue users to select from multiple sets of evaluation rules and to compare the hypothesis evaluations that result.

Crafting SPIN rules requires knowledge of SPARQL, which, while being used in a number of life-science related projects[3, 5, 20-22], may present a barrier to some users. Similarly, representing hypotheses as RDF to submit to HyQue is not a trivial activity. To address the latter, we have developed an online form based system for specifying hypothesis details and converting them to RDF, available at the project website.

The Rule Interchange Format (RIF)⁴ is the W3C standard for representing and exchanging rules between rule systems. SPIN, a W3C member submission, has been identified as an effort complimentary to RIF[23] and because there is some discussion of RIF and RDF compatibility⁵, SPIN and RIF may become compatible if the RIF working group remains active⁶. HyQue provides a relevant use case and motivation for enabling such compatibility. Given that SPIN rules may be represented as RDF and executed over any RDF store using SPARQL (both W3C standards), however, and that the motivation of SPIN is specifically to execute SPARQL as rules, in the context of HyQue compatibility with RIF is not of immediate concern.

5 Conclusions

We present an extended version of HyQue that uses SPIN rules to evaluate hypotheses encoded as RDF, and makes the evaluation, including the data it is based upon, also available as RDF. In this way, users are able to explicitly trace a path from hypothesis to evaluation and the supporting experimental data, and *vice versa*. We have demonstrated how HyQue evaluates a specific hypothesis about the GAL gene network in yeast with an explanation of the scoring rules used and their outcomes. Evaluations of additional hypotheses, as well as HKB data and HyQue SPIN rules are available at <http://hyque.semantic-science.org>.

References

1. Neylon, C., Wu, S.: Article-Level Metrics and the Evolution of Scientific Impact. *PLoS Biology* 7(11), e1000242 (2009)
2. Howe, D., Costanzo, M., Fey, P., Gojobori, T., Hannick, L., Hide, W., Hill, D.P., Kania, R., Schaeffer, M., St Pierre, S., et al.: Big data: The future of biocuration. *Nature* 455(7209), 47–50 (2008)
3. Belleau, F., Nolin, M.-A., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: towards a mashup to build bioinformatics knowledge systems. *Journal of Biomedical Informatics* 41(5), 706–716 (2008)

⁴ <http://www.w3.org/TR/2010/NOTE-rif-overview-20100622/>

⁵ <http://www.w3.org/TR/2010/REC-rif-rdf-owl-20100622/>

⁶ <http://www.w3.org/Submission/2011/02/Comment/>

4. Tari, L., Anwar, S., Liang, S., Cai, J., Baral, C.: Discovering drug-drug interactions: a text-mining and reasoning approach based on properties of drug metabolism. *Bioinformatics* 26 (18), i547–i553
5. Nolin, M.A., Dumontier, M., Belleau, F., Corbeil, J.: Building an HIV data mashup using Bio2RDF. *Briefings in Bioinformatics* (2011)
6. Blonde, W., Mironov, V., Venkatesan, A., Antezana, E., De Baets, B., Kuiper, M.: Reasoning with bio-ontologies: using relational closure rules to enable practical querying. *Bioinformatics* 27(11), 1562–1568 (2011)
7. Villanueva-Rosales, N., Dumontier, M.: yOWL: an ontology-driven knowledge base for yeast biologists. *Journal of Biomedical Informatics* 41(5), 11 (2008)
8. Karp, P.D.: Artificial intelligence methods for theory representation and hypothesis formation. *Comput. Appl. Biosci.* 7(3), 301–308 (1991)
9. Karp, P.: Design Methods for Scientific Hypothesis Formation and Their Application to Molecular Biology. *Machine Learning* 12(1-3), 89–116 (1993)
10. Karp, P.D., Ouzounis, C., Paley, S.: HinCyc: a knowledge base of the complete genome and metabolic pathways of *H. influenzae*. In: *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, vol. 4, pp. 116–124 (1996)
11. Zupan, B., Bratko, I., Demsar, J., Juvan, P., Curk, T., Borstnik, U., Beck, J.R., Halter, J., Kuspa, A., Shaulsky, G.: GenePath: a system for inference of genetic networks and proposal of genetic experiments. *Artif. Intell. Med.* 29(1-2), 107–130 (2003)
12. King, R.D., Rowland, J., Oliver, S.G., Young, M., Aubrey, W., Byrne, E., Liakata, M., Markham, M., Pir, P., Soldatova, L., et al.: The automation of science. *Science* 324(5923), 85–89 (2009)
13. Soldatova, L., King, R.D.: Representation of research hypotheses. In: *Bio-Ontologies 2010: Semantic Applications in Life Sciences*, Boston, MA (2010)
14. Callahan, A., Dumontier, M., Shah, N.: HyQue: Evaluating hypotheses using Semantic Web technologies. In: *Bio-Ontologies: Semantic Applications in the Life Sciences*, Boston, MA (2010)
15. Callahan, A., Dumontier, M., Shah, N.H.: HyQue: evaluating hypotheses using Semantic Web technologies. *J. Biomed. Semantics* 2(suppl. 2), S3 (2011)
16. Ideker, T., Thorsson, V., Ranish, J.A., Christmas, R., Buhler, J., Eng, J.K., Bumgarner, R., Goodlett, D.R., Aebersold, R., Hood, L.: Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science* 292(5518), 929–934 (2001)
17. Racunas, S.A., Shah, N.H., Albert, I., Fedoroff, N.V.: HyBrow: a prototype system for computer-aided hypothesis evaluation. *Bioinformatics* 20(suppl. 1), i257–i264 (2004)
18. Bhat, P.J., Murthy, T.V.: Transcriptional control of the GAL/MEL regulon of yeast *Saccharomyces cerevisiae*: mechanism of galactose-mediated signal transduction. *Mol. Microbiol.* 40(5), 1059–1066 (2001)
19. Peng, G., Hopper, J.E.: Evidence for Gal3p's cytoplasmic location and Gal80p's dual cytoplasmic-nuclear location implicates new mechanisms for controlling Gal4p activity in *Saccharomyces cerevisiae*. *Mol. Cell Biol.* 20(14), 5140–5148 (2000)
20. Kobayashi, N., Ishii, M., Takahashi, S., Mochizuki, Y., Matsushima, A., Toyoda, T.: Semantic-JSON: a lightweight web service interface for Semantic Web contents integrating multiple life science databases. *Nucleic Acids Res.* 39(Web Server issue), W533–W540 (2011)

21. Chen, B., Dong, X., Jiao, D., Wang, H., Zhu, Q., Ding, Y., Wild, D.J.: Chem2Bio2RDF: a semantic framework for linking and data mining chemogenomic and systems chemical biology data. *BMC Bioinformatics* 11, 255 (2010)
22. Antezana, E., Kuiper, M., Mironov, V.: Biological knowledge management: the emerging role of the Semantic Web technologies. *Brief Bioinform.* 10(4), 392–407 (2009)
23. Polikoff, I.: Comparing SPIN with RIF (July 05, 2011), <http://topquadrantblog.blogspot.com/2011/06/comparing-spin-with-rif.html> (accessed December 7, 2011)

Declarative Representation of Programming Access to Ontologies

Stefan Scheglmann, Ansgar Scherp, and Steffen Staab

Institute for Web Science and Technologies
University of Koblenz-Landau, Germany
{schegi,scherp,staab}@uni-koblenz.de

Abstract. Using ontologies in software applications is a challenging task due to the chasm between the logics-based world of ontologies and the object-oriented world of software applications. The logics-based representation emphasizes the meaning of concepts and properties, i.e., their semantics. The modeler in the object-oriented paradigm also takes into account the pragmatics, i.e., how the classes are used, by whom, and why. To enable a comprehensive use of logics-based representations in object-oriented software systems, a seamless integration of the two paradigms is needed. However, the pragmatic issues of using logic-based knowledge in object-oriented software applications has yet not been considered sufficiently. Rather, the pragmatic issues that arise in using an ontology, e.g., which classes to instantiate in which order, remains a task to be carefully considered by the application developer. In this paper, we present a declarative representation for designing and applying programming access to ontologies. Based on this declarative representation, we have build OntoMDE, a model-driven engineering toolkit that we have applied to several example ontologies with different Characteristics.

1 Introduction

One of the most challenging issues in implementing Semantic Web applications is that they are built using two different technologies: object-oriented programming for the application logic and ontologies for the knowledge representation. Object-oriented programming provides for maintainability, reuseability and robustness in the implementation of complex software systems. Ontologies provide powerful means for knowledge representation and reasoning and are useful for various application domains. For accessing ontological knowledge from object-oriented software systems, there are solutions like ActiveRDF [8] and Jastor[1]. Most of these frameworks make use of the structural similarities of both paradigms, e.g., similar inheritance mechanisms and utilize simple solutions known from the field of object-relational mapping. But with the use of these existing tools some problems cannot be solved: Typically, the structural similarities lead to a one-to-one mapping between ontology concepts, properties and individuals and object-oriented classes, fields and objects, respectively. This leads to a data-centric

¹ <http://jastor.sourceforge.net/> last visit June 24, 2011.

object-oriented representation of the ontology which ignores the responsibility-driven [17] nature of object-orientation. It is up to the API developer to provide additional object-oriented layers which allow the use of the generated class representations. In addition, not all concepts and relations that must be defined in the ontology are useful in the object-oriented model. Again it is up to the API developer to provide proper encapsulations to hide such concepts from the application developer. Since this additional programming effort of the API developer relies on the one-to-one class representations of a specific ontology, changes in the ontology easily end up in excessive adaptation work of the API. In addition, as the experiences in the WeKnowIt-project show, new requirements and changes in the ontology may imply tedious and complex updates of the programming access to the logics-based representation. What is needed is a tool that comprehensively supports API developers in designing pragmatic programming access to ontological knowledge.

In this paper, we present a declarative representation for pragmatic access to ontological structures that supports the developer in building programmatic access to ontologies. We present OntoMDE, a Model-Driven Engineering toolkit for the generation of programming access to ontologies that is based on these declarative representations. OntoMDE supports the developer in building APIs adapted to concrete application needs. We define our problem and introduce a scenario and running example in the following section. In Section 3, we define the requirements for developing programming access to ontologies. Based on these requirements, we introduce our approach in Section 4. We have applied our approach at the examples of selected ontologies presented in Section 5. In Section 6, we discuss the related work, before we conclude the paper.

2 Scenario, Example and Problem

First, we present a scenario to motivate our work. Subsequently, an example ontology is introduced to demonstrate the problems of today's API generation tools conducting a one-to-one mapping. We compare the API resulting from the use of existing tools with an API that would be more natural to have in a purely object-oriented model.

2.1 Scenario: An Ontological Multimedia Annotation Framework

Jim works for a multimedia company and is responsible for the integration of knowledge-base access in an object-oriented media annotation framework. The media annotation framework should support the user in annotating multimedia content such as images or video clips. Jim shall use an ontology for representing annotated media as well as the multimedia annotations. He has not been involved in the design of the ontologies. His task is to define the programming interfaces to access and update the knowledge-base seamlessly from the application. He has to consider that further specializations toward domain-specific annotations could result in changes of the implementation.

2.2 Example: Ontology-Based Modeling of Multimedia Metadata

Figure 1(a) shows an excerpt of the ontology used by Jim to model the multimedia metadata. The example is based on the Multimedia Metadata Ontology (M3O) [13] for representing annotation, decomposition, and provenance information of multimedia data. It models the annotations of an image with an EXIF² geo-point wgs84:Point³ and a Foaf⁴ person foaf:Person as image creator. As we can see from the different namespaces, the m3o:Image, wgs84:Point and foaf:Person concepts and their superconcepts dul:InformationEntity, dul:Object and finally dul:Entity are defined in different ontologies. The inheritance and import relationships are shown in Figure 1(b), which is needed important for a proper API representation.

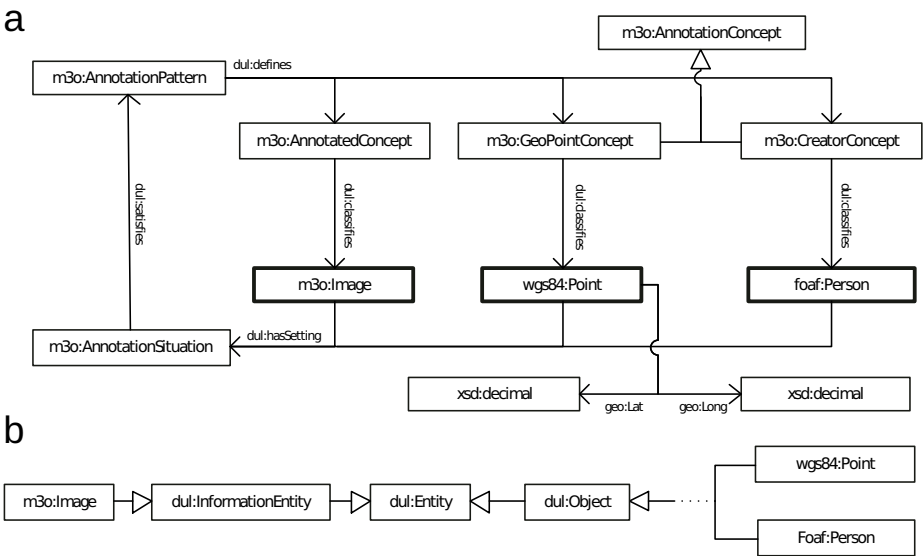


Fig. 1. Annotation of an Image with its Geo-location and Creator

2.3 Issues with APIs Provided by Existing Frameworks

Jim uses a simple ontology API generation framework with a one-to-one mapping like those mentioned in the introduction to generate a programming access to the ontology. Figure 2 shows the generation result for the ontology excerpt presented above using such an existing tool. The framework creates a class representation

² <http://www.exif.org/> last visit dec 05, 2011.
³ Basic Geo (WGS84 lat/long) Vocabulary <http://www.w3.org/2003/01/geo/> provides the namespace, last visit dec 05, June 2011.
⁴ <http://www.foaf-project.org/> last visit dec 05, 2011.

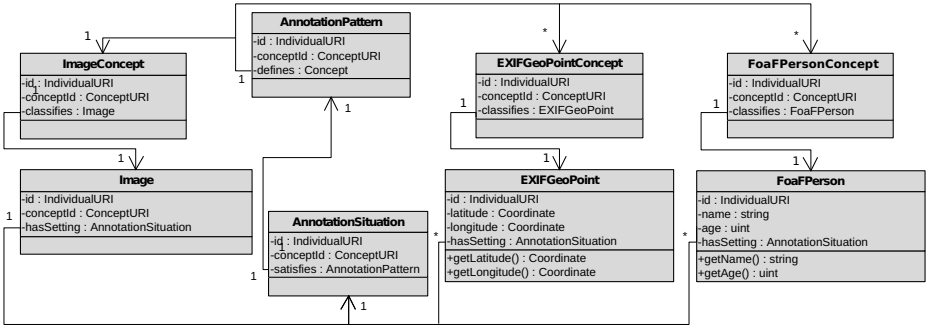


Fig. 2. Naive ontology API implementation generated by existing tools

for each of the concepts defined in the ontology. The relationships between concepts are represented as fields of the domain classes, e.g., the *satisfies* relationship between the `m3o:AnnotationSituation` and the `m3o:AnnotationPattern` concept is represented as `satisfies` field of type `AnnotationPattern` in the `AnnotationSituation` class. The generated class structure gives Jim no information about how to use it, i.e., which classes to instantiate when annotating an image with a geo-point or a creator. In fact one has to instantiate the class representations `AnnotationPattern`, `AnnotationSituation`, `Image`, `EXIFGeoPoint`, `ImageConcept` and `EXIFGeoPointConcept` and fill all the fields representing the relationships, namely `defines`, `classifies`, `hasSetting` and `satisfies`.

Furthermore not all class representations are of direct concern for Jim's application. Some of these representations provide direct *content* for the application, like the annotated entity — the `Image` — or the annotation entities — the `EXIFGeoPoint` and the `FoaFPerson`. Other classes only provide the structure necessary for a proper knowledge representation. The M3O ontology uses the Description & Situation (D&S) ontology design pattern. Description & Situation is another reification [3] formalism in contrast to the RDF reification [5]. For using D&S as reification formalism one has to add additional resources, the description, situation and the classifying concepts. The class representation for these concepts are of no use for Jim when using the API in his application. For this reason, he decides to encapsulate them from direct access and hide them from an eventual application developer.

2.4 Solution: Reference API for the Example Ontology

Due to the problems arising with the use of simple one-to-one mappings, Jim decides to build a programming interface to the ontology without the use of an API generation framework. Please note that the subsequently described API results from the design decisions made by Jim and represents only one possible model of an API for accessing this ontology. The API model designed by Jim is presented in Figure 3. In addition Figure 4 shows two further possible

⁵ <http://www.w3.org/TR/rdf-mt/#ReifAndCont> last visit dec 10, 2011.

models. All the API models are used in our evaluation in Section 5. Jim first identifies the functionality to be provided by the API, the annotation of images. Jim decides to provide a class for this annotation, the annotation class. In the following, we describe the different designs of the three APIs. **API-1:** He defines the set of concepts and properties involved in this functionality. Jim classifies the concepts in this set according to how they are used in the application and he splits them into two disjoint sets. The first set contains all concepts representing the *content* the application works on. In our terminology, we call them *content concepts*. We would like to emphasize that in our scenario Jim as an API developer will not have to know about the terminology we use at all; but it is significantly easier in this paper to use our terminology to explain the different decisions he may take when developing the API. For our example Jim chooses the `m3o:Image`, the `wgs84:Point` and the `foaf:Person` to provide the *content*. The other set contains the concepts of *structural* concern for the knowledge representation. Subsequently, we call these concepts *structure concepts*. For Jim these concepts are `m3o:AnnotationPattern`, `m3o:AnnotationSituation`, `m3o:AnnotatedConcept`, `m3o:GeoPointConcept`, and `m3o:CreatorConcept` and he wants his API to encapsulate and hide class representations of such concepts from the application. In our terminology, we call a set of concepts and relations related to an API class a *semantic unit* $SU = (CO, SO, R)$ with CO the set of *content concepts*, SO the *structure concepts* and R the set of relations. For our example, *semantic units* are, e.g., the annotation as described above or the geopoint consisting of the `wgs84:Point` together with its latitude and longitude. Jim wants his API to be prepared for arbitrary multimedia content and new types of annotations. The ontology provides abstract concepts for multimedia content and annotations in its inheritance structure presented in Figure 1b. But not all concepts from this structure are of interest to the application. Thus Jim decides to use only the least common subsumers, e.g., `dul:InformationObject` for annotatable multimedia content and `dul:Object` for annotations. Jim implements interfaces representing these two concepts.

Jim is now able to design the API. He defines a class for the annotation functionality as shown in Figure 3. In addition, he defines a class for each *content concept* the application works on, in this case **Image**, **EXIFGeoPoint** and **FoafPerson**. These classes implement the interfaces derived from the inheritance structure of the ontology, **InformationEntity** and **Object**. The **InformationEntity** interface has to be realized by classes representing multimedia content, e.g., by the **Image** class. The **Object** interface has to be realized by annotation entities, e.g., the classes **EXIFGeoPoint** and **FoafPerson**. All these classes and interfaces together with the operations form a so-called *pragmatic unit*. A *pragmatic unit* is a tuple $PU = (C, F, M)$ that contains the classes C , the fields F and the methods M of an object-oriented model and that relates to a specific *semantic unit* in the underlying knowledge model.

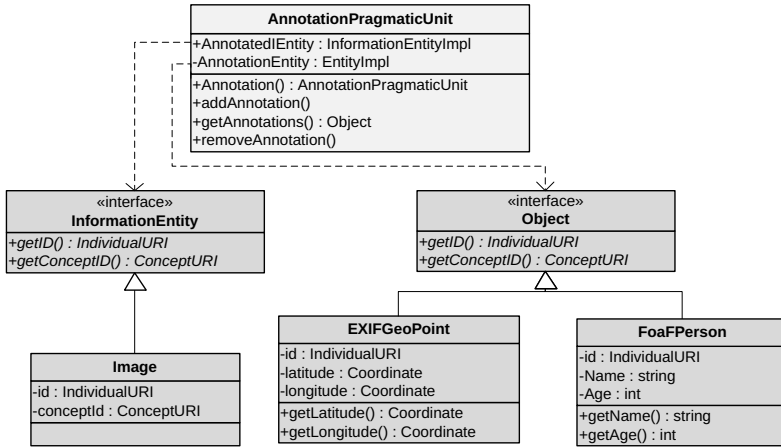


Fig. 3. API for the Running Example developed by Jim

API-2: Another possible model is API 2 shown in Figure 4, which is a more lightweight API for an image-viewer. The API only consists of three classes, a representation for the `m3o:Image`, the `wgs84:Point` and the `foaf:Person`. The class representation of the annotation *semantic unit* is integrated within the `m3o:Image content concept` class representation.

API-3: The decisions behind API 3, shown in Figure 4 are basically the same as for API 1 with the difference that the annotation *semantic unit* class representation should be identifiable by an URI. For this purpose the annotation *semantic unit* class representation is integrated with the `AnnotationSituation` class representation. In this API model the `AnnotationSituation` is classified as *content concept* and encapsulates the annotation *semantic unit*.

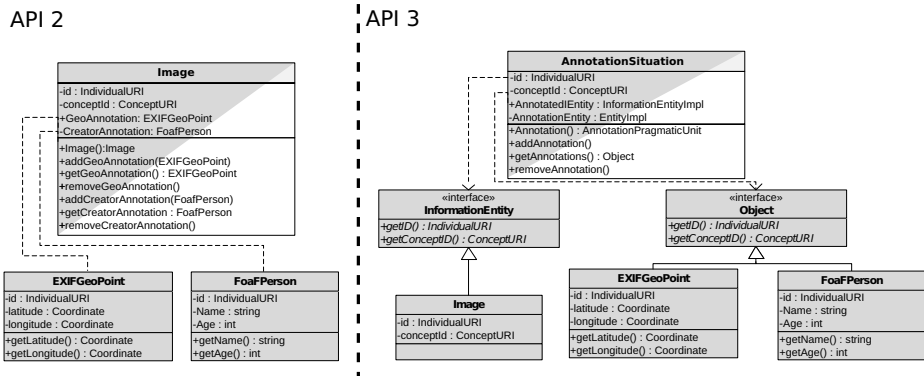


Fig. 4. Alternative APIs for the Running Example Ontology

3 Requirements for Programming Access to Ontologies

We analyze the requirements for the generation of programming access to ontologies. The requirements have been derived from real world implementation efforts made for different projects in our workgroup, e.g., the EU project WeKnowIt⁶. We use the scenario in Section 2 and the implementation of the reference API described in Section 2.4 to motivate the requirements. The requirements are distinguished into two sets of requirements: (1) requirements directly related to the programming access described in Section 3.1, typically in form of an API; (2) requirements related to a process that generates such an API described in Section 3.2.

3.1 Requirements on the Pragmatic Programming Access

(R1) Concept Representations. Programming access to ontologies has to represent the ontology concepts as classes in the object-oriented software system similar to Data Access Objects⁷ (DAOs), ActiveRecord or Data Mapper (both [1]) in the world of relational databases. Frameworks like those presented in the related work usually map each ontology concept to an object-oriented class representation and map the concept's properties to fields of this class. For our example, such a mapping is shown in Figure 2.

(R2) Encapsulation. Not all concepts of the ontology are of concern for an application developer. In Section 2.4, Jim identifies several concepts providing the *content* for his application, the *content concepts*. The rest of the concepts are classified as *structure concepts*. These *structure concepts* are only of concern for the proper knowledge representation. A programming access should provide for encapsulating concepts that are not interesting for an application developer.

(R3) Mapping of Inheritance Structures. There are differences between the inheritance structure of an API and of an ontology. In object-orientation, a class can inherit both data (attributes) and behavior (methods) from an ancestor class. Furthermore some object-oriented languages do not support multiple inheritance, e.g. Java. For generating programming access to ontologies, we need information how to generate a lean and useful inheritance structure from the ontology for the API.

(R4) Pragmatic Units. APIs provide a programming interface for their responsibility, e.g., the annotation of images like the API from our example in Section 2.3. Such a programming interface supports methods to perform operations, like in our example adding, removing or manipulating annotations and images. Performing such operations in programming access to ontologies often results in the manipulation of multiple ontology entities and thus multiple concept-class

⁶ <http://www.weknowit.eu/> last visit dec 5, 2011.

⁷ DAOs as Core J2EE Pattern <http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html>

representations. Our API should provide classes to support the application developer in performing these operations in an easy and well encapsulated way.

(R5) Method Behavior. APIs provide methods to access or manipulate API entities or to query for entity properties. In some cases, it might be necessary to fall back to reasoning on the ontology [10] to be able to answer queries. For example querying for all instances of a specific concept could be such a question. A method for such a query performed on the Java representation could guarantee soundness but never completeness. The same also applies for consistency preservation. In some cases, the API could restrict its behavior in a way that it ensures the consistency of the represented knowledge. We expect the API to either inform the calling method or throw an exception that the requested action would affect the consistency of the represented knowledge. Sometimes, it is not possible or practical for complexity reasons to restrict the API behavior. In this case the API cannot ensure the consistency. Currently, we focus on cases where restrictions or query answering on the API are possible, e.g., qualified number restrictions on properties. A reasoner integration to ensure validity of operations remains for future work.

3.2 Requirements on the Process for Generating Programming Access to Ontologies

(R6) Customizing generated APIs. The output of the API generation process is strongly driven by the developer and the context of the target application. For instance, in Section 2.4 we have demonstrated how three different APIs might have been defined for a given ontology, reflecting different needs of the target applications. The generation process has to support the developer in controlling and customizing the output. From our observation, we know that concept classification and assignment to *semantic units* is mostly uniform for various application scenarios but choice of *pragmatic units* and their arrangement can vary strongly from case to case. The import of ontologies and the intended inheritance structure in the API can also vary for different application scenarios.

(R7) Legacy APIs integration. The API developer should be able to integrate legacy APIs. Let us assume Jim uses the image class of the AWT API⁸. To use this image class, Jim has to integrate it with the ontology API and provide ontology access functionalities for this class.

(R8) Import. The generation process has to deal with import instructions in the ontologies. A generation process has to manage all imports and decide which are important for the API generation process.

(R9) Deanonimization of Concepts. Ontologies allow for anonymous concepts in complex class expressions in OWL or blank nodes in RDFS. However, there are no anonymous classes in object orientation. For this reason, we only allow named concepts in ontologies and need to de-anonymize anonymous concepts first, if necessary.

⁸ <http://download.oracle.com/javase/1.4.2/docs/api/java/awt/Image.html>

4 Programming Access to Ontologies with OntoMDE

In order to alleviate application developers from building the pragmatics of accessing Semantic Web knowledge in object-oriented applications, we present OntoMDE a Model-driven Engineering (MDE) approach for the generation of programming access APIs from an input ontology. The OntoMDE framework guides the developer through the semi-automatic generation process. Figure 5 depicts the whole process with its two intermediate models, the MoOn and the OAM. OntoMDE provides tools to support the user in adding declarative information about the pragmatic programming access to the intermediate models.

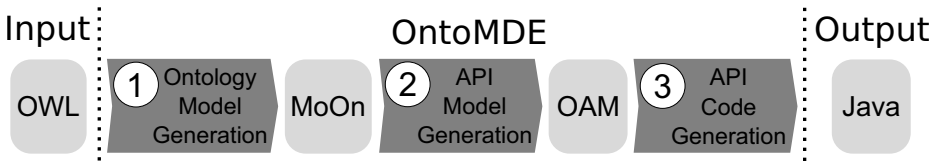


Fig. 5. The API Generation Process

In the first step, the Model of Ontologies (MoOn) is used to represent crucial properties of the target API as properties of the ontology in a declarative manner. In MoOn, concepts are classified as either being *content concepts* or *structure concepts*. *Semantic units* are defined and one can adapt parts of the ontology’s inheritance structure to the API. Figure 6 shows the *semantic unit annotation* from our running example in the MoOn-based representation.

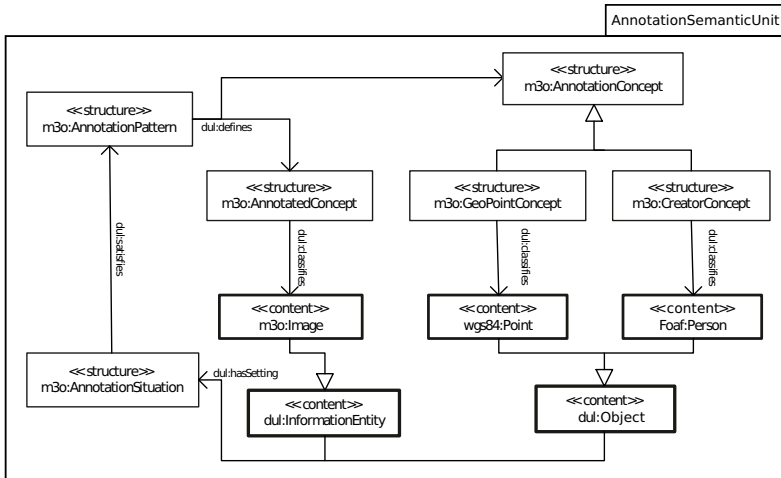


Fig. 6. The Annotation *Semantic Unit* in the MoOn

In the second step, the MoOn-based representation is transformed to the Ontology API Model (OAM). The OAM provides a declarative representation of API properties that cannot be tied to the structure of the ontology, like legacy API integration or method behavior customization. In addition, the OAM enables to embed information relevant for the code generation process, e.g., to tailor the concrete API to a particular repository backend. Figure 3 shows the OAM for our running example. Finally, the code is generated from the OAM in fully automated manner.

In the following sections, we describe the different transformation steps along the example from Section 2 in more detail and associate the design decision with the requirements from the previous section.

4.1 Step 1: From Ontology T-Box to MoOn

MoOn is based on an adaptation of the ECore Metamodel for OWL2⁹. The transformation of OWL-based ontology entities into a MoOn representation is inspired by the OWL-to-UML mappings described in [6, 4, 11], see the discussion on mapping models in the related work in Section 6. A MoOn model for an ontology results from two different steps: First, a fully automatic transformation of the ontology in an ECore model. Second, a manual extension of this ECore model with declarative information about the pragmatic programming access.

Transformation from OWL to MoOn: First, we have to represent the ontology in the MoOn. This preparation of the MoOn includes the representation of all relevant concepts, see (R1). For this reason, ontologies distributed over multiple files are accumulated, imports in the ontology are resolved, see (R8) and implicit knowledge of the ontology is materialized using reasoning. After these steps, we substitute anonymous concepts by named concepts (R9). This is easily possible in Description Logics based languages as OWL by just naming all anonymous classes. In a last step, we consider the parts of the inheritance structure that are carried over to the MoOn-based ontology representation (compare Figure 1 and Figure 6). To adapt the inheritance structure in MoOn-based ontology representations to our needs, the proper concepts from the inheritance structure are selected, e.g., by choosing the least common super-concept (R3).

Adding Declarative Information to the MoOn: The next step is to add responsibility-driven information, i.e., information about how to use ontology concepts in context of the applications. The user defines *semantic units* and allocate concepts to them, see (R4). For our example, Jim represents the annotation functionality as *semantic unit* and allocates all concepts shown in Figure 6 to it. Additionally, the concepts have to be classified into *structure concept* and *content concepts* (R2). For Jim, m3o:Image, wgs84:Point and foaf:Person are the *content concepts* and m3o:AnnotationPattern, m3o:AnnotationDescription, m3o:AnnotatedConcept and m3o:GeoPointConcept are the *structure concepts*.

⁹ MOF-Based Metamodel for OWL2

http://www.w3.org/2007/OWL/wiki/MOF-Based_Metamodel

OntoMDE provides for user support in concept assignment and classification tasks. Based on an existing *semantic unit* allocation, OntoMDE suggests for concept classification and based on concept classifications OntoMDE can give advices for *semantic unit* allocation.

4.2 Step 2: From MoOn to OAM

The OAM uses the syntax of UML2 with profiles¹⁰ in order to represent the target API. The primary purpose of the OAM is to provide declarative representations of additional information used during code generation. For example, information to integrate a particular repository backend (R6) or information about the integration of legacy API classes (R7). Very important is information about the characteristics of properties such as symmetry or transitivity. This is used to support dedicated method behavior (R5) in the ontology API. The API representation in the OAM is generated fully automatically from the MoOn-based ontology representation. In this transformation, class representations for *content concepts*, *semantic units* and interfaces for the inheritance structure are generated, similar to what Jim did in Section 2.4 (R1,R3,R4). Table 1 summarizes the mappings between MoOn entities and the API entities.

Table 1. Overview of Mappings between MoOn and OAM

MoOn based ontology representation	Ontology API Model (OAM)
Content concepts	Content classes & class fields
Content individuals	Content objects
Structure concepts	Class attributes
Structure individual	Individual URI and concept URI
Semantic unit	Pragmatic unit class
Concept properties & relations	Encapsulated in Pragmatic unit classes or class fields
Property characteristics	declarative extension in OAM

Step 3: Generating the Code of an API from the OAM

In the last step, we generate code from the API representation in the OAM. This fully automated process is supported by the OntoMDE toolkit using Java Emitter Templates¹¹ (JET) as code generation framework.

5 Case Studies and Lessons Learned

The primary objective of our case studies is to demonstrate the applicability of our approach. In addition, we want to show the flexibility and adaptability of the approach.

¹⁰ http://www.omg.org/technology/documents/profile_catalog.htm

¹¹ <http://www.eclipse.org/modeling/m2t/?project=jet#jet> last visit dec 5, 2011.

To show the applicability of our approach, we have developed and applied the OntoMDE toolkit to generate APIs from different ontologies. We have selected ontologies with different characteristics in terms of complexity, level of abstraction, degree of formalization, provenance, and domain-specificity. We have used the OntoMDE toolkit to generate APIs for the Pizza^[12] and Wine^[13] ontologies. As less formal real world ontologies, we have chosen the Ontology for Media Resources (OfMR)^[14] of the W3C and the CURIO^[15] ontology used in the We-KnowIt project^[16]. And last, we have used OntoMDE to generate APIs for the M3O^[13], our running example is based on, and the Event-Model-F (EMF)^[14].

To demonstrate the flexibility and adaptability, we used OntoMDE to generate different APIs from the same input ontology, from slightly changed versions of the same ontology and to integrate legacy APIs into our ontology access API. We have selected the M3O ontology, OfMR aligned with the M3O and an EXIF^[17] ontology aligned to the M3O as input ontology for this study. As outlined for our example in Section 2.4, we designed different possible APIs for accessing the M3O. Then, we generated these APIs from the M3O ontology by changing the declarative information about programming access on the MoOn and the OAM. To show the integration capabilities of OntoMDE, we use the OAM to integrate legacy APIs for the `Image` class in the M3O API.

With the first use case, the generation of APIs for the Pizza and Wine ontologies, we have shown that our approach is capable of processing OWL ontologies, (R1,R9). From applying OntoMDE to multiple ontologies with different characteristics, we can conclude that the general idea of distinguishing concepts into *content concepts* or *structure concepts* is applicable to all tested ontologies. The concrete sets of *content concepts* or *structure concepts* strongly depends on the characteristics of the ontology. In simple, less formal ontologies most of the concepts are *content concepts* of direct concern for the application. Whereas, in complex ontologies with a high level of abstraction and intense use of reification more of the concepts tend to be *structure concepts*. The organization of concepts in *semantic units* is also applicable to all kinds of ontologies. Again, we encounter differences depending on the characteristics of the ontology. Simple ontologies often only allow for few and usually small *semantic units*. Complex ontologies allow for multiple partially overlapping *semantic units* with potentially many concepts.

We have also investigated the flexibility and adaptability of our approach. Regarding the adaptability, we have integrated the `java.awt.image` package as legacy APIs for representing images into the APIs of our example. Using the OAM, the integration of the generated API and the legacy API could be conducted in a

¹² The pizza ontology <http://www.co-ode.org/ontologies/pizza/2007/02/12/> last visit dec 5, 2011.

¹³ <http://www.w3.org/TR/owl-guide/wine.rdf> last visit dec 5, 2011.

¹⁴ <http://www.w3.org/TR/mediaont-10/> last visit dec 5, 2011.

¹⁵ http://www.weknowit.eu/content/curio_collaborative_user_resource_interaction_ontology last visit dec 5, 2011.

¹⁶ <http://www.weknowit.eu/> last visit dec 5, 2011.

¹⁷ <http://www.exif.org/specifications.html> last visit dec 5, 2011.

few steps. As mentioned, we have generated different APIs for the ontology from our example. We have also shown that changes of the API model could be accomplished by modifications on the MoOn, such as "choice of pragmatic units" or "choice of content concepts". As you can see, these changes result in different numbers of pragmatic units and generated concept classes. To demonstrate the flexibility regarding the actual RDF-persistence layer used, we have changed the back-end API of the OntoMDE approach. We used our own RDF-persistence layer Winter [12] as well as the RDF-persistence layer Alibaba [18]. This change of the backend could be conducted within a short time of about one hour. This addresses requirements (R5), (R6), and (R7).

6 Related Work

The problem space of object relational impedance mismatch and the set of conceptual and technical difficulties is addressed frequently in literature, e.g. in [5,15,16,2]. Among others, Fowler provides in his book [1] a wide collection of patterns to common object relational mapping problems. Due to the fact that many problems in persistence and code generation for ontologies are similar to problems from the field of relational databases many approaches utilize object-relational strategies for object-triple problems, for example like ActiveRDF [8], a persistence API for RDF adapting the object-relational ActiveRecord pattern from Fowlers book or OTM [19] a framework that resembles some of Fowlers patterns to the field of object-triple mapping. Most of the other frameworks, like Alibaba, OWL2Java [6], Jastor [20], OntologyBeanGenerator [21], Àgogo [9], and others, use similar techniques adapting object-relational solutions. An overview can be found at Tripresso [22], a project web site on mapping RDF to the object-oriented world. These frameworks use a simple mapping model for transforming each concept of the ontology into a class representation in a specific programming language like Java or Ruby. Properties are mapped to fields. Only Àgogo [9] is a programming-language independent model driven approach for automatically generating ontology APIs. It introduces an intermediate step based on a Domain Specific Language (DSL). This DSL captures domain concepts necessary to map ontologies to object-oriented representations but it does not captures the pragmatics.

The mappings used to generate the MoOn from the OWL ontologies are based on the work done for the Ontology Definition Metamodel (ODM) [4,11]. The Ontology Definition Metamodel [7] is an initiative of the OMG [23] for defining an ontology development platform on top of MDA technologies like UML.

¹⁸ <http://www.openrdf.org/doc/alibaba/2.0-alpha4/> last visit dec 5, 2011.

¹⁹ <https://projects.quasthoffs.de/otm-j> last visit dec 5, 2011.

²⁰ <http://jastor.sourceforge.net/> last visit dec 5, 2011.

²¹ <http://protege.cim3.net/cgi-bin/wiki.pl?OntologyBeanGenerator> last visit dec 5, 2011.

²² <http://semanticweb.org/wiki/Tripresso> last visit dec 5, 2011.

²³ <http://www.omg.org/> last visit dec 5, 2011.

7 Conclusion

We have presented with MoOn and OAM a declarative representation of properties of ontologies and their entities with regard to their use in applications and application programming interfaces (APIs). On this basis, we have introduced a multi-step model-driven approach to generate APIs from OWL-based ontologies. The approach allows for user-driven customizations to reflect the needs in a specific application context. This distinguishes our approach from other approaches performing a naive one-to-one mapping of the ontology concepts and properties to the API classes and fields, respectively. With our approach, we alleviate the developers from the tedious and time-consuming API development task such that they can concentrate on developing the application's functionalities. The declarative nature of our approach eases reuseability and maintainability of the generated API. In the case of a change of the ontology or the API, most of the time only the declarative representation has to be adapted and a new API could be generated. In our case studies, we applied our approach to several ontologies covering different characteristics in terms of complexity, level of abstraction, degree of formalization, provenance, and domain-specificity. For our future work, we plan to integrate the support for different method behaviors (see R5) and the dynamic extensibility of ontologies. The support of the dynamic extensibility of ontologies strongly depends on the persistence layer used. Another idea is to use the declarative representation in combination with the ontology to prove consistency of the data representation and manipulation in the API regarding the ontology.

Acknowledgements. This research has been co-funded by the EU in FP7 in the SocialSensor project (287975).

References

1. Fowler, M.: Patterns of Enterprise Application Architecture. Addison-Wesley Longman, Amsterdam (2002)
2. Fussell, M.L. (ed.): Foundations of Object Relational Mapping (2007), http://www.database-books.us/databasesystems_0003.php
3. Gangemi, A., Mika, P.: Understanding the Semantic Web through Descriptions and Situations. In: Meersman, R., Schmidt, D.C. (eds.) CoopIS 2003, DOA 2003, and ODBASE 2003. LNCS, vol. 2888, pp. 689–706. Springer, Heidelberg (2003)
4. Hart, L., Emery, P.: OWL Full and UML 2.0 Compared (2004), <http://uk.builder.com/whitepapers/Oand39026692and60093347p-39001028qand00.html>
5. Ireland, C., Bowers, D., Newton, M., Waugh, K.: A classification of object-relational impedance mismatch. In: Chen, Q., Cuzzocrea, A., Hara, T., Hunt, E., Popescu, M. (eds.) DBKDA, pp. 36–43. IEEE Computer Society (2009)
6. Kalyanpur, A., Pastor, D.J., Battle, S., Padget, J.A.: Automatic Mapping of OWL Ontologies into Java. In: SEKE (2004)
7. Ontology Definition Metamodel. Object Modeling Group (May 2009), <http://www.omg.org/spec/ODM/1.0/PDF>

8. Oren, E., Delbru, R., Gerke, S., Haller, A., Decker, S.: Activerdf: object-oriented semantic web programming. In: WWW. ACM (2007)
9. Parreiras, F.S., Saathoff, C., Walter, T., Franz, T., Staab, S.: à gogo: Automatic Generation of Ontology APIs. In: IEEE Int. Conference on Semantic Computing. IEEE Press (2009)
10. Parreiras, F.S., Staab, S., Winter, A.: Improving design patterns by description logics: A use case with abstract factory and strategy. In: Khne, T., Reisig, W., Steimann, F. (eds.) Modellierung. LNI, vol. 127, pp. 89–104. GI (2008)
11. Rahmani, T., Oberle, D., Dahms, M.: An Adjustable Transformation from OWL to Ecore. In: Petriu, D.C., Rouquette, N., Haugen, Ø. (eds.) MODELS 2010, Part II. LNCS, vol. 6395, pp. 243–257. Springer, Heidelberg (2010)
12. Saathoff, C., Scheglmann, S., Schenk, S.: Winter: Mapping RDF to POJOs revisited. In: Poster and Demo Session, ESWC, Heraklion, Greece (2009)
13. Saathoff, C., Scherp, A.: Unlocking the Semantics of Multimedia Presentations in the Web with the Multimedia Metadata Ontology. In: WWW. ACM (2010)
14. Scherp, A., Franz, T., Saathoff, C., Staab, S.: F—a model of events based on the foundational ontology DOLCE+DnS Ultralight. In: K-CAP 2009. ACM, New York (2009)
15. Ambler Scott, W.: Crossing the object-data divide (March 2000), <http://drdobbs.com/architecture-and-design/184414587>
16. Ambler Scott, W.: The object-relational impedance mismatch (January 2010), <http://www.agiledata.org/essays/impedanceMismatch.html>
17. Wirfs-Brock, R., Wilkerson, B.: Object-Oriented Design: A Responsibility Driven Approach. SIGPLAN Notices (October 1989)

Clinical Trial and Disease Search with Ad Hoc Interactive Ontology Alignments

Daniel Sonntag¹, Jochen Setz¹, Maha Ahmed-Baker¹, and Sonja Zillner²

¹ German Research Center for AI (DFKI)
Stuhlsatzenhausweg 3, 66123 Saarbruecken, Germany

² Siemens AG, Corporate Technology
Otto-Hahn-Ring 6, 81739 Munich, Germany

Abstract. We will explain how an LODD application based on diseases, drugs, and clinical trials can be used to improve the (ontology-based) clinical reporting process while, at the same time, it improves the patient follow-up treatment process. Specific requirements of the radiology domain let us aggregate RDF results from several LODD sources such as DrugBank, Disesome, DailyMed, and LinkedCT. The idea is to use state-of-the-art string matching algorithms which allow for a ranked list of candidates and confidences of the approximation of the distance between two diseases at query time. Context information must be provided by the clinician who decides on the “related”-mappings of patient context and links he wants to follow in order to retrieve disease and medication information.

1 Introduction

In many industrial domains such as medical radiology, a vast amount of images is produced and the medical image annotations must be refined and augmented during a complex medical workflow. We use several technologies for the semantic annotation of medical images and radiology reports [10]. The problem is that these annotations only capture descriptive information, i.e., the clinical observations, the various identified symptoms, and the discovered findings. But in practice, clinicians often want to search for higher level information such as interventions and the respective side effects, or associated information such as related drugs and diseases in the context of identified symptoms, etc. As a matter of fact, expert knowledge about diseases, drugs, and clinical trails is often not available through internet searches or too imprecise.

Our experiences throughout the THESEUS MEDICO¹ and RadSpeech² research projects (which focus on semantic medical image search and user interaction [12], respectively) have shown us that several types of knowledge contained in Linked Data are relevant for the annotation of the images. In other words, when radiologists (or other clinicians) examine their patients’ medical

¹ <http://theseus-programm.de/en/920.php>

² <http://www.dfki.de/RadSpeech/>

images they would additionally like to know whether previous diagnoses exist, if there has been a change in the case, and what kind of medication and treatment plan is foreseen. This requires the medical images to be annotated accordingly so that the radiologists can obtain all the necessary information starting with a computer tomography (CT) or magnet resonance (MR) image, and the case description in form of a patient record. We will explain how an LODD (<http://www.w3.org/wiki/HCLSIG/LODD>) application based on diseases, drugs, and clinical trials can be used to improve the (ontology-based) clinical reporting process while, at the same time improving the patient follow-up treatment process (i.e., monitoring the patient's health condition and the development of the disease). We will focus on the essential part of ontology matching between the medical reference ontology, Radlex³ [5], and the available and relevant LODD data contained in DrugBank, DailyMed, and Diseasesome which are mediated through the LinkedCT resources. LinkedCT, see <http://linkedct.org>, aims at publishing the first open Semantic Web data source for clinical trials data; it contains more than 60,000 trails, 14,243 conditions, and 67,271 interventions.

Essentially, the important mapping task between LinkedCT and Diseasesome must be seen in the context of a more complex medical workflow which we will explain in detail. In addition, the mapping of several additional resources can only be done interactively, at query time, to meet both the data and the intentions of the clinician who searches for trail and drug information. The reason for this is that a radiologist can only decide in an ad hoc fashion whether two proposed “equality” or “related” matches are appropriate in a specific patient and knowledge retrieval context. This paper is structured as follows. Section 2 describes the clinical problem statement and argues in favour of a context-based interactive approach; section 3 describes the workflow we created in order to meet the clinical requirements while embedding the context-based interactive approach into a concrete knowledge retrieval scenario. Section 5 provides a first evaluation of the approach to meet the clinical requirements; section 6 concludes.

2 Clinical Problem Statement

In our carrier project THESEUS MEDICO, we envision a flexible and generic image understanding software. Semantics of the images plays the major role for access and retrieval. The next generation of intelligent, scalable, and robust search engines for the medical imaging domain should be based on semantic technologies. With the incorporation of higher level knowledge represented in ontologies, different semantic views of the same medical images (such as structural aspects, functional aspects, and disease aspects) can be explicitly stated and integrated.

³ Radlex is a controlled vocabulary developed and maintained by the Radiological Society of North America (RSNA) for the purpose of indexing and retrieving radiology images and related information. Radlex contains 11,962 domain related terms (e.g., *anatomy pathology* or *imaging techniques*). Synonym information is partially indicated, such as *Schatzki ring* and *Lower esophageal mucosal ring*.

A first analysis of the envisioned search functionality revealed that the association between observed diseases (e.g., lymphoma) and different, related types of the same disease serve as a valuable knowledge resource (as it can be used for refining the search query) when searching for similar patients and/or clinical trials. We identified related LODD resources to capture valuable associations between the various high-level concepts such as diseases, interventions, medications, symptoms, etc. Those concepts occur within the clinical diagnostic process and are thus very relevant for defining search queries. The two related LODD resources have been identified, Drugbank⁴, Diseasome⁵, and DailyMed⁶. Figure 1 shows the identified LODD resources and a potential interlinking.

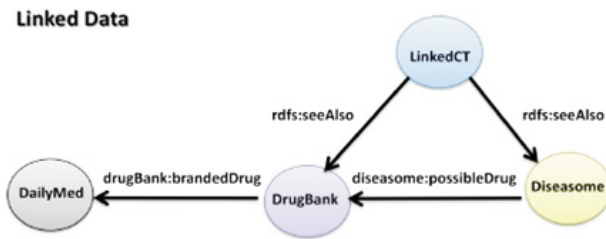


Fig. 1. Identified LODD resources

Existing medical ontologies for anatomy and disease related aspects (cf. FMA, RadLex or NCI-Thesaurus) usually focus on one particular domain, such as anatomy or radiology, and do not cover relations that link concepts from other domains such as those which link associated findings with diseases. Most medical ontologies of this scale for anatomy, disease, or drug aspects can be summarised as: (a) they are very large models, (b) they have extensive is-a hierarchies up to ten thousands of classes which are organised according to different views, (c) they have complex relationships in which classes are connected by a number of different relations, (d) their terminologies are rather stable (especially for anatomy) in that they should not differ too much in the different ontologies

⁴ DrugBank is a large repository of small molecule and biotech drugs that contains detailed information about drugs including chemical, pharmacological, and pharmaceutical data in addition to sequence, structure, and pathway information. The Linked Data DrugBank contains 1,153,000 triples and 60,300 links.

⁵ Diseasome contains information about 4,300 disorders and disease genes linked by known disorder's gene associations. It also indicates the common genetic origin of many diseases. The list of disorders, disease genes, and associations between them comes from the Online Mendelian Inheritance in Man (OMIM), which is a compilation of human disease genes and phenotypes. The Linked Data Diseasome contains 88,000 triples and 23,000 links.

⁶ Dailymed provides up-to-date information about marketed drugs. Human Prescription Labels, OTC Labels, and Homeopathic Labels sum up to several million entries.

(we will show the opposite for the cancer disease parts), and (e) their modelling principles are well defined and documented.

A variety of methods for ontology alignment have been proposed [13,24,18,6]. The objective of the state-of-the-art in ontology mapping research includes the development of scalable methods (e.g., by combining very efficient string-based methods with more complex structural methods), and tools for supporting users to tackle the interoperability problem between distributed knowledge sources (e.g., editors for iterative, semi-automatic mapping with advanced incremental visualisations [9]). In addition, cognitive support frameworks for ontology mapping really involve users [3], or try to model a natural language dialogue for interactive semantic mediation [11].

One of the ontology matching problems in medicine is still that, in many cases, complex ontology matching algorithms cannot be used because they do not scale to sizes of medical ontologies—complex methods for ontology alignment in the medical domain turned out to be unfeasible because the concept and relation matrix is often on the scale of 100000×100000 alignment cells and appropriate subontologies cannot be created with state-of-the-art methods because of complex inter-dependencies.

Another problem is that when using those methods, we can only work with static mapping as a result of an offline matching process in which the mappings are independent of the context in which they are used. In the context of our medical use case, however, we learned in discussions with our clinicians that establishing clinical relevant associations between given clinical concepts has the potential to improve the search functionality. But that comes with the condition of a context-dependent quality and relevance of established associations (i.e., alignments) between clinical concepts which determines to which extent the search functionality can be improved.

We argued in [14] that annotating medical images with information available from LODD can eventually improve their search and navigation through additional semantic links. One outcome of our ontology engineering methodology [15,13] was the semi-automatic alignment between radiology-related OWL ontologies (FMA [7] and Radlex). This alignment could be used to provide new connections in the medicine-related linked data cloud. The fact that context-dependency may play a pivotal role for static alignments from FMA to Radlex was shown in [16]. Why should this problem be more severe in the context of an online information retrieval task where diseases from LinkedCT and Diseasesome have to be aligned?

Basically speaking, small changes in the nomenclature can make a big difference in the adequacy of the proposed mappings. This is driven by the fact that, in medicine, usually a very specific difference in the concept names might make a huge difference for their interpretation (therefore we should not even try to map LinkedCT and Diseasesome unless there are exact matches.) But the absence of globally unique identifiers for diseases (the URIs) forces us to provide the mappings. Instead of trying to infer such mappings on the large scale for disease data sources (LinkedCT, for example, has only 830 *owl:sameAs* links to

Diseasome but contains more than 4600 different disease URIs), we try to establish an ad hoc mapping. Further, we ask whether static *nton* mappings between two medical ontologies are really necessary and represent really what is desired in a specific query situation. In the following, we will argue that, at least in our medical usecase, a rather different set of requirements exists.

For example, a clinical expert identifies patient cancer cells in the imaging data and is sure that they are from type “lymphoma”. Hence our (Radlex) search term is “lymphoma”. In order to decide on the follow-up treatment steps of the patient, he wants to search for similar patients/trials where similar patients have been successfully treated (case-based reasoning). Lymphoma diseases of patients can, however, be distinguished along three orthogonal dimensions:

1. the type (e.g., Hodgkin and non-Hodgkins lymphoma);
2. the stage (e.g., stage I to stage IV); and
3. the grade (e.g., low, intermediate or high grade).

As lymphomas of different type, stage, and grade grow at different rates, they respond differently to specific treatments. For that reason, clinicians need to know the particular type, stage, and grade of a patient’s lymphoma for an adequate treatment. Accordingly, we have to filter out the relevant trails, or in other words, align the trials according to this complex information background.

For that reason, we cannot rely on semantic/structured-based knowledge models to establish associations for “fine-tuning” our search space. To the best of our knowledge, no formal knowledge structure exists that relies these three dimension (type, stage, and grade) in a proper manner/model.

For the same reason, we also did not use a semantic similarity measure. Initially, it might appear appropriate to use abbreviation lists, synonym sets, etc. Although it seems to be obvious that purely syntactic (approximate) string matching techniques are not sufficient to deal with different data representations, different linguistic surface forms, and missing information about type, stage, and grade, it is wrong to believe that the potential increase in recall when using such query expansion only means a little loss of precision.

However, our interactive workflow is designed to increase precision at a stable recall level as our examples from LinkedCT and Diseasome will show. (The recall level can, however, be controlled by declaring the thresholds for individual ad hoc matchers.)

3 Proposed Workflow

Figure 2 shows the proposed workflow.

1. The user inputs a medical term or refers to the existing CT or MR image annotation. In both cases, a Radlex term is derived and used as the “disease_name”.

2. MEDICO then executes a SPARQL query that includes an (approximate) string matching filter to LinkedCT. (We made positive experience with the SPARQL operator implemented in Virtuoso (*filter(regex(?LCT_diseases_name, “*disease_name*”, ”i”)*). (This accounts for the online processing demand as well as the user’s expectation of the LinkedCT’s “first results”.)
3. The user browses the retrieved trails with a faceted browsing functionality we implemented (see section 5) and selects the trails he is interested in.
4. Then, he selects a disease from a specific trail.
5. The system now either follows the link to Diseaseome (trivial case) or invokes the interactive, approximate string-based matching procedure when there is no link available.
6. According to the found disease-URIs and the de-referenced possible drugs from DrugBank, the user can click further to obtain the highly desired DailyMed contents.

This process is utterly transparent to the user for two reasons. First, during the direct-manipulation faceted browsing and search interface, he knows exactly at which stage he is and exactly when additional input from him is required (stage 5). Second, built on the mental model of the retrieval stages and the context knowledge he has gained by inspecting the patient file, reflecting on the Radlex term, and following only the disease links he is interested in, the clinician knows how to interpret the proposed mappings. It becomes clear that the clinician is not necessarily interested in pure equality (or subsumption mapping), but uses a more underspecified “related-to” mapping/alignment. Interestingly, this “related-to” mapping only makes sense in this dynamic retrieval process where he or she uses the proposed mappings as a further anchor point in his extended search context. In other words, the user defines what he means by the mappings in an ad hoc fashion. Therefore, we cannot use static mappings at all or record our mappings as static ones. The question is whether this procedure really enhances the robustness of the overall search interface in specific, usecase-relevant retrieval situations (cf. section 5).

4 String-Based Interactive Matching Approach

The idea is to use state-of-the-art string matching algorithms which allow for a ranked list of candidates and confidences of the approximation of the distances between two strings. In addition, the individual results must be homogeneous in the sense that they are comparable and can be combined. The combination of the results of the string matching approaches is essential because the variety of similarity functions that exist in literature have specific positive but also negative characteristics that have to be taken into account and used in the right way. First, we define the string matching and the under context filter functions.

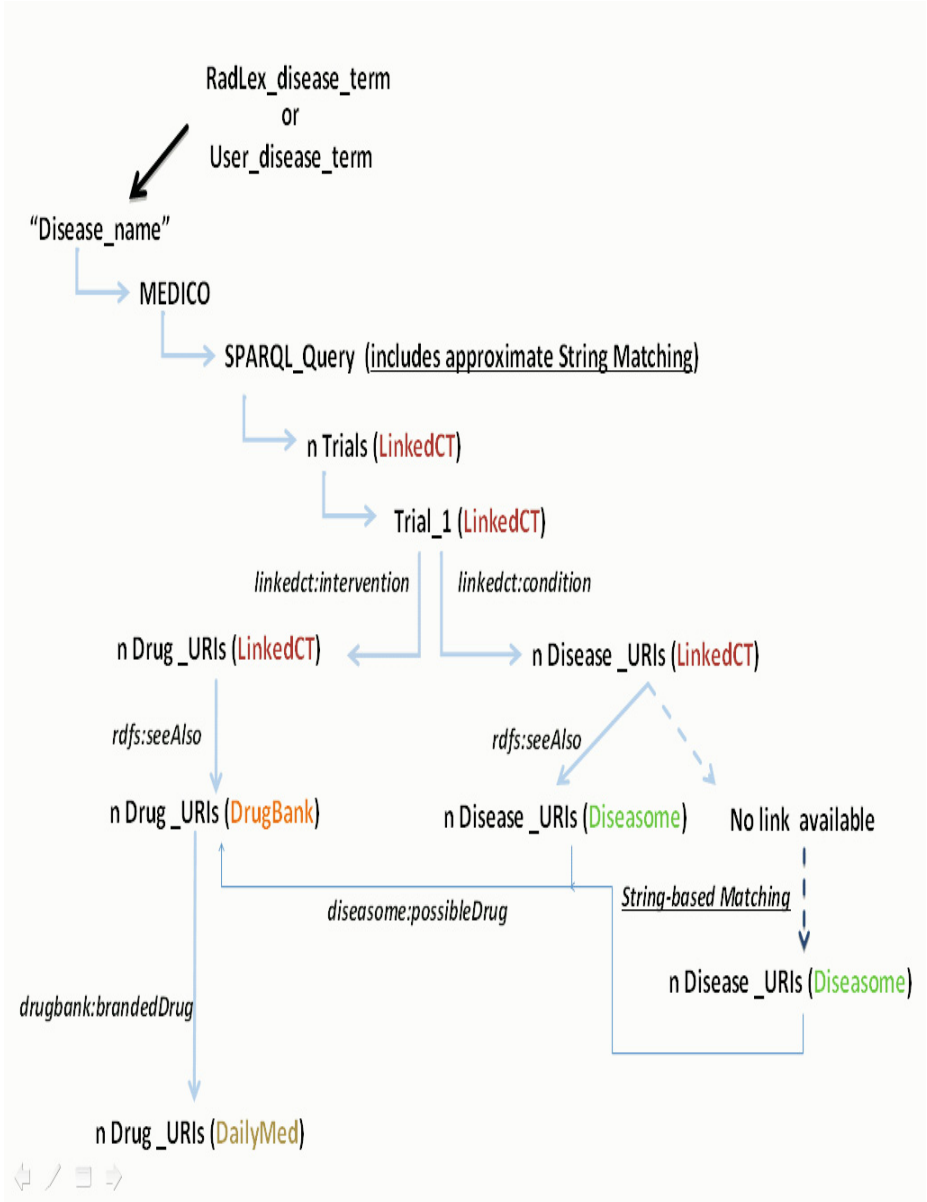


Fig. 2. Workflow of the Clinical Information Retrieval and Ontology Matching Tasks

Definition 1 (String Matching Function). Given a input search term t and a LODD knowledge repository L , be π a string matching filter function $\pi : t \times L \rightarrow \{(t, l) \mid l \in L \wedge \sigma(t, l)\}$, with σ a boolean similarity function s that is true whenever the terms t and l yield a string similarity according to the established similarity measure.

Definition 2 (User Context Filter Function). Given a input search term t , S a set of terms, then we define a user context filter function $\pi_{user} : t \times S \rightarrow \{(t, s) \mid s \in S \wedge \sigma_{relevant}(t, s)\}$, with $\sigma_{relevant}$ being a boolean function that is true whenever the similarity of the terms t and s is relevant in the context of the user.

Function π let’s us define a threshold on the individual contributors to an aggregate measure of string matches we take into account for the interaction step; π_{user} allows us to formalise the selection step the clinician performs according to his context information. If $\pi_{user} = true$, we established an ad hoc “related-to” mapping.

We used the similarity measures $sim_{WJaccard}$, sim_{WJaro} , $sim_{WJaro-Winkler}$, and sim_{ed} . and aggregated their results to a “star recommendation” scheme $starsTotal$, whereby $starsTotal = \sum_{i=1}^4 \pi : (\sigma = sim_i)$ The individual measures are defined as follows.

Formally, the Weighted Jaccard measure takes two disease names d_1, d_2 and computes (only) the fraction of tokens that are present in both multi-word terms.

$$sim_{WJaccard}(d_1, d_2) = \frac{\sum_{t \in \mathbf{d}_1 \cap \mathbf{d}_2} w(t, R)}{\sum_{t \in \mathbf{d}_1 \cup \mathbf{d}_2} w(t, R)}$$

$$sim_{WJaro}(d_1, d_2) = \frac{1}{3} * \left(\frac{|d'_1|}{|d_1|} + \frac{|d'_2|}{|d_2|} + \frac{|T_{d_1, d_2}|}{|d'_1|} \right)$$

The Jaro-Winkler variant of the Jaro measure takes the longest common prefix of d_1 and d_2 as length P to compute an edit distance. Letting $P' = max(P, 4)$, the Jaro-Winkler distance is defined by

$$sim_{WJaro-Winkler}(d_1, d_2) = sim_{WJaro}(d_1, d_2) + \frac{P'}{10} * (1 - sim_{WJaro}(d_1, d_2)).$$

The Damerau-Levenshtein Algorithm is implemented as follows: we use the function $sim_{ed} : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ taking $i = |d_1|$ and $j = |d_2|$ where $i, j \mapsto x$ is defined by:

$$sim_{ed}(0, 0) = 0$$

$$sim_{ed}(i, 0) = i$$

$$sim_{ed}(0, j) = j$$

$$sim_{ed}(i, j) = min\{sim_{ed}(i - 1, j) + 1, sim_{ed}(i, j - 1) + 1,$$

$$sim_{ed}(i - 1, j - 1) + d(d1_i, d2_j),$$

$$sim_{ed}(i - 2, j - 2) + d(d1_i, d2_{j-1}) + d(d1_{i-1}, d2_j) + 1\},$$

whereby the function $d(d1_i, d2_j)$ counts the distance between two letters.

The more measures we have that agree on two (disease) terms as being “related”, the more stars are visible as our recommendation for the clinician. However, the decision whether the recommendation is in line with the user’s expectations is taken by the clinician and includes the outcome of π_{user} .

For example, n-gram measures such as Weighted Jaccard make it possible to measure the similarity based on specific words and ignore others which are expected to be unimportant. But how should we know which tokens are unimportant? Formally, the Weighted Jaccard measure takes two disease names d_1, d_2 and computes (only) the fraction of tokens that are present in both multi-word terms.


5 Evaluation

We evaluated the results of our individual similarity measures and found some special characteristics of the measures when applied to our specific data. The Weighted Jaccard method is useful to crop off stop words like “disease” as in “Castleman’s Disease” to map against “Castleman”, but is completely useless for, e.g., the stages of a disease case, a factor extremely important for lymphoma cases. This outcome reveals that when using this measure for medical linked data, it would be based on the wrong assumption that medical concept names contain negligible tokens. But our staging information and related type information is coded in those tokens. For example, the LinkedCT terms “Lymphocytic Leukemia, L1” and “Lymphocytic Leukemia, L2” differ only in the stage number, but this information is essential for finding related trials and drugs.

But also the traditional string similarity measures such as *sim_{ed}* cannot be used without caution. For example, in [16] we emphasised (in the context of the FMA ontology) that it contains closely related concepts such as “Anterior cervical lymph node” and “Set of anterior cervical lymph nodes” which could not be identified as duplicates with the *sim_{ed}* function. The variation of the linguistic surface forms might vary even to a greater extent when taking multiple ontologies (i.e., LinkedCT and Diseasesome) into account. On the other hand, *sim_{ed}* works very well in identifying related staging cases when only the stage numbers or similar sub-expressions are different (cf. the “Lymphocytic Leukemia L1 / L2”).

As a result of the observation that several distance functions have different performances depending on the characteristics of the data (such as length of the string, token permutations, etc.) we evaluated our ensemble measure with the “star” recommendations. Since the functions are independently applied to the disease names d_1, d_2 and aggregated into a combined measure by specifying the thresholds and weights of the individual distance function calls, a vast improvement in the robustness could be achieved. A closer examination of the disease names stored in LinkedCT and Diseasesome provides an explanation: many disease name contain long digit sequences, for instance ‘G09330582163324’ (LinkedCT). In most cases the digits refer to important type, stage, and grade

Lymphoma, Mantle cell (LinkedCT)



	Disease Name (Diseasome)	Trivial ED	Tr ED Weighted	Jaccard	Jaro Winkler
(1)	Lymphoma, mantle cell	1	1	0.076	0.031
(2)	Lymphoma, T-cell	5	5	0.076	0.121
(3)	Lymphoma, MALT	8	8	0.037	0.134
(4)	Lymphoma, somatic	9	9	0.333	0.169
(5)	Leukemia, acute myelogenous	1000(max)	1000(max)	0.444	1000(max)

Fig. 3. Mantle Cell Lymphoma as input from LinkedCT

	Disease Name (Diseasome)	Quality	Select
(1)	Lymphoma, mantle cell	★★★★	<input checked="" type="checkbox"/>
(2)	Lymphoma, T-cell	★★★	<input checked="" type="checkbox"/>
(3)	Lymphoma, MALT	★★	<input type="checkbox"/>
(4)	Lymphoma, somatic	★★	<input type="checkbox"/>
(5)	Leukemia, acute myelogenous	★	<input type="checkbox"/>

Fig. 4. Mantle Cell - Choice box for the proposed and selected “related”-mappings to Diseasome

information which is covered by the combined measure. The type, stage, and grade information we finally optimised for, are the following:

1. the type of the disease: “Diabetes Mellitus, Type 1” , “Diabetes Mellitus, Type 2” ,
2. the stage of the disease: “Early Stage Breast Cancer (Stage 1-3)” ,
3. the age of the patient as an indirect type classification : “ICU Patients 18 Years or Older” ,
4. the date of the disease outbreak: “2009 H1N1 Influenza” ,
5. genetic information; location of deleted region in the chromosome: “22q11.2 Deletion Syndrome” .

We then evaluated the interactive procedure in the context of our lymphoma case. The lymphoma case reveals that LinkedCT enumerates 459 lymphoma disease URIs (and Diseasome only 25 URIs) with the same name observations. Since either the LinkedCT term or the Diseasome term often lacks context information, we can only rely on the interactive workflow because the context is provided by the context factors explained above—and which are accessible to the expert while focussing on the suggestions the system provides. Here is an example of the individual measure outcomes for “Mantle Cell Lymphoma” from LinkedCT (figure 3), for which no links to Diseasome exist.

The aggregated “star recommendation” is attached to the suggested choice box for the as hoc “related” mappings (figure 4).

Linkedct-Result: Trial NCT00723099 (extern)	
Endpoint Classification: Efficacy Study, Intervention Model: Single Group Assignment, Masking: Open Label, Primary Purpose:Treatment (Interventional)	
LinkedCT-Diseases: Stage IV Cutaneous T-cell Non-Hodgkin Lymphoma, Stage III Cutaneous T-cell Non-Hodgkin Lymphoma, Stage II Cutaneous T-cell Non-Hodgkin Lymphoma, Stage I Cutaneous T-cell Non-Hodgkin Lymphoma, and Recurrent Cutaneous T-cell Non-Hodgkin Lymphoma	LinkedCT-Drugs: allogeneic hematopoietic stem cell transplantation, umbilical cord blood transplantation, total-body irradiation, mycophenolate mofetil, cyclosporine, cyclophosphamide, and fludarabine phosphate
Disease-Diseases:	Dailymed-Drugs: DOXYCYCLINE HYCLATE, Mycophenolate Mofetil, Cyclosporine, Cyclophosphamide, Sodium Chloride, and Fludarabine Phosphate
(October 2010)	Fred Hutchinson Cancer Research Center
Linkedct-Result: Trial NCT00808171 (extern)	
Allocation: Randomized, Endpoint Classification: Safety/Efficacy Study, Intervention Model: Parallel Assignment, Masking: Double Blind (Subject,Caregiver, Investigator, Outcomes Assessor), Primary Purpose: Supportive Care (Interventional)	
LinkedCT-Diseases: Non-Hodgkin Lymphoma	LinkedCT-Drugs: nitrous oxide
Disease-Diseases: Non-Hodgkin lymphoma	Dailymed-Drugs:
(December 2008)	Federal University of Minas Gerais

Fig. 5. Results of the Trial search

MEDICO Patients
Trial Overview
Radlex
Details
MEDICO RadSpeech

Query 105 T
Query 105 1
Query 105 b

5 recurrent-cutaneous-t-cell-non-hodgkin-lymphoma

5 stage-iv-cutaneous-t-cell-non-hodgkin-lymphoma

5 edqueryresult filtered from 20 originally (Reset All Filters)

Disease name (Disease)	Quality -	Select
Non-Hodgkin lymphoma	★★★★★	🔍
Non-Hodgkin_lymphoma	★★★★★	🔍
Prader-Willi_syndrome	★	🔍
Hypochondroplasia	★	🔍
Hypodontia	★	🔍

Disease (LCT)
2 Adult Non-Hodgkin Lymphoma

Disease (DIS)
47 (missing this field)
58 Non-Hodgkin lymphoma

Linkedct-Result: Trial NCT00723099 (extern)

Endpoint Classification: Efficacy Study, Intervention Model: Single Group Assignment, Masking: Open Label, Primary Purpose:Treatment (Interventional)

LinkedCT-Diseases: Stage IV Cutaneous T-cell Non-Hodgkin Lymphoma, Stage III Cutaneous T-cell Non-Hodgkin Lymphoma, Stage II Cutaneous T-cell Non-Hodgkin Lymphoma, Stage I Cutaneous T-cell Non-Hodgkin Lymphoma, and Recurrent Cutaneous T-cell Non-Hodgkin Lymphoma

Disease-Diseases:

LinkedCT-Drugs: allogeneic hematopoietic stem cell transplantation, umbilical cord blood transplantation, total-body irradiation, mycophenolate mofetil, cyclosporine, cyclophosphamide, and fludarabine phosphate

Dailymed-Drugs: DOXYCYCLINE HYCLATE, Mycophenolate Mofetil, Cyclosporine, Cyclophosphamide, Sodium Chloride, and Fludarabine Phosphate

Fig. 6. Display and Selection of the Proposed ad hoc Mappings

The choice box is being displayed within the faceted browsing tool. Our tool allows a user to filter thousands of results according to the best-ranked string matches and linked data relations in a convenient way. Clinician can retrieve the trails they might be interested in in just a few seconds. We implemented the graphical user interface by using open-source knowledge management tools

such as Exhibit, see <http://www.simile-widgets.org/exhibit/>. Figure 5 shows the embedded iPad GUI widget at stage 4 of the proposed workflow. The circle below indicates the trivial matching case - an equality link was provided by LinkedCT. The circle above shows the interesting case, the Disease disease is unknown. A click on a particular Stage I, II, or III lymphoma case evokes the ad hoc similarity search which results in the interactive mapping suggestion displayed in figure 6.

6 Conclusion

We explained how an LODD application based on diseases, drugs, and clinical trials can be used to improve the clinical reporting process. In order to get information about trails, drugs, and diseases, several LODD sources can be addressed and the contained knowledge can be combined. The clinical problem statement suggested that in order to make the application useful for improving the patient follow-up treatment process, specific non-existing mappings must be provided. The important mapping task lies between LinkedCT and Disease in the context of a more complex medical workflow which we developed. We argued that ad hoc interactive string-based ontology alignments should be used to propose several ad hoc mappings to the user. The user can then verify them at the knowledge retrieval stage while using the faceted browsing tool which implements the graphical user interface of the proposed workflow. An evaluation has shown that the interactive approach is useful and that underspecified “related” mappings are often more useful than precise “equality” mappings in the medical domain. In general, the discovery of such “related” links which are typed, dynamic, and bound to a specific user, data, and application context, should be an active research area.

Acknowledgements. We would like thank all colleagues at DFKI and all partners in the MEDICO and RadSpeech projects for their valuable contributions to the iterative process described here. This research has been supported by the THESEUS Programme funded by the German Federal Ministry of Economics and Technology (01MQ07016).

References

1. Doan, A., Madhavan, J., Domingos, P., Halevy, A.: Ontology matching: A machine learning approach. In: Handbook on Ontologies in Information Systems, pp. 397–416. Springer (2003)
2. Euzenat, J., Shvaiko, P.: Ontology matching. Springer, Heidelberg (2007)
3. Falconer, S.M., Noy, N., Storey, M.-A.D.: Towards understanding the needs of cognitive support for ontology mapping, vol. 225 (2006)
4. Kalfoglou, Y., Schorlemmer, W.M.: Ontology mapping: The state of the art. In: Semantic Interoperability and Integration (2005)
5. Langlotz, C.P.: Radlex: A new method for indexing online educational materials. RadioGraphics 26, 1595–1597 (2006)

6. Noy, N.F.: Tools for mapping and merging ontologies. In: *Handbook on Ontologies*, pp. 365–384 (2004)
7. Noy, N.F., Rubin, D.L.: Translating the foundational model of anatomy into OWL. *Web Semant.* 6, 133–136 (2008)
8. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. *The VLDB Journal* 10, 334–350 (2001)
9. Robertson, G.G., Czerwinski, M.P., Churchill, J.E.: Visualization of mappings between schemas. In: *CHI 2005: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 431–439. ACM, New York (2005)
10. Seifert, S., Kelm, M., Möller, M., Mukherjee, S., Cavallaro, A., Huber, M., Comaniciu, D.: Semantic annotation of medical images. In: *Proceedings of SPIE Medical Imaging*, San Diego, CA, USA (2010)
11. Sonntag, D.: Embedded benchmarking and expert authoring for ontology mapping and alignment generation. In: *Proceedings of the Fifth International Conference on Formal Ontology in Information Systems, FOIS* (2008)
12. Sonntag, D., Schulz, C., Reuschling, C., Galarraga, L.: Radspeech, a mobile dialogue system for radiologists. In: *Proceedings of the International Conference on Intelligent User Interfaces, IUI* (2012)
13. Sonntag, D., Wennerberg, P., Buitelaar, P., Zillner, S.: Pillars of Ontology Treatment in the Medical Domain. In: *Cases on Semantic Interoperability for Information Systems Integration: Practices and Applications*, pp. 162–186. Information Science Reference (2010)
14. Sonntag, D., Wennerberg, P., Zillner, S.: Applications of an ontology engineering methodology. *AAAI Spring Symposium Series* (2010)
15. Wennerberg, P., Zillner, S., Möller, M., Buitelaar, P., Sintek, M.: KEMM: A Knowledge Engineering Methodology in the Medical Domain. In: *Proceedings of the 2008 Conference on Formal Ontology in Information Systems*, pp. 79–91. IOS Press, Amsterdam (2008)
16. Zillner, S., Sonntag, D.: Aligning medical ontologies by axiomatic models, corpus linguistic syntactic rules and context information. In: *Proceedings of the 24th International Symposium on Computer-based Medical Systems, CMBS* (2011)

Towards Fuzzy Query-Relaxation for RDF

Aidan Hogan¹, Marc Mellotte¹, Gavin Powell², and Dafni Stampouli²

¹ Digital Enterprise Research Institute (DERI),
National University of Ireland, Galway, Ireland
{aidan.hogan,marc.mellotte}@deri.org

² Innovation Works, European Aeronautic Defence and Space Company (EADS),
Newport, UK
{gavin.powell,dafni.stampouli}@eads.com

Abstract. In this paper, we argue that *query relaxation* over RDF data is an important but largely overlooked research topic: the Semantic Web standards allow for answering crisp queries over crisp RDF data, but what of use-cases that require approximate answers for fuzzy queries over crisp data? We introduce a use-case from an EADS project that aims to aggregate intelligence information for police post-incident analysis. Query relaxation is needed to match incomplete descriptions of entities involved in crimes to structured descriptions thereof. We first discuss the use-case, formalise the problem, and survey current literature for possible approaches. We then present a proof-of-concept framework for enabling relaxation of structured entity-lookup queries, evaluating different distance measures for performing relaxation. We argue that beyond our specific scenario, query relaxation is important to many potential use-cases for Semantic Web technologies, and worthy of more attention.

1 Introduction

RDF is a flexible data format, and is well-suited to data integration scenarios. However, specifying precise queries over integrated, incomplete, heterogeneous data is much more challenging than likewise in closed, homogeneous settings. Writing precise queries requires precise knowledge of the modelling and content of the data. Even if a querying agent knows its exact information needs—and is able to specify those needs in a crisp, structured request—often, the query will not align well with heterogeneous data. Further, a querying agent may not be able to specify the precise “scope” of answers it is interested in, but may instead only be able to specify some “ideal” criteria that would be desirable.

Current Semantic Web standards and tools only go so far towards matching the needs of the querying agent and the content of the dataset. RDFS and OWL only facilitate finding more *crisp answers* to queries—answers matched directly by the data or its entailments—and do not directly support a continuous notion of *distance* (or *similarity*) for resources. For example, a query asking for a *2012 blue sports car on sale in New York* may also be interested in a *2010 navy roadster on sale in Newark*. Although the subsumption relationship between a sports car and a roadster could be modelled in RDFS, the distance of resources

such as blue/navy (vs. blue/red) and New York/Newark (vs. New York/Los Angeles) cannot be succinctly axiomatised in RDFS/OWL for interpretation by the query answering system. Instead, we argue that the RDF descriptions of resources can be used to compute an inductive, generic notion of distance.

In this paper, we thus advocate a relaxed form of RDF query-answering where the query should be interpreted as specifying the *ideal* criteria for answers, such that other relevant (but non-crisp) scored answers are returned. This is similar to top- k query-answering for Information Retrieval engines: a paradigm that works well in highly-heterogeneous, incomplete scenarios, including Web search.

We first present an industrial use-case from the European Aeronautic Defence and Space Company (EADS) that requires matching witness observations against crisp knowledge integrated from various law-enforcement and intelligence agencies (§ 2). Next, we provide a survey of literature that relates to the needs of EADS’ use-case and to query relaxation (§ 3). We then propose a generic framework for building a relaxed RDF query engine (§ 4); we currently focus on entity-lookup queries using similarities of RDF terms. Subsequently, we discuss a generic technique for extracting distance/similarity scores between resources based on their structured descriptions (§ 5). We then outline an early prototype for relaxing queries—that represent witness observations—against a dataset of vehicle descriptions, testing different similarity measures (§ 6). We conclude that our own results are too preliminary for deployment, but argue that RDF query relaxation (or more generally “fuzzy querying”) is an important, timely research topic not only for EADS’ use-case, but may unlike various potential and diverse Semantic Web applications involving vague/uncertain user requirements.

2 Use-Case Overview

Our use-case arises from an on-going research project at the European Aeronautic Defence and Space Company (EADS): a large European aerospace, defence and military contractor. EADS Innovation Works is the corporate research and technology department of EADS that explores areas of mobility, security and environment. One of the team’s key interests relates to civilian security, and enabling increased agency collaboration through use of intelligent systems. EADS has been working on the development of systems for intelligence analysis to aid post-crime police investigations [30], where analysts need to process raw information, determine valuable evidence, and identify the entities involved and their relationships. Investigations often rely on human observations, including police reports, or statements from victims, witnesses and informers.

Such data are the result of subjective assessment and often carry inherent vagueness and uncertainty. Human observations provide an *estimate* of the entity observed, described in natural language, and may be imprecise (i.e., stating that a suspect was 1.77 m tall when (s)he was 1.79 m tall) or vague (i.e., “*between 1.7 m – 1.85 m*”, or “*average height*”, etc.). Previous work [28,29] analysed issues with using human intelligence data (HUMINT), and presented methods to align data in different formats (numeric, textual and ranges). Herein, we view such observations as structured *fuzzy queries* to be executed over crisp data.

Posing complex but potentially vague/approximate queries against a crisp index broaches various issues. Numeric attributes—such as height—can be made “vague” using standard range queries. However, answers will not be scored: those near the “sweet-spot” of the range are not distinguished from those near the borders of *plausibility*. Given multiple attributes, the need for scoring becomes more pronounced. Further, given non-numeric attributes (e.g., a navy getaway car, probably a Toyota), standard query answering offers no support for vagueness. Ideally, the query engine should return ranked approximate answers by relaxing both numeric and non-numeric values. Further, the query should allow for specifying different levels of relaxation for different attributes; e.g., that Toyota is more vague and should be relaxed more than navy in the results.

We first considered applying ontologies and deductive methods to the use-case, looking to formalise the domain and to use fuzzy/annotated reasoning and querying techniques [20]. We abandoned this approach since (i) no legacy formal models were already in use; (ii) creating a speculative formal model from scratch (and fostering agreement) was deemed infeasible given the idiomatic nature of observations; (iii) inference would be too coarse-grained given the low resolution of formal knowledge available for the scenario. Instead, we decided to pursue an *inductive* approach, touching upon the area of *cooperative answering*.

3 Background and State-of-the-Art

Cooperative answering involves the application of Grice’s Maxims [13]—which describe how to be helpful in conversation—to information systems. Cooperative answering covers a broader area than our *current* scope, also trying to detect/circumvent user misconception, to provide additional justifications for the answer set, and to include entailed answers (as per deductive reasoning). We refer the reader to a survey of seminal works by Gaasterland et al. [10] and to a more recent survey of cooperative databases by Chu [5]. A pertinent technique in the area of cooperative answering is *query relaxation* or *generalisations*, whereby the criteria specified by the user are “relaxed” to include further, relevant content in the answers [9]. For example, Schumacher and Bergmann [27] propose using similarity measures to perform query relaxation in support of case-based reasoning over databases. Bruno et al. [3] investigate the use of query relaxation for efficient top-*k* retrieval over numeric, multi-attribute database relations.

Recently, some authors have looked at *deductive* query relaxation mechanisms for RDF [16,17,7,25]. Huang et al. [16], Hurtado et al. [17] and Poulouvasilis & Wood [25] all propose using RDFS semantics to perform relaxation, through, e.g., generalising query patterns up class or property hierarchies, etc. Dolog et al. [7] use query rewriting rules to perform logical relaxation based on explicit user-preference models. For our use-case, logical relaxation *alone* is too coarse; e.g., members of the same class are viewed analogously for type relaxation, etc.

Other authors have proposed similarity-based, inductive query relaxation for RDF [19,8], but focus on lexical analyses. Kiefer et al. [19] propose integrating “customised similarity functions” into SPARQL, allowing for string similarity

measures such as Levenstein, Jaccard and TF-IDF to be invoked. More recently, Elbassuoni et al. [8] propose relaxation based primarily on measuring entity similarity as the Jensen–Shannon divergence of the language models of virtual prose documents constructed for each entity. Again, such lexical similarity techniques are too shallow for our use-case (consider `red` vs. `blue` vs. `navy`); they still do not leverage the rich, *structured* descriptions of entities during matching. (Herein, we may use the dual terms *distance* and *similarity* interchangeably.)

Where numerical values are consistently defined for entities (e.g., lat./long. for places, or $L^*a^*b^*$ triplets for colours, etc.), distances can be computed based on Euclidean spaces where each attribute is a dimension (i.e., $\sqrt{\sum_{i=1}^n (a_i - b_i)^2}$ for a_i, b_i comparable numerical values for entity A and B resp.). However, matching based on categorical attributes—which are prevalent in RDF data—is more difficult [2]. An *overlap measure* coarsely assigns a constant distance d (typically $d = 1$) between entities that do not match for a given categorical attribute, or zero distance otherwise. Otherwise, Boriah et al. [2] survey a number of finer-grained methods for categorical matching; e.g., the Goodall measure [12] assigns a higher similarity between entities sharing a more selective category (i.e., a rare string value). Such measures are *data-driven*, relying on statistics from the data to compute distances: consequently, the similarity of two entities can be influenced by other peers (unlike, e.g., absolute Euclidean distances).

RDF similarity measures are also studied for *instance matching*. Although instance matching focuses on finding `owl:sameAs` alignments—and although many such frameworks rely on deductive methods (e.g., LN2R [26]) or lexical similarity methods (e.g., KNOFUSS [22], RDFSIM [18]) to generate alignments—a few inductive similarity metrics have been proposed based on overlaps in the descriptions of entities [23,15,14]; one such approach is later used in §5.

4 Relaxation Framework

We now propose a conceptual framework for relaxation of an *entity query*: a list of attribute–value or attribute–variable pairs $Q := \langle (p_1, o_1), \dots, (p_n, o_n) \rangle$ such that each p_i is a property URI and each o_i is either a variable, a URI or a literal (i.e., $Q \subset \mathbf{U} \times \mathbf{VUL}$)¹. A *crisp* response consists of entities (subjects) with predicate–object edges directly matching the query, as well as bindings for any variables in o_1, \dots, o_n . In SPARQL terms, this query model roughly corresponds to basic graph patterns with a common subject variable; for example:

```
SELECT * WHERE {?s :colour :blue ; :city :NY ; :type :Sport ; year 2010 ; reg ?r .}
```

To relax queries, we define a matcher as a function $M : \mathbf{VUL} \times \mathbf{UL} \rightarrow \mathbb{R}_{[0,1]}$ that maps a pair of values into a *relaxation score*: a value in $[0, 1]$ where 1 indicates that the two values are not interchangeable, and 0 indicates perfect interchangeability (e.g., $M(c, c) := 0$, $M(?v, c) := 0$). Each matcher is a *distance function* between the query and entity values, respectively. The match function

¹ The query is given an ordering for later convenience. We re-use standard RDF notation: \mathbf{V} denotes variables, \mathbf{U} URIs and \mathbf{L} RDF literals. AB denotes $A \cup B$.

may not be symmetric for pairs of values at different levels of specificity: e.g., $M(\text{:blue}, \text{:navy})$ might be 0.2 suggesting :navy as a good relaxation for generic :blue , whereas $M(\text{:navy}, \text{:blue})$ might be 0.5 since :navy is more specific.

Matchers then form the core of the relaxation framework, and can be instantiated in different ways (cf. [24]). For numeric attribute matchers (e.g. :year), normalised distances can be used: letting max_i and min_i denote the max./min. values for a numeric property p_i appearing in the data, qv_i a value in the query and ev_i a value for an entity, we can apply a normalised numeric matcher $M_i : (qo_i, eo_i) \mapsto \left| \frac{qo_i - eo_i}{max_i - min_i} \right|$. For string attributes with functional character strings (e.g., registration plates), lexical matchers can be used; we later use a Levenshtein edit-distance matcher for licence plates such that $M_i : (qo_i, eo_i) \mapsto \frac{Lev(qv_i, ev_i)}{\max(|qo_i|, |eo_i|)}$; other matchers can be used as appropriate. For *categorical* attributes—with URIs or a discrete set of literals as values (e.g., colour , city)—creating a matcher often requires background knowledge about the different values; as per Schumacher and Bergmann [27], we thus propose to use a similarity table for such attributes, computed by a background matching process (discussed later in § 5).

Thus, the relaxation framework may involve multiple matchers: a toolbox of appropriate matchers can be offered to an administrator. Where a suitable matcher is not found for a pair of values, the query engine can resort to returning standard “crisp” answers, allowing for an ad-hoc, incremental relaxation framework. We currently do not consider inference or relaxation of properties etc.; our framework could perhaps be extended as per the literature surveyed in § 3.

For a query $Q = \langle (p_1, o_1) \dots (p_n, o_n) \rangle$ and entity E , the matchers generate a tuple of numeric distances $M_{i..n}(Q, E) = \langle d_1, \dots, d_n \rangle$. Considering the query as the origin, the matchers map entities onto points in an n -dimensional Euclidean space with each dimension ranging over $[0, 1]$ (a unit n -cube). Where an entity has multiple values for a given attribute, the closest to the query is used; where an entity is not assigned a query-attribute, the matcher returns 1 [2]. Thereafter, entities on the origin are crisp matches. Otherwise, the distance from an entity to the query-origin can be measured straightforwardly as a Euclidean distance (in this case, $\sqrt{\sum_{i=1}^n d_i^2}$), or with *root-mean squared deviation* (*RMSD*: $\sqrt{\frac{\sum_{i=1}^n d_i^2}{n}}$).

The overall distance from the query-origin to each entity gives an *overall relaxation score* that can be used to order presentation of results, or to perform top- k thresholding. Further, users can annotate query attribute–value pairs with a *vagueness* score that allows for controlling the relaxation of individual facets (e.g., to allow more relaxation for :colour than :city). Thus a *vague query* is defined as $Q' := \langle (p_1, o_1, v_1), \dots, (p_n, o_n, v_n) \rangle$ where $v_1, \dots, v_n \in \mathbb{R}_{[0,1]}$ (i.e., $Q' \subset \mathbf{U} \times \mathbf{VUL} \times \mathbb{R}_{[0,1]}$). A value v_i indicates a threshold for d_i such that the entity will only be considered a result if $d_i \leq v_i$ (e.g., if $v_1 := 0$, then (p_i, o_i) must have a crisp match). Thus, the origin and the coordinate $\langle v_1, \dots, v_n \rangle$ prescribe a region of space (an m -orthotope for m the number of non-crisp query attributes) within which results fall into, allowing to tweak relaxation results.

² Intuitively, this is the relaxed form of standard conjunctive query answering.

5 Generating Similarity Tables for Categorical Values

The query relaxation framework relies on matchers to define distances between attribute values. We discussed direct matchers for numerical values (based on normalised Euclidean distance/RMSD) and for string values (based on normalised edit distances), but left the generation of similarity tables for categorical values. Such tables can be generated in a number of ways. First, if the set of categorical values is small enough, tables can be generated manually (on a pay-as-you-go basis); however, the number of scores needed is quadratic for the number of values. Otherwise, tables can be (semi-)automatically generated by any form of similarity measure; e.g., certain categorical attributes (like colours or places) are described using numeric attributes, allowing use of Euclidean distances.

Background information about the values can also be used to compute similarity scores. For instance, given a sufficient *unstructured* corpus, distributional semantic relatedness [11] can generate similarities between terms based on their co-occurrences in prose text. Given a sufficient *structured* RDF corpus describing the terms, instance-matching techniques can be used to generate similarity measures. Herein, we investigate the latter approach, using a generic RDF similarity technique which we had previously proposed [14], viz. *concurrency*, which is designed to cope with diverse RDF corpora; the core premises of concurrency have also been extended for the purposes of instance matching by other authors [15]. We now introduce the concurrency algorithm, which we use later (§ 6) to mine make-model similarity scores from an RDF corpus extracted from DBpedia.

Concurrency matches RDF resources based on their shared property-value pairs (generalising in-links and out-links: i.e., considering both *sp* and *po* pairs). Values are considered purely categorical, such that only crisp matches on shared pairs are used. Similarity is influenced more by pairs that are determined to be highly selective (i.e., to be exclusive): if a group of resources share a categorical value, the similarity score between these resources is boosted proportionate to the size of the group. This is similar in principle to Goodall’s measures [12][2].

Since RDF assumes an Open World (i.e., incomplete data), the concurrency method is “monotonic”: similarity is not punished when two resources have different values for an attribute since attributes can have multiple values, and (without OWL machinery checking cardinalities and ground (in)equalities), it is difficult to ascertain whether or not two RDF resources necessarily *disagree* on an attribute value (as opposed to just the *known* values differing). Instead, each additional shared pair monotonically increases the similarity score.

Formally, for an RDF dataset, let $\text{card}(p, v)$ denote the number of resources that share a given property-value pair (abstracting subject/object directionality), indicating how exclusive that pair is in the data. Next, let $\overline{\text{card}}(p)$ denote the mean cardinality of p across all such values v in the data, indicating how exclusive the property is (in general). The base similarity score given to all resources sharing the pair (p, v) is then defined as $\text{concur}(p, v) := \frac{1}{\text{card} \times \overline{\text{card}}(p, v)}$, which returns a value in $[0, 0.5)$ if the pair (p, v) is shared at least once.

Now, two resources sharing multiple pairs are given a multiset of *concur* scores $C := \{c_1, \dots, c_n\}$. The concurrency method combines these using a *probabilistic*

sum, such that $\perp_{\text{sum}}(c_a, c_b) := c_a + c_b - c_a \times c_b$. Since this function is associative and commutative, it is well-defined for the multiset C (i.e., by summing pairs in whichever order)³. The function is also monotonic and outputs a value in $[0, 1]$ (assuming $\forall c_i \in C(0 \leq c_i \leq 1)$). The intuition behind this aggregation function is that each additional match score reduces the current distance between the two resources by a product of itself; for example $\perp_{\text{sum}}(0.5, 0.5) = 0.75$, or $\perp_{\text{sum}}(0.8, 0.2) = 0.8 + (1 - 0.8) \times 0.2 = 0.84$. As per the examples, the probabilistic sum gives stronger weight to individual high values than an arithmetic sum.

The baseline concurrence method is also adjusted to account for some cases of dependence between property-value pairs. To reduce the effect of groups of (e.g., sub-)properties being given the same value—i.e., pairs $\{(p_1, v), \dots, (p_n, v)\}$ —only the most exclusive such pair (for which $\text{concur}(p_i, v)$ gives the highest score) will be used, and the rest discarded. To reduce the effect of repeated shared values for a given property—i.e., pairs $\{(p, v_1), \dots, (p, v_n)\}$ —all shared pairs for each property p are (probabilistically) summed up separately to a maximum value of $\overline{\text{card}}(p)$, with these scores then summed to an overall total.

Finally, generating quadratic pair-wise similarity scores may not be feasible for large datasets. Thus, a threshold t (s.t. $t > 2$) is specified for large-scale scenarios where pairs $\text{card}(p, v) > t$ are ignored: the number of matches generated are quadratic wrt. $\text{card}(p, v)$ but the respective scores are inverse-proportional. For example, if $\text{card}(\text{type}, \text{vehicle}) = 50,000$, this shared pair would require generating $\frac{50,000^2 - 50,000}{2}$ atomic matches with a score of $< \frac{1}{50,000}$. The threshold thus allows for pruning potentially massive volumes of low-scoring matches.

Concurrence is implemented using batch processing techniques (sorts/scans), and have been demonstrated on a corpus of one billion quadruples of openly-crawled Linked Data (with $t := 38$). Further details are available in [14].

6 Proof of Concept

In this section, we investigate proof-of-concept for the original use-case.

6.1 Vehicles Scenario and Data

Relating to the crime observation use-case, we focus on vehicle descriptions: automobile observations are often integral to police investigations, and it would be easier to find rich, publically available, representative, structured datasets for the vehicles use-case than, e.g., for weapons or criminals.

The case study was then to match structured, partial, possibly vague and imprecise queries against a crisp set of car instances. This dataset would emulate information provided by the UK Driver and Vehicle Licensing Agency (DVLA), or a Home Office database of recently stolen cars, etc. An observation by a witness would be something like "a blue getaway car that looked like a Toyota with

³ The probabilistic sum (aka. algebraic sum) is the dual t -conorm of the product t -norm; it also refers to the probability of a disjunction of independent events.

an ‘LD’ licence plate”. The relaxation framework is used to derive a ranked list of cars from the dataset in order of their relevance to the observation. In this respect, the observation acts like a query which should be executed against the car instances. Results should include not only those cars which directly match the characteristics given in the observation, but also similar cars. Different characteristics of the observation can be annotated with different vagueness values.

For demonstration purposes, we decided that the chosen dataset should contain information about a significant number of car instances, with attributes for (at least) make, model, colour and body-type, covering common facets of vehicle observations. We thus took an existing structured dataset describing 50,000 car instances based on a popular Irish website advertising used cars. Each car instance is described using the following six properties: vehicle make (48 unique values; e.g., Toyota), make–model (491 values; e.g., Toyota Corolla), body style (8 values; e.g., Saloon), fuel type (5 values; e.g., Diesel), colour (13 values after normalisation; e.g., navy), and registration (i.e., unique licence plate; 50,000 values). Taking the raw data, colours were normalised into a set of thirteen defined values, a new set of UK-style licence plates was randomly generated, and the data were modelled in RDF using Hepp’s Vehicle Sales Ontology (VSO)⁴

Notably, all vehicle attributes except licence-plates are categorical, and thus require tables that encode similarity/distance scores. To relax licence-plate values, we allow wildcard characters in the query and use the normalised Levenshtein measure mentioned earlier. For colour, the thirteen values were mapped to an $L^*a^*b^*$ three-dimensional colour space, where *Delta-E* was used to compute (Euclidean) distances between the colours and generate a matrix for relaxation⁵. An open, non-trivial challenge was then posed by the other properties. For fuel-type, it was particularly unclear what kind of relaxation behaviour should be expected for the use-case; a set of regular distance scores were manually defined. Of more interest were the make–model, model and body-style attributes, for which further background information was needed.

6.2 Mining DBpedia for Background Information

To acquire a background, structured corpus on the make and make–model values appearing in the data, we enriched the baseline RDF data with selected DBpedia exports⁶ from Wikipedia: we noted that DBpedia exported a rich set of data about many of the vehicle make–models, including width, height, categories, engine, transmission, etc. The resulting RDF data would then serve as input for similarity techniques to generate scores for subsequent query relaxation.

The first step was to map string values in the data (e.g., Toyota Corolla) to DBpedia URIs (e.g., http://dbpedia.org/resource/Ford_Mondeo). First attempts were made using the reconciliation function of the “RDF Extension for Google Refine”⁶, which allows for mapping strings to resources in a SPARQL

⁴ Available at <http://www.heppnetz.de/ontologies/vso/ns>; retr. 2011/12/12.

⁵ Distances in the $L^*a^*b^*$ colour space correspond more closely with discrepancies in human perception than RGB/CMYK models^[32].

⁶ See <http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/>; retr. 2011/12/12/

Table 1. Top ten make–model matches overall (left) and for distinct makes (right)

Nº	TOP OVERALL MATCHES		Nº	TOP MATCHES ACROSS MAKES	
1	Honda_Accord	Honda_Odyssey	8	Audi_80	Volkswagen_Passat
2	Mercedes-Benz_S-Class	Mercedes-Benz_SL-Class	11	Audi_A3	Skoda_Octavia
3	Suzuki_Alto	Suzuki_Wagon_R	13	Audi_TT	SEAT_León
4	Ford_Tourneo_Connect	Ford_Transit_Connect	25	Citroën_C1	Peugeot_107
5	Mitsubishi_Colt	Mitsubishi_Mirage	27	Hyundai_i10	Kia_Picanto
6	Volvo_S60	Volvo_S80	28	Audi_A3	Seat_León
7	Nissan_Maxima	Nissan_Murano	30	Daewoo_Winstorm	Opel_Antara
8	Audi_80	Volkswagen_Passat	38	Hyundai_Tuscan	Kia_Sportage
9	Mercedes-Benz_S-Class	Mercedes-Benz_W220	39	Citroën_C3	Hyundai_Getz
10	SEAT_Córdoba	SEAT_Ibiza	40	SEAT_Toledo	Volkswagen_Jetta

endpoint and refining these mappings in a second phase [21]. Although numerous matches were found, many make–models were not reconciled to DBpedia URIs.

Instead, we adopted a manual approach by appending make–model strings onto the DBpedia namespace URI, replacing space with underscore. However, this approach also encountered problems. First, of the 491 string values, 68 models (14%) did not have a corresponding reference in DBpedia (404 Not Found): some of the unmatched models were colloquial UK/Irish names (e.g., the make–model Citroen Distpatch is known elsewhere as Citroën Jumpy), some were misspelt, and some had encoding issues. These values were manually mapped based on suggestions from Wikipedia search. Second, some of the matches that were found returned little data. Of these, some were redirect stub resources (e.g., Citroen C3 redirects to Citroën C3), where we “forwarded” the mapping through the redirect. Others still were disambiguation pages (e.g., Ford Focus disambiguates to Ford Focus International, Ford Focus America and Ford Focus BEV). Furthermore, some resources redirected to disambiguation pages (e.g., Ford Focus ST redirect to the Ford Focus disambiguation page). Here, we mapped strings to multiple resources, where Ford Focus was mapped to the set of DBpedia resources for { Ford Focus, Ford Focus America, Ford Focus International and Ford Focus BEV }. In total, 90 strings (18.3%) had to be manually mapped to DBpedia. Given the mappings, we retrieved 53k triples of RDF data from DBpedia, following redirects and disambiguation links.

We applied concurrence over the dataset with a threshold $t := 200$ (each shared pair with $\text{card}(p, v) = 200$ would generate $>20,000$ raw concur scores at least below 0.005). For the 491 original make–model values, concurrence found non-zero similarity scores for 184k (76%) of the 241k total car-model pairs possible, with an average absolute match score of 0.08 across all models.

The top ten overall results are presented in Table 1, where we also present the top ten matches for models with different makes; note that matches are symmetric and that all matches presented in the table had a similarity score

exceeding 0.99 indicating highly-relaxable values. Similarity scores are convertible to distance scores for relaxation using $dist = 1 - sim$ ⁷

The results were of varying quality for our use-case. For the top make-model match—Honda Accord/Honda Odyssey—the former is a saloon and the latter is a minivan, and as such, the two models are physically diverse; however both models share specific/exclusive components (e.g., the same engine) which causes the high concurrence score. Another such example is given by Nissan Maxima/Nissan Murano; also the Seat Cordoba is the saloon version of the SEAT Ibiza hatchback. This raises a more general issue with the subjective nature of relaxation: although these pairs may be dissimilar for a witness-observation use-case, they are similar from a sales or manufacturing perspective. Still, we see some potentially interesting relaxation for the witness use-case. Many of the highest-scoring cars are physically similar; besides those of the same make, many of the matches with different makes are (or have been) based on the same platform: i.e., are fundamentally the same car with different skins.

Besides manual inspection of top-ranked results, in the absence of a gold standard, evaluating the large set of concurrence results from DBpedia is difficult. Although inspection yielded promising examples of relaxation, conversely, unintuitive/undesirable results were also discovered. For example, the Westfield SE (a rare model of “kit” sports-car) had very little information in DBpedia, and hence had no notable concurrence matches; similar problems were encountered for other rare models.

A major disadvantage of concurrence is that distances are not geometric in nature: considering resources A , B , C as points, then the concurrence “distance-vectors” \vec{AB} , \vec{BC} and \vec{AC} cannot be considered as true spatial vectors since the additive property thereof does not hold: $\vec{AB} + \vec{BC} \neq \vec{AC}$. Thus, for example, if concurrence gives a score of ~ 1 indicating near-perfect similarity between B and C (distance of ~ 0), this does not imply that B and C have similar scores to other cars ($|\vec{BC}| \approx 0 \not\approx |\vec{BA}| \approx |\vec{CA}|$). Taking an example from our evaluation, the Audi RS4 and Audi A4 were matched closely with 0.97; the Audi RS4 and Audi A8 models were matched with a score of 0.8; but the Audi A4 and Audi A8 models were matched with a low score of 0.29 despite being very closely matched to a common model: the first two pairs gained their concurrence scores through a largely distinct (incomplete) attributes. This results in unintuitive, incomparable distance scores when looking beyond just pairs of resources.

Furthermore, the results of the concurrence method is heavily dependent on the underlying data. DBpedia does not distinguish between different editions of make-models down through the years, where for example, the RDF data for six diverse editions of Ford Fiesta—spanning forty years—are presented together under one resource since they are described in one Wikipedia article (albeit with separate info-boxes). Due to these, and other issues relating to data quality, we decided to pursue tests over a smaller, cleaner dataset.

⁷ Alternatively, the similarities could be normalised into a non-parametric, rank-based distance, where a relaxation value of 0.5 includes the top-half similar models.

6.3 Comparing Concurrency vs. Numerical Matching

For evaluating the concurrency method, we wanted to compare its results against standard Euclidean distances over numerical attributes. We attempted to extract numerical values from the DBpedia make–model data, but the presence of multiple values per attribute and the incompleteness of the data precluded any meaningful numerical analysis. Hence, a manual evaluation corpus was gathered directly from Wikipedia for which concurrency and standard numerical approaches could be compared. The corpus consisted of tabular data describing 200 make–model–edition values in terms of numerical values and ranges (years produced, doors, engine capacity, wheelbase, length, width, height and curb weight) as well as categorical values (make, model, edition, body-style); the dataset was, however, incomplete as not all values could be found for all editions surveyed.

Since this evaluation dataset focuses primarily on numeric values, we modified the discrete concurrency algorithm to consider continuous values. Given a set of property-value pairs $\{(p, v_1), \dots, (p, v_n)\}$ with each $v_i \in \mathbb{R}$ a value for a specific car-edition, and given two resources (editions) with pairs (p, v_a) and (p, v_b) respectively (where $v_a \leq v_b$), we define the numeric cardinality between the two values as $\text{ncard}(p, v_a, v_b) := |\{v_i : \exists(p, v_i) \text{ and } v_a \leq v_i \leq v_b\}|$, denoting the number of resources that fall in the inclusive range of those two values for that property $[v_a, v_b]$, assuming single-valued attributes for brevity. The concurrency score for a crisp categorical value becomes $\text{concur}(p, v) := \frac{1}{\text{card}(p, v)}$ and for a numeric value becomes $\text{concur}(p, v_a, v_b) := \frac{1}{\text{ncard}(p, v_a, v_b)}$. Further, we turned off concurrency’s thresholding and dependence filters, which were not required for the clean, small dataset at hand (reverting to a simple probabilistic sum).

We then measured the correlation between results for the modified concurrency algorithm, and for non-normalised RMSD (distances not divided by the $\text{max} - \text{min}$ denominator) and normalised RMSD (distances for each attribute pre-normalised into $[0, 1]$) computed over numerical attributes. To quantify the correlation, we used *Kendall’s* τ , which measures correlation between two orderings: the τ value is in the range $[-1, 1]$ where -1 indicates a perfect negative correlation (the orderings are “reversed”), 0 indicates independence of orderings, and 1 indicates a perfect correlation (the same ordering). The correlation between the concurrency and non-normalised RMSD was positive but weak (~ 0.1): without the pre-normalisation of values, attributes with larger units (such as years in the thousands) tended to have high influence. However, between concurrency and normalised RMSD values, the correlation was higher at ~ 0.54 : the main difference was attributable to the monotonic nature of concurrency, which did not punish mismatches between single values to the same extent as the normalised RMSD measure. RMSD had the more favourable results in this regard due in part to the clean nature of the data. Conversely, concurrency gave better matches for incomplete data, where RMSD gave very high scores.

Finally, we compiled the 40,000 make–model–edition similarity scores for each approach into similarity scores for make–models, makes and body-styles by taking the average across all pairs in the generalised groups. For example, to generate a similarity score between `saloon` and `hatchback` body styles (say the sets S

and H resp.), we took the average of all edition-similarities between both groups (i.e., the arithmetic mean of scores for all edition-pairs in $S \times H$).

6.4 Experimenting with Query Relaxation

With various matcher mechanisms and tables in hand, we turned to testing query relaxation against the 50k vehicles dataset. We developed a simple prototype to take a vague query, perform a full scan of the dataset computing relaxation scores for each entity based on the matchers, and return a ranked list of answers. Although the query algorithm is linear, we acknowledge that sub-linear (and sub-second) query times might be required for deployment. There are various avenues to enable sub-linear performance (for entity search): (i) if crisp facets are given, these can be looked up directly where relaxation is then used to filter initial results (similar in principle to SPARQL FILTERs); (ii) given a matcher based on a table of similarities or on numeric distance, the corresponding vagueness score for the query facet can be used to compute a range query that can be executed as a table lookup (assuming an index with the correct sort order is available); (iii) special relaxation indices can be built, for example, using Locality Sensitive Hashing to index Euclidean points and enable efficient neighbourhood searches [4]. Different optimisations are feasible for different types of matchers. We leave further investigations of optimisations for related work: our current aim is to validate the proposed techniques and offer proof-of-concept.

Table 2 presents the top-5 results for three example witness observations, which were modelled as structured vague queries and run against the vehicles dataset. Vagueness scores are manually chosen for proof-of-concept: mapping textual vagueness to numeric vagueness is out of scope. For the matchers, we used a normalised Levenshtein edit-distance for licence plates; the $L^*a^*b^*$ -based similarity table for colour; and for make, model, edition and body-style, we used three configurations with similarities computed from the 200 vehicle-editions dataset (for which we could compute three sets of results) using (i) concurrence, (ii) normalised Euclidean and (iii) absolute Euclidean distance measures. The scores are based on RMSD from the query origin (subtracted from one).

OBSERVATION A gives an example of relaxation for colours; however, note that the original query also requests relaxation for models, but where colour distances are typically much shorter. No difference occurs between the different matcher configurations for car-make (the first relaxed car-make appears in position 10–11). This is a weakness of the framework: different matchers may produce incomparable distances, creating an “imbalance” in the relaxation across different attributes; a possible solution is the use of rank-based distances. From OBSERVATION B, we see **maroon** being returned as a crisp match for **red** and see example relaxation of models; from the scoring, we also note that different matchers may vary in terms of inclusiveness. Finally, OBSERVATION C uses the Levenshtein edit-distance matcher for licence-plates in combination with colour relaxation, where the **black** result is questionable.

In the absence of a gold standard, we could not evaluate precisely the effectiveness of the relaxation framework for generating relevant, approximate,

Table 2. Example observations, relaxed queries and results

OBSERVATION A:		“A greenish car, maybe a Peugeot”.					
RELAXED QUERY:		{(colour, green, 0.2), (make, Peugeot, 0.8)}					
N₂		ALL APPROACHES (SAME RESULTS)					
	<i>result</i>					<i>score</i>	
1	Peugeot			green			1.00
2	Peugeot			yellow			0.96
3	Peugeot			brown			0.95
4	Peugeot			teal			0.95
5	Peugeot			aqua			0.93

OBSERVATION B:		“A red SUV, looked like a Land Rover Freelander”.					
RELAXED QUERY:		{(colour, red, 0), (body, SUV, 0), (model, LR.-Freelander, 0.8)}					
N₂		CONCURRENCE		NORM. EUCLIDEAN		ABS. EUCLIDEAN	
	<i>result</i>	<i>score</i>	<i>result</i>	<i>score</i>	<i>result</i>	<i>score</i>	
1	LR. Freelander red	1.00	LR. Freelander red	1.00	LR. Freelander red	1.00	
2	LR. Freelander maroon	1.00	LR. Freelander maroon	1.00	LR. Freelander maroon	1.00	
3	Hyundai Trajet red	0.86	Hyundai Tuscon red	0.93	Hyundai Tuscon red	0.84	
4	Kia Sorento red	0.86	Hyundai Tuscon maroon	0.93	Hyundai Tuscon maroon	0.84	
5	Kia Sorento maroon	0.86	Kia Sorento red	0.92	Renault Scenic red	0.84	

OBSERVATION C:		“A light Audi A3 8L, 2006 UK reg. starts with SW and ends with M”.					
RELAXED QUERY:		{(colour, white, 0.4), (edition, Audi-A-8L, 0.1), (reg, SW?6??M, 0.4)}					
N₂		ALL APPROACHES (SAME RESULTS)					
	<i>result</i>					<i>score</i>	
1	Audi A3 8L		SW06RWM		yellow	0.92	
2	Audi A3 8L		SF56GCN		white	0.91	
3	Audi A3 8L		BW06LJN		gray	0.90	
4	Audi A3 8L		SW04TVH		black	0.85	
5	Audi A3 8L		AE56MWM		maroon	0.83	

well-graded answers. Generating high-quality distance scores based on categorical values is much more challenging than for numeric attributes [2], but a crucial part of inductive query relaxation for RDF. Table 2 provides some preliminary results towards query relaxation for RDF, but based on the outlined problems, the results were deemed currently unsuitable for deployment in the use-case. However, we believe that with further investigation, such methods can be improved and adapted for use in other less critical applications, such as relaxing query results from public SPARQL endpoints.

7 Conclusion

In this paper, we introduced a use-case from EADS involving matching witness observations against structured entity descriptions. We proposed query-relaxation as a framework within which to tackle the problem. We discussed how matchers can be used to enable query relaxation, how different matchers can be combined and used for different attributes, and how RDF similarity techniques can be used to compile similarity scores for categorical values. We presented the results of various proof-of-concept experiments with the goal of performing query-relaxation over 50k car descriptions. We discussed using DBpedia to

mine background information for make-model similarity scores, computed using our proposed *concurrency* method. We subsequently compared the correlation between the results of a modified version of our concurrency method and that of standard Euclidean distance measures. Finally, we presented some example query-relaxation results based on vague observations of vehicles as per the use-case. Unfortunately, the results were not deemed reliable enough for deployment.

As such, lots more work is left to do and many challenges are left unaddressed. Beyond our use-case, we argue that cooperative answering and query relaxation is an important, timely topic for Semantic Web researchers to pursue: RDF stores often index diverse datasets with complex schemata, against which writing precise queries is extremely challenging. Query relaxation would then find application in various areas, including Web search and recommender systems [3], e-commerce [6], case-based reasoning [27], reconciliation [21], etc. As discussed, current instance matching techniques can be repurposed for relaxation.

In summary, we would hope to see further proposals towards “cooperative SPARQL engines” which intelligently aid users—using a mixture of deductive and inductive techniques—in the daunting task of answering potentially vague queries over diverse RDF datasets. We have taken tentative steps in this direction, looking at query relaxation for entity queries. Further focus on inductive techniques—like those proposed by Stuckenschmidt [31] or Hu et al. [15]—will also better position the Semantic Web community to support applications needing “intelligence” beyond *just* crisp semantics and logics.

Acknowledgements. This paper was funded in part by the Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2). We thank Nuno Lopes, Axel Polleres, Ali Hasnain and Maciej Dabrowski for their contributions.

References

1. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - a crystallization point for the Web of Data. *J. Web Sem.* 7(3), 154–165 (2009)
2. Boriah, S., Chandola, V., Kumar, V.: Similarity measures for categorical data: A comparative evaluation. In: *SDM*, pp. 243–254 (2008)
3. Bruno, N., Chaudhuri, S., Gravano, L.: Top-*k* selection queries over relational databases: Mapping strategies and performance evaluation. *ACM Trans. DB Syst.* 27(2), 153–187 (2002)
4. Chaudhuri, S., Datar, M., Narasayya, V.R.: Index selection for databases: A hardness study and a principled heuristic solution. *IEEE Trans. Knowl. Data Eng.* 16(11), 1313–1323 (2004)
5. Chu, W.W.: Cooperative database systems. In: *Wiley Encyclopedia of Computer Science and Engineering*, John Wiley & Sons, Inc. (2008)
6. Dabrowski, M., Acton, T.: Modelling preference relaxation in e-commerce. In: *FUZZ-IEEE*, pp. 1–8 (2010)
7. Dolog, P., Stuckenschmidt, H., Wache, H., Diederich, J.: Relaxing RDF queries based on user and domain preferences. *J. Intell. Inf. Syst.* 33(3), 239–260 (2009)

8. Elbassuoni, S., Ramanath, M., Weikum, G.: Query Relaxation for Entity-Relationship Search. 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. 62–76. Springer, Heidelberg (2011)
9. Gaasterland, T.: Cooperative answering through controlled query relaxation. *IEEE Expert* 12(5), 48–59 (1997)
10. Gaasterland, T., Godfrey, P., Minker, J.: An overview of cooperative answering. *J. Intell. Inf. Syst.* 1(2), 123–157 (1992)
11. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In: *IJCAI*, pp. 1606–1611 (2007)
12. Goodall, D.W.: A new similarity index based on probability. *Biometrics* 22(4) (1966)
13. Grice, P.: Logic and conversation. *Syntax and Semantics* 3 (1975)
14. Hogan, A., Zimmermann, A., Umbrich, J., Polleres, A., Decker, S.: Scalable and distributed methods for entity matching, consolidation and disambiguation over Linked Data corpora. *J. Web Sem.* 10, 76–110 (2012)
15. Hu, W., Chen, J., Qu, Y.: A self-training approach for resolving object coreference on the semantic web. In: *WWW*, pp. 87–96 (2011)
16. Huang, H., Liu, C., Zhou, X.: Computing Relaxed Answers on RDF Databases. In: Bailey, J., Maier, D., Schewe, K.-D., Thalheim, B., Wang, X.S. (eds.) *WISE 2008*. LNCS, vol. 5175, pp. 163–175. Springer, Heidelberg (2008)
17. Hurtado, C.A., Poulouvasilis, 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)
18. Ioannou, E., Papapetrou, O., Skoutas, D., Nejdl, W.: Efficient Semantic-Aware Detection of Near Duplicate Resources. 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. 136–150. Springer, Heidelberg (2010)
19. Kiefer, C., Bernstein, A., Stocker, M.: The Fundamentals of iSPARQL: A Virtual Triple Approach for Similarity-Based Semantic Web Tasks. 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. 295–309. Springer, Heidelberg (2007)
20. Lopes, N., Polleres, A., Straccia, U., Zimmermann, A.: AnQL: SPARQLing Up Annotated RDFS. 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. 518–533. Springer, Heidelberg (2010)
21. Maali, F., Cyganiak, R., Peristeras, V.: Re-using cool URIs: Entity reconciliation against LOD hubs. In: *LDOW* (2011)
22. Nikolov, A., Uren, V.S., Motta, E., De Roeck, A.: Integration of Semantically Annotated Data by the KnoFuss Architecture. In: Gangemi, A., Euzenat, J. (eds.) *EKAW 2008*. LNCS (LNAI), vol. 5268, pp. 265–274. Springer, Heidelberg (2008)
23. 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)
24. Oldakowski, R., Bizer, C.: SemMF: A framework for calculating semantic similarity of objects represented as RDF graphs. In: *ISWC (Poster Proc.)* (2005)

25. Poulouassilis, A., Wood, P.T.: Combining Approximation and Relaxation in Semantic Web Path Queries. 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. 631–646. Springer, Heidelberg (2010)
26. 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)
27. Schumacher, J., Bergmann, R.: An Efficient Approach to Similarity-Based Retrieval on Top of Relational Databases. In: Blanzieri, E., Portinale, L. (eds.) EWCBR 2000. LNCS (LNAI), vol. 1898, pp. 273–284. Springer, Heidelberg (2000)
28. Stampouli, D., Brown, M., Powell, G.: Fusion of soft information using TBM. In: 13th Int. Conf. on Information Fusion (2010)
29. Stampouli, D., Roberts, M., Powell, G.: Who dunnit? An appraisal of two people matching techniques. In: 14th Int. Conf. on Information Fusion (2011)
30. Stampouli, D., Vincen, D., Powell, G.: Situation assessment for a centralised intelligence fusion framework for emergency services. In: 12th Int. Conf. on Information Fusion (2009)
31. Stuckenschmidt, H.: A Semantic Similarity Measure for Ontology-Based Information. In: Andreasen, T., Yager, R.R., Bulskov, H., Christiansen, H., Larsen, H.L. (eds.) FQAS 2009. LNCS, vol. 5822, pp. 406–417. Springer, Heidelberg (2009)
32. Tkalčič, M., Tasič, J.F.: Colour spaces: perceptual, historical and applicational background. In: IEEE EUROCON, pp. 304–308 (2003)

Learning Driver Preferences of POIs Using a Semantic Web Knowledge System

Rahul Parundekar and Kentaro Oguchi

Toyota InfoTechnology Center, U.S.A.
Mountain View, CA

{rparundekar,koguchi}@us.toyota-itc.com

Abstract. In this paper, we present the architecture and implementation of a Semantic Web Knowledge System that is employed to learn driver preferences for Points of Interest (POIs) using a content based approach. Initially, implicit & explicit feedback is collected from drivers about the places that they like. Data about these places is then retrieved from web sources and a POI preference model is built using machine learning algorithms. At a future time, when the driver searches for places that he/she might want to go to, his/her learnt model is used to personalize the result. The data that is used to learn preferences is represented as Linked Data with the help of a POI ontology, and retrieved from multiple POI search services by ‘lifting’ it into RDF. This structured data is then combined with driver context and fed into a machine learning algorithm that produces a statistical model of the driver’s preferences. This data and model is hosted in the cloud and is accessible via intelligent services and an access control mechanism to a client device like an in-vehicle navigation system. We describe the design and implementation of such a system that is currently in-use to study how a driver’s preferences can be modeled by the vehicle platform and utilized for POI recommendations.

Keywords: Semantic Web, Linked Data, Knowledge Base, Preference Modeling.

1 Introduction

The in-vehicle navigation system tries to minimize driver distraction by enforcing restrictions on the way information is presented to the driver. One such constraint restricts the number of items that can be displayed in a list on a single screen to a fixed number of slots. When the driver searches for banks in the car, for example, the search results are displayed as a list filled in these slots. If the number of search results exceeds the number of slots, then the extra results are pushed to the next page. A more personalized experience can be provided by showing these results sorted according to the driver’s preferences. Using the history of Points of Interests (POIs) that the driver visits, we can build a model of what kind of places he/she (henceforth referred to as *he*) is more likely to prefer. For example, the driver may have certain preferences for banks. We want

to understand his bank preferences so that they can be used to personalize the result the next time he is searching for one. Our system, which uses a Semantic Web Knowledge System to explore this personalization, is called Semantic User Preference Engine or *Supe*. The places that the driver visits are stored in our Knowledge Base as data that is structured using RDF. We first use a Machine Learning algorithm to build a preference model of the places that the driver likes from this history. The next time a driver searches for places of a certain category, we use this model to re-order the search results according to his/her estimated affinity.

We describe how *Supe* is used to study driver preferences of POIs in this paper as follows. First, we discuss our system and how it relies on RDF and Linked Data to represent information. Then, we describe how our the ontology and the Linked Data about the preferred POIs is collected from multiple sources and used to build the preference model for the driver. This includes how the Linked Data is translated into the desired input for Machine Learning algorithms that were used. This is followed by describing how *Supe* uses the built preference model to reorder a result of a POI search to match the driver’s estimated affinity of these places. We then explain the implementation and the evaluation of our prototype system. We also include relevant work in using Linked Data and Semantic Web for modeling user preferences and recommendations. Lastly, we conclude by summarizing our findings along with future work.

2 The *Supe* System

2.1 Overview

The primary objective of *Supe* is to collect driver preferences and use the preference model to provide personalized POI search results to the driver. To be successful in modeling driver preferences, it needs to understand the driver as well as the POIs. By defining the semantics in a Knowledge Base, we tie the

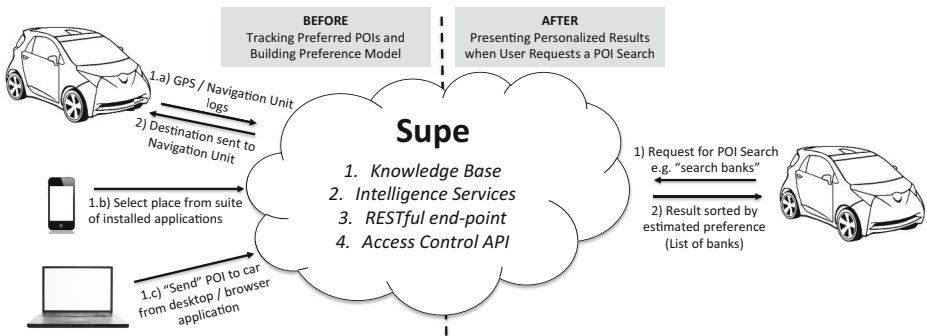


Fig. 1. *Supe* system Overview

understanding of place and driver data to an ontology. Supe also provides Intelligent Services, which use a machine learning engine in the background, for the presentation of the preferred places to the driver. Fig. 2 depicts the system overview.

The Semantic Web Knowledge System is at the heart of Supe and resides in the cloud. It contains a Knowledge Base, Intelligent Services, RESTful endpoints and access control mechanisms which are described in detail in Section 2.2. Supe is also connected to multiple devices that help collect POI data for the driver. Applications running on these devices use the exposed services to search and look-up places from multiple sources on the web. A place that the driver selects from the search results is accessible on the navigation system in his vehicle. For example, the driver might want to ‘send’ a POI from their desktop before getting into the car. Alternatively, they may select a place from their smart phones using their personal suite of installed applications and send it to the vehicle to be used as a destination for the in-vehicle navigation system. A third scenario finds places that the driver has visited using the history (GPS logs, etc.) of the in-vehicle head unit. In all three cases, Supe keeps track of the place consumed (i.e. navigated to, called, etc.) in order to add it to the driver’s preferences. Once the preference model is built, it can be used to personalize the POI search in the in-vehicle navigation system. When the navigation system requests a place search, e.g. a bank, the system first retrieves POIs that match the search criteria, reorders these results according to the driver’s preferences and returns the list to the navigation system.

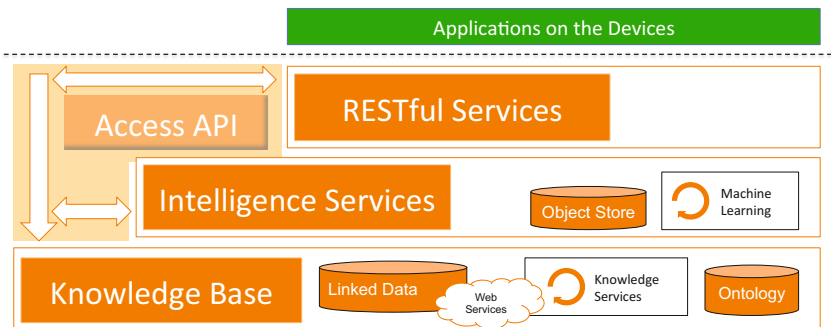


Fig. 2. Semantic Web Knowledge System Layer Cake

2.2 The Supe Semantic Web Knowledge System

Fig. 2 shows the architecture of Supe using a layer cake diagram. The components of the Semantic Web Knowledge System are discussed below.

Knowledge Base. The Supe Knowledge Base is based on Semantic Web and Linked Data standards. It is responsible for describing and storing the driver and place information. It also allows for using Web Services on-the-fly as a source for real-time place data in RDF. The Knowledge Base is composed of the following:

1. **Ontology:** Place-data in the system is represented using an RDFS ontology that covers the POI domain. The ontology defines a concept hierarchy of places based on their categories along with the properties associated with each of those place types. Fig. 3 shows a small subset of the ontology. It also has supporting concepts for representing user information, like home or work addresses, etc. All concepts and properties in the ontology belong to the *ontology namespace*.

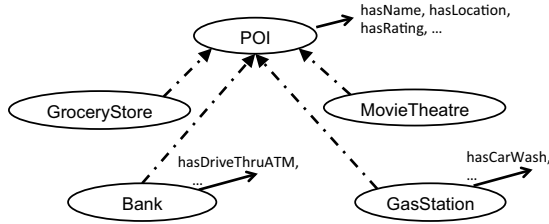


Fig. 3. Place Ontology (subset)

2. **Linked Data:** All instances for place and user data in the Knowledge Base are identified by URIs belonging to the *instance namespace* and are represented in RDF. Internally, the Intelligent Services can access this data as server objects that can be modified locally using the Jena framework [9]. Any data is also accessible in RDF or JSON [4] to the applications running on the devices for easy consumption as dereferencable URIs using the Linked Data Design Principles [2]. Locally, the RDF data for each instance is grouped together into an *instance molecule* and stored in a database with its URI as the primary key. We base our grouping of Linked Data triples into instance molecules by specializing the browse-able graph [2] terminology, where triples that have the URI for the instance in the object part of the triple are absent. Grouping triples into instance molecules allows us for a loosely coupled browse-able graph of data without duplicates. The search look-up is realized using a separate index that uses only the location and category. Since we are using Jena on the server side, any necessary RDFS inferences on the molecule can be performed when it is retrieved from the Knowledge Base. A similar concept to instance molecule, defined as *RDF molecule*, is also found in Ding et al. [5].

We also create a Linked Data wrapper around multiple sources on the web to find and retrieve POI instance data. For example, we can access each place already present on Yelp as Linked Data, as we wrap the Yelp API [1] and ‘lift’ the associated place data into our ontology. We use similar wrappers for other sources, like Google Places API [2], or data stored locally in an RDBMS store. Lifting all the data into the same ontology, allows us to integrate multiple sources and also create a richer set of information for modeling driver preference by merging such data.

¹ http://www.yelp.com/developers/getting_started/api_overview

² <http://code.google.com/apis/maps/documentation/places/>

3. **Knowledge Services:** If an instance molecule corresponds to an object and its attributes in an *object oriented programming language*, then Knowledge Services correspond to its functions. These services control what operations can be performed on the instance molecule (e.g. ensuring that the triples conform to the forward-only browse-able RDF graph terminology). A Knowledge Service can also build on other services. For example, the service that adds user and situation context to a place instance, uses the instance molecule modification knowledge service. Another use of Knowledge Services is to create more sophisticated data structures like lists, stacks, queues, etc. for the instances. For example, the POI Search service, given the location and category, uses one of the POI Web Services to search for places and then presents the result as a sorted list of instance molecules to the requesting Intelligent Services.

Intelligent Services: These are responsible for the intelligence in Supe. It contains the machine learning component that is used for modeling the driver's preferences and presenting a personalized search result. Since we use Linked Data to represent knowledge, the Intelligent Services need to convert RDF to data suitable for the machine learning algorithm used. The Intelligent Services are also supported with a database where the objects that it needs to persist, e.g. the driver's preference model, can be stored. These objects are also identified using URIs. The Intelligent Services for building preferences and personalizing search results are described in Section 2.3 and Section 2.4.

RESTful Services: POI data and Intelligent Services are exposed to client applications using RESTful end-points. Peripheral services for modifying user data (like home or work address, etc.) or services that facilitate the scenarios from Section 2.1 are also included. The end-points also know if these services are to be executed synchronously or asynchronously, depending on whether the client device needs to wait on service output or not. For example, the client App does not have to wait while the preference model is getting updated with a new place instance.

Access Control API: Due to the highly personal nature of the data and to prevent RESTful services, Knowledge Services, Intelligent Services and applications running on the client devices (collectively referred to as *platform services* below) from corrupting other service or instance data, Supe also has an access control mechanism in place. Users and platform services use an identifier and a secret passkey combination for authentication. To simplify access control, we use an approach that restricts access based on namespaces rather than on individual instances. In order to access instances in the necessary namespace, an authentication challenge needs to be met successfully. Access control rules for the different namespaces are described below in brief:

1. **Ontology Namespace:** All platform services have access to the ontology.
2. **Instance Namespace:** All platform services have access to the Linked Data that is not user or service specific.

3. **Namespaces for each user:** All user specific data, like his home or work address, etc., that is not owned by any particular platform service belongs to his separate namespace. Access is granted to all platform services and client devices that can respond to user authentication challenge.
4. **Namespaces for each platform service:** Data belonging to a platform service but not specific to any user (e.g. service configuration, service static objects, etc.) belongs to this namespace. A platform service cannot access data in some other service’s namespace.
5. **Subspaces for each platform service within a user’s namespace:** All platform services with user specific data save it in a sub-space created for that service within the user’s namespace. A platform service cannot access data belonging to a user that it does not have the necessary authentication for. Alternatively, even if it does have user authentication, it cannot access another service’s data.

To access Linked Data or Intelligent Services internally, a platform service needs to pass the user and the service credentials to the Access Control API as function parameters. On the client side, any authentication parameters to be passed to the platform services are forwarded as HTTPS request query parameters (i.e. as a secured GET / POST query). HTTPS also ensures the privacy of the personal information transfer between the client applications and Supe. URIs in Supe start with ‘https://’ and are thus dereference-able as well as secure. Use of HTTPS relieves the application layer of the task of encryption of the data and makes the implementation of the system easier.

2.3 Building Driver Preferences from Tracked POI Data

When client applications send data to the in-vehicle navigation system, Supe tracks the visited/consumed POIs and stores them as driver history. This data is then used to build a statistical model of the driver’s place preferences by training a machine learning algorithm. This process, is described below. Fig. 4 shows the steps involved in converting POI data about a bank, which the user selected in the navigation system, to machine learnable data.

Fetching Linked Data for POIs (Lifting): Identifiers of POIs that are tracked are used to build the preference model. Data for these places is retrieved from multiple web sources using the *POI Search knowledge service*. This returns place data in RDF using the ontology after lifting the web service response. For example, when the user selects a bank called “Bank of America”, the lifting process converts the JSON response of the web service (e.g. Yelp API) into an instance molecule as shown in Fig. 4 (a). Lifting data to a single ontology also allows us to support multiple sources for the POI data. Different Web Services like Google Places API have different representations for data (e.g. different metrics, different category hierarchy, etc.). Search results from these sources can be integrated into the same instance molecule, using appropriate lifting logic.

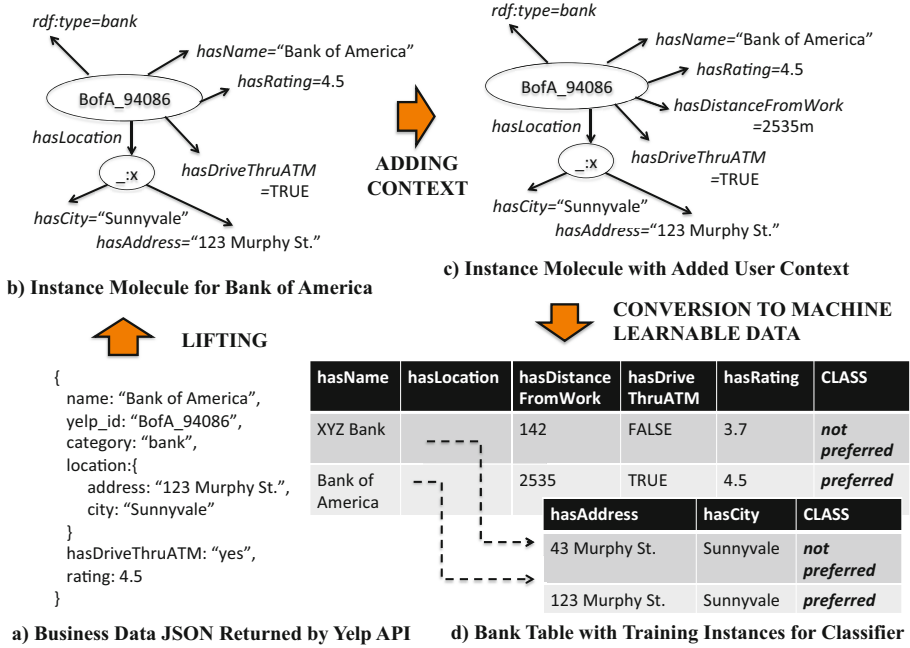


Fig. 4. Building User Preferences (Note: Only subset of the actual data used, is shown.)

Adding Context: A major design consideration for using an ontology for representation in Supe is to automatically extrapolate context that would help for a richer description of the data. The *Context knowledge service* is used to add user and situation context to POI instance molecule using rules programmed in the service. To add user data, it first loads the data for the driver from the local Linked Data store. Based on the programmed rules, it then automatically adds triples for the user context to the place data instance molecule. For example, if the user has his home or work address stored, then the service is able to add *distance from home* or *distance from work* context. As shown in Fig. 4 (b) and (c), user context *hasDistanceFromWork* is added to the Bank of America instance molecule. Another context that can be added is Situation. For example, information whether the visit was in the morning or evening along with the time spent can help understand the if the driver accessed an ATM or actually had to stop by at the bank for a longer time.

Conversion of RDF Instance Molecule to Machine Learnable Data: The instance molecule containing the POI and context data is used to learn the user’s preferences using a content based approach. This instance needs to be first converted to a representation that machine learning algorithms can understand. The translation from Linked Data to machine learning instances is relatively straight-forward. We explain this translation as a conversion of instance molecules into a table containing training data, as used by conventional machine learning algorithms, below. (Due to lack of space, Fig. 4 (d) only shows

a representative set of columns that can be derived from Fig. 4 (c). The tables are implemented as frequency counts for values in each column and persisted in the object store. The learning algorithm used is described later, in Section 2.4.

1. **The Table:** All instances belonging to a certain category are grouped together in a single table. For example, Fig. 4 (d) shows a subset of the table for banks.
2. **Rows in the Table:** Each instance to be added to the preference model translates to each row in the table. The URI can either be dropped from the training (since row identifiers usually do not contribute to the learnt model) or can alternatively be added as a new column in the table. We use the latter approach to bias the model with a higher preference to a previously visited place³.
3. **Columns in the table** The property-value pairs for the instance get translated as columns and values in the table. The properties for which the type of the table is a domain, appear as columns. For example, while the *hasDriveThruATM* is a property of Banks, the *hasName* property is inherited from the parent concept *POI* and is also present in the table in Fig. 4.
4. **Values in the Table:** RDF Literal values are translated as one of string, numeric or nominal values in a column. For example, the translation of the values for the *hasName* & *hasRating* properties from Fig. 4 (c). to columns in the table in Fig. 4 (d) is trivial. These translation rules can be specified explicitly in the ontology at construction time. Alternatively, a type detection mechanism could be used to identify the data types. For values of properties that are blank nodes, we use nesting of tables. This results in tracking inner frequency counts for the properties. Section 2.4 describes how the inner nested tables are used to compute preference. For values of properties that are URIs, we can choose to either (i) use the lexical value of the identifier as the value of the attribute in the machine learning table or (ii) represent the instance in a nested table, similar to blank nodes.
5. **Class Column in the Table:** The class value in the table for the training instances is marked as either ‘preferred’ or ‘notpreferred’ based on the way the data is tracked. The POIs in the driver history of visited places are marked as preferred (e.g. Bank of America is marked as *preferred* in the class column in Fig. 4 (d)).

Incrementally Building and Storing the Model: Instead of building the entire preference model from scratch every time a new instance is added, we use an incremental approach to building the model. Each time a new place is to be added to this model, the stored model is retrieved from the object store, updated and persisted back into the object store. To make sure that the performance doesn’t start degrading once the size of the model becomes large, it only contains the n -most recent preferred places. Optimization is also made to reduce disk writes by using a delayed write of the preference model object.

³ This column is not depicted in Fig. 4 (d) due to lack of space.

2.4 Ordering POIs According to User's Preference

Once the preference model contains some POI data, applications requesting a search of POIs can be presented with personalized results. In our use-case, the user might want to search for banks around him using the in-vehicle navigation system, when in a place that he does not know much about. The steps involved in realizing this are part of the *Personalized POI Search Intelligent Service*, and are described below.

Fetching POIs and Adding Context: The personalized POI search service uses the POI search knowledge service to retrieve places that match the search criteria from web sources. Necessary user and situation context are added to the list of instances, which were already lifted into RDF by the knowledge service, if necessary data is present. These two steps are similar to the steps described in Section 2.3. The search service then uses the Preference Scoring service to estimate the user's affinity to each place.

Scoring Each POI Using the Stored Preference Model: The user's preference model is loaded by the Preference Scoring Intelligent Service from the object store. Each place instance it receives is first converted into a form suitable for input to scoring by the machine learning algorithm. For calculating the preference of a POI to the user, we use a scoring function that computes the Euclidean distance, in a way similar to the nearest neighbor calculation in unsupervised classification algorithms. The scores for the POIs are returned to the search service. It can then sort the list of POIs according to the score and present the personalized list to the requesting client application.

Distance-from-Unity Metric: We introduce a similarity metric for preference that uses a 'nearest neighbor' approach to calculate the most preferred place. For an instance that is to be scored, we project the the likelihood probabilities for the values of its attributes on orthogonal axes. For a value of a property, the likelihood probability is calculated as $P('value' | preferred)$ based on the frequency count table of that column. On each axis, the likelihood probability gets projected as a continuous numeric function with maximum possible score of 1.0 for a value that is always preferred, and a score of 0.0 for a value that is absent from the table. The instance gets projected as a point in this multi-dimensional space. A hypothetical instance that will always be preferred will have its likelihood value 1.0 on all axes. The point representing this hypothetical instance will have the co-ordinates (1.0, 1.0, 1.0, 1.0, 1.0 ...) in the multidimensional space. We call this point as 'Unity'. For the instance that we want to score, we calculate the Euclidean distance of the projected point from Unity. The personalized search service can then sort the list of POIs according to this distance such that the instance whose distance is minimum is considered most preferred.

For literal values the probability is calculated as follows: (i) probabilities for numeric columns are calculated using a Gaussian distribution, (ii) probabilities for string values are calculated similar to a naïve-Bayes probability based document similarity metric and (iii) probabilities for nominal values are calculated

as the probability of the symbol occurring in the column. In case of numeric attributes, where we use a Gaussian Distribution, we normalize the probability density function by dividing it with the probability density of the mean. This puts the probability ‘distance’ between zero and one. To calculate the distance for object values for an attribute, we explode that attribute into its own hyperspace. We calculate the distance of the inner objects to the unity point in this hyperspace. Dividing this distance with the distance from origin to unity in that hyperspace normalizes it to a value between 0 and 1. The hyperspace for the attribute is then collapsed back to a single axis and the normalized distance value becomes distance from unity on that axis, in the original multi-dimensional space. For a multivalued property, we take the average of the distances of all its values. If we were to score the affinity for Bank of America from Fig. 4 (d) for testing, the calculation would be as follows.

$$D(BofA_94086) = \sqrt{\begin{matrix} (1 - P(\text{“Bank of America”} \mid preferred))^2 \\ +(1 - P(2353 \mid preferred))^2 + (1 - P(TRUE \mid preferred))^2 \\ +(1 - P(4.5 \mid preferred))^2 + D(_ : x)^2 \end{matrix}}$$

$$\text{Where } D(_ : x) = \sqrt{\frac{\begin{matrix} (1 - P(\text{“Sunnyvale”} \mid preferred))^2 \\ +(1 - P(\text{“123 Murphy St.”} \mid preferred))^2 \end{matrix}}{2}}$$

Naïve-Bayes Probability: To establish a baseline for comparison, we also use a second scoring function based on naïve-Bayes classification algorithms. For an instance to be scored, we calculate the naïve-Bayes probability of it being preferred. For two instances that are normalized, we can compare the probabilities of each of them belonging to the class ‘preferred’ to decide which one of them is more likely to be preferred by the user. The probabilities can be calculated easily by using the frequency count tables for the columns. The calculation of the probability for Bank of America from Fig. 4 (d) is shown below.

$$P(preferred \mid BofA_94086) = P(preferred) \times \frac{P(\text{“Bank of America”} \mid preferred)}{P(\text{“Bank of America”})} \times \frac{P(2353 \mid preferred)}{P(2353)} \times \frac{P(TRUE \mid preferred)}{P(TRUE)} \times \frac{P(4.5 \mid preferred)}{P(4.5)} \times \frac{P(_ : x \mid preferred)}{P(_ : x)}$$

$$\text{Where } \frac{P(_ : x \mid preferred)}{P(_ : x)} = \frac{P(\text{“Sunnyvale”} \mid preferred)}{P(\text{“Sunnyvale”})} \times \frac{P(\text{“123 Murphy St.”} \mid preferred)}{P(\text{“123 Murphy St.”})}$$

While using naïve-Bayes however, we take three important precautions. First, we also populate the table with negative training examples so that the fractions do not cancel each other out. In the absence of explicit information about the places that the driver dislikes, we train the model with instances, other than the one that was selected while performing the POI search, as belonging to the class *not preferred*. Second, unlike conventional classification, where the probability of an instance belonging to a certain class is compared with its probability in belonging to other classes, we compare the probability of one instance being preferred to the probability of another instance being preferred. This demands that we normalize all the instances (i.e. have the same set of properties and use the same frequency tables for calculating the probabilities) by ensuring that all instances have the same set of attributes by either assigning default values or dropping missing attributes for all instances. Third, in case of multivalued attributes, we take averages for the score of each value for ensuring that the scores are normalized.

3 Evaluation and Discussion

We first describe an initial evaluation of the scoring mechanisms using dummy users and then discuss the preliminary findings of an actual user study, currently in progress. To see if the scoring functions work, we constructed an experiment to build and test a preference model for dummy users. A dummy-user is a computer program that is able to perform selections of POIs based on an embedded user profile. Each (dummy) user performs a POI search task for places belonging to a category (e.g. restaurants, banks, grocery stores, etc.) and programmatically selects one POI (or multiple correct POIs) based on its profile. The profiles for these users are listed below:

1. uNearest: Always selects the nearest place.
2. uHighestRated: Always selects the highest rated place.
3. uBrand: Always selects certain brands like Starbucks, Chevron, etc.
4. uCategory: Always selects places belonging to certain sub-categories (e.g. Japanese restaurants, etc.)
5. uYelpOrder: Always trusts the system and selects the first item. In this case the list is ordered by the underlying Web Service(Yelp) that sorts places using a proprietary special mix.
6. uPerson: This user was modelled based on the preferences of a real person after asking him a set of questions and then programming his choices into the dummy user.

Training was done on 50 tasks in an area near the user's routine commute. For example, some of the search use-cases were 'find a gas station near home', 'find a restaurant for lunch near work', etc. Following this, testing was done on a set of 25 POI search tasks in an unknown location along with the fifty examples from the training set. The testing tasks were formulated on hypothetical use-cases of likely requests by the user in a new area. For example, some of the use-cases were 'find a restaurant near the airport (when the user picks a friend up)', "Find

a coffee shop in Santa Cruz while driving San Francisco for a break”, etc. We compared the two scoring functions where the accuracy was calculated as follows. For the list of places returned sorted according to either scoring mechanism, if the dummy user selection would have matched the first place, we count this as the preference modeling for the task as a success. Accuracy represents the % of successful tasks. The results of the experiment are shown below in Fig. 5. These two scoring mechanisms were selected for (i) checking that the preferences worked by manually checking ‘under-the-hood’ of the learnt models, and (ii) establishing a baseline for incorporating more sophisticated scoring techniques in the future.

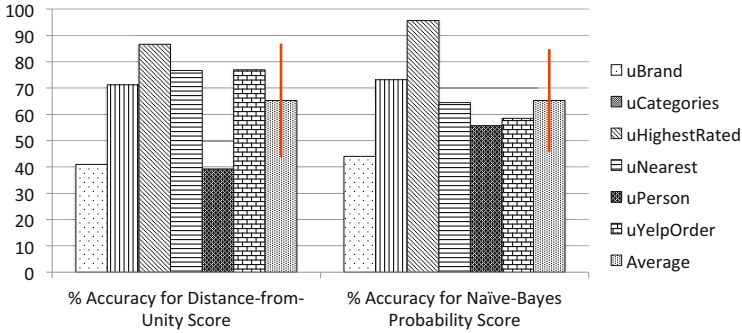


Fig. 5. Accuracy of Scoring Functions for Dummy Users

As these results were promising, we implemented a smart-phone application emulating the in-vehicle navigation system using client and server side implementations for a user study. We emulated the in-vehicle head unit by using a smart-phone app that allows users to search for POIs belonging to any category, while the cloud based server was implemented using the description of Supe above. Though it is still too early to provide experimental results on the effectiveness of the preference model for actual users, the following screenshot (Fig. 6) shows one case where a user found the preference model to work effectively. The first image shows the results for banks in the user’s daily commute area. The ones marked with a pin are the places that he has visited and are used to build his preference model. When the user was away from his regular zone and in need of cash, he searched for banks using the POI Search service. The second capture shows that two of his preferred banks were ranked among the top three in the list. Intuitively the reader may figure out that this user prefers a specific banking company. The model is able to detect this preference because the likelihood probability on the *hasName* axis is high. We are finding similar patterns in other users as well for other categories, like gas stations, restaurants, etc. and hope to present these results at the conference. The *Distance-from-Unity* metric was used to find the results shown in the screenshots. While the naïve-Bayes scoring function outperforms the *Distance-from-Unity* scoring function in the preliminary experiments, in practice users rarely have a rigid choice pattern like always selecting the nearest place. The preliminary experiment relied on the order information on properties like distance, rating, etc. to figure out these preferences.

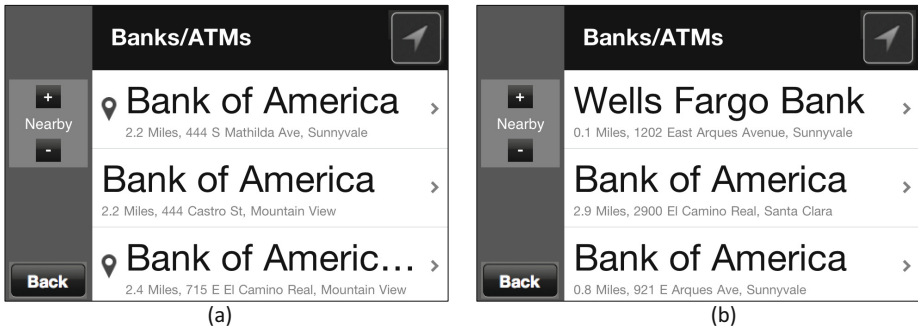


Fig. 6. Screenshots of the Implemented User Study Application: Search result for banks (a) in daily commute area (places previously visited are marked) (b) in a new area using Distance-from-Unity metric

In the absence of such explicit input, the *Distance-from-Unity* function was able to better hone in the driver's preferences than the naïve-Bayes scoring function and was chosen to personalize the search results.

4 Related Work

Before we discuss similar work that create content based preference models for users, we briefly describe other work related to parts of our system, which have been explored in the Semantic Web. The POI Search Knowledge Services that we define are wrappers around existing Web Services and produce RDF as output similar to Semantic Web Services [10]. They borrow the 'lifting' concept from work in Semantically Annotated WSDL services [7], where the output of the web service is 'lifted' from XML into a semantic model. In our case, we have programmed the logic for lifting in our different Knowledge Services that may choose from different sources like Yelp, Google Places for place data, since the domain is fairly constant. But there are other solutions (e.g. Ambite et. al [1]) that can be alternatively used to automatically find alignments between sources and construct Semantic Web Services. Though access control mechanisms for the Semantic Web have been explored for a long time, it has only actively been researched for the Linked Data more recently. Hollenbach et. al [6] describe a decentralized approach to access control for Linked Data at the document level. Wagner et. al [14] describe how policies can be implemented for controlling access to private Linked Data in the semantic smart grid. Our approach for controlling access does not include RDF descriptions of authentication and authorization credentials but is instead based on existing practices for Web Services which allow for easy integration with client devices. It is also easier to implement.

In the past year, the combination of RDF and machine learning has gained some traction. The work that is perhaps most similar to our approach on converting RDF to a machine learning table, is found in Lin et. al [8]. For the movie domain, they try to predict if a movie receives more than \$2M in its opening week by converting the RDF graph for the movie to a Relational Bayesian Classifier. One ma-

jor difference in their approach to ours is the translation of multi-valued object properties (e.g. `hasActor`) into the relational table. While doing so, their approach ‘flattens’ all objects and groups values into properties. For example, names of all actors get aggregated into a single ‘has actor name’ column and all actors’ year of birth get aggregated into a ‘has actor YoB’ column. In doing so, the associativity within an instance (e.g. of an actor’s name to his age) is lost. In our approach, the advantage of using a graph for instance representation which is the associativity between the properties is maintained since (i) the inner objects are scored independently and (ii) in case of multi-valued object properties, we take an average of individual scores. However, a more in-depth comparison of the two approaches needs to be conducted on common data for better analysis. Another related work in learning from RDF has been explored in Bicer et. al[3], where relational kernel machines are used for movie recommendations. Content based preference modeling has received large research attention in recommendation systems over the past few years[13]. These algorithms, use machine learning techniques like linear regression, naïve-Bayes, etc. for finding recommendations. For example, the Syskill & Webert system[12] uses a naïve Bayes-classifier for classifying web sites as either ‘hot’ or ‘cold’. Similar to our approach, this system also uses the probability score to rank pages according to user’s preferences. More recently, personalized information retrieval has also been gaining traction in the Semantic Web. For example, *dbrec*[11] is a music recommendation system based on DBpedia that uses a semantic link distance metric for its recommendations.

5 Conclusion and Future Work

In this paper we have described the architecture and implementation of a system that combines the Semantic Web, a Linked Data Knowledge Base, and machine learning to build a preference model of the places that a driver likes using a content-based approach. The use of a Semantic Web Knowledge System that combines knowledge and intelligence has advantages. First, it helps represent the preference model in a way that can be consistently interpreted by knowledge services as well as intelligent services. Second, data about places, user, etc can be enriched by integrating it from multiple sources, along with extrapolating user and situation context. Lastly, this RDF data can automatically be translated into a format compatible with machine learning algorithms for building the preference model. Preliminary results from our ongoing user study show that the preference model is able to predict the driver’s estimated affinity to a place. It can thus be used for POI recommendation in the vehicle, where we would like to minimize the driver interaction by providing a personalized experience.

Though our preliminary evaluation shows promising results with actual users for using simple metrics, it has scope for improvements. Since a major focus of the study is to understand the preference model and improve its accuracy, we intend to explore other machine learning techniques (e.g. linear regression) and feature selection techniques to improve the preference model, as future work. During the course of the user study, we are also dealing with performance issues of implementing a Semantic Web Knowledge System in the cloud, e.g. the overhead associated

with integrating from multiple sources, adding contexts, RDF to machine learning translations, etc. Optimizing this performance would prove as a great learning opportunity. Lastly, another direction for future work is in using the system for exploring other domains like music, navigation, etc. for personalization in the vehicle. We believe that Supe has the potential of graduating from an in-use intelligent system for studying and understanding driver preferences of POIs to a real world deployment that can provide a comprehensive personalized experience in the car.

References

1. Ambite, J.L., Darbha, S., Goel, A., Knoblock, C.A., Lerman, K., Parundekar, R., Russ, T.: Automatically Constructing Semantic Web Services from Online Sources. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 17–32. Springer, Heidelberg (2009)
2. Berners-Lee, T.: Design issues: Linked data (2006), <http://www.w3.org/DesignIssues/LinkedData.html>
3. Bicer, V., Tran, T., Gossen, A.: Relational Kernel Machines for Learning from Graph-Structured RDF Data. 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. 47–62. Springer, Heidelberg (2011)
4. Crockford, D.: The application/json media type for javascript object notation, json (2006), <https://tools.ietf.org/html/rfc4627>
5. Ding, L., Finin, T., Peng, Y., Da Silva, P., McGuinness, D.: Tracking RDF graph provenance using RDF molecules. In: Proc. of the 4th International Semantic Web Conference, Poster (2005)
6. Hollenbach, J., Presbrey, J., Berners-Lee, T.: Using rdf metadata to enable access control on the social semantic web. In: Workshop on Collaborative Construction, Management and Linking of Structured Knowledge (2009)
7. Kopecký, J., Vitvar, T., Bournez, C., Farrell, J.: Sawsdl: Semantic annotations for wsdl and xml schema. IEEE Internet Computing, 60–67 (2007)
8. Lin, H.T., Koul, N., Honavar, V.: Learning Relational Bayesian Classifiers from RDF 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. 389–404. Springer, Heidelberg (2011)
9. McBride, B.: Jena: A semantic web toolkit. IEEE Internet Computing 6(6), 55–59 (2002)
10. McIlraith, S.A., Son, T.C., Zeng, H.: Semantic web services. IEEE Intelligent Systems 16(2), 46–53 (2001)
11. Passant, A.: dbrec — Music Recommendations Using DBpedia. 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. 209–224. Springer, Heidelberg (2010)
12. Pazzani, M., Billsus, D.: Learning and revising user profiles: The identification of interesting web sites. Machine learning 27(3), 313–331 (1997)
13. Pazzani, M.J., Billsus, D.: Content-Based Recommendation Systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 325–341. Springer, Heidelberg (2007)
14. Wagner, A., Speiser, S., Raabe, O., Harth, A.: Linked data for a privacy-aware smart grid. In: INFORMATIK 2010 Workshop-Informatik für die Energiesysteme der Zukunft (2010)

An Approach for Named Entity Recognition in Poorly Structured Data

Nuno Freire^{1,2}, José Borbinha¹, and Pável Calado¹

¹ INESC-ID/Instituto Superior Técnico - Technical University of Lisbon,
Av. Rovisco Pais, 1049-001 Lisboa, Portugal

² The European Library, National Library of the Netherlands,
Willem-Alexanderhof 5, 2509 LK The Hague, Netherlands
{nuno.freire, jlb, pavel.calado}@ist.utl.pt

Abstract. This paper describes an approach for the task of named entity recognition in structured data containing free text as the values of its elements. We studied the recognition of the entity types of *person*, *location* and *organization* in bibliographic data sets from a concrete wide digital library initiative. Our approach is based on conditional random fields models, using features designed to perform named entity recognition in the absence of strong lexical evidence, and exploiting the semantic context given by the data structure. The evaluation results support that, with the specialized features, named entity recognition can be done in free text within structured data with an acceptable accuracy. Our approach was able to achieve a maximum precision of 0.91 at 0.55 recall and a maximum recall of 0.82 at 0.77 precision. The achieved results were always higher than those obtained with Stanford Named Entity Recognizer, which was developed for grammatically well-formed text. We believe this level of quality in named entity recognition allows the use of this approach to support a wide range of information extraction applications in structured data.

Keywords: named entity recognition, structured data, metadata, conditional random fields.

1 Introduction

A wide range of potentially usable business information exists in unstructured forms. Although that information is machine readable, it consists of natural language texts (it was estimated that 80% to 90% of business information may exist in those unstructured forms [1] [2]).

As businesses become more data oriented, much interest has arisen in these unstructured sources of information. This interest gave origin to the research field of *information extraction*, which looks for automatic ways to create structured data from unstructured data sources [3]. An information extraction process can be characterized by an intention of selectively structure and combine data that is found in text, either explicitly stated or implied. The final output of the process will vary according to the

purpose, but typically it consists in semantically richer data, which follows a structured data model, and on which more effective computation methods can be applied.

Information resources in digital libraries are usually described, along with their context, by structured data records. These data, which is commonly referred in the digital library community as *metadata*, may serve many purposes, and the most relevant being resource discovery. Those records often contain unstructured data in natural language text, which might be useful to judge about the relevance of the resource. The natural hypothesis is if that information can be represented with finer grained semantics, then the quality of the system is expected to improve.

This paper addresses a particular task of information extraction, typically called named entity recognition (NER), which deals with the textual references to entities, that is, when they are referred to by means of names occurring in natural language expressions, instead of structured data. This task deals with the particular problem of how to locate these references in the data set and how to classify them according their entity type [4].

We describe a NER approach, which we studied on the particular case of metadata from the cultural heritage domain, represented in the generic Dublin Core¹ data model, which typically contains uncontrolled free text in the values of its data elements. We refer to this kind of data as poorly structured data. Typical examples of such data elements are the titles, subjects, and publishing information.

NER has been extensively researched in grammatically well-formed text. In poorly structured data however, the text may not be grammatically well-formed, so our assumption is also that the data structure provides a semantic context which may support the NER task.

This paper presents an analysis of the NER problem poorly structured data, describes a novel NER approach to address this kind of data, and presents an evaluation of the approach on a real set of data. The paper will follow with an introduction to NER and related work in Section 2. The proposed approach is presented in Section 3, and the evaluation procedure and results are presented in Section 4. Section 5 concludes and presents future work.

2 Problem and Related Work

The NER task refers to locating atomic elements in text and classifying them into predefined categories such as the names of persons, organizations, locations, expressions of time, quantities, etc. [4].

Initial approaches were based on manually constructed finite state patterns and/or collections of entity names [4]. However, named entity recognition soon was considered as a typical scenario for the application of machine learning algorithms, because of the potential availability of many types of evidence, which form the algorithm's input variables [5]. Current solutions can reach an F-measure accuracy around 90% [4] in grammatically well-formed text, thus a near-human performance.

¹ <http://dublincore.org/>

However, previous work suggested that current NER techniques underperform when applied to texts existing within structured digital library records [6] [7] [8]. Most research on NER has focused mainly on natural language processing, involving text tokenization, part-of-speech classification, word sequence analysis, etc. Recognition with these techniques is therefore language specific and dependent of the lexical evidence given by the natural language text.

The most similar scenarios we are aware of have researched information extraction within poorly structured data, and with a different focus than us. Research described in [9] proposes the use of information extraction techniques within relational database management systems, in order to exploit existing unstructured data within databases. This approach also was followed in [10], which addresses information extraction in a similar type of data as we do, but applies simultaneously named entity recognition and entity resolution (the recognized names are resolved in a data set of known entities). The contribution of this work for advancing in NER techniques in this type of data is somewhat limited, since it only addressed the recognition of entities that are present in the source data set. Similarly to our experience, this work also reports difficulties with the NER solutions for natural language text (although only one tool was evaluated [11]). However, this approach differs significantly from ours. In order to improve the NER results, this approach was based on the evidence provided by structured data about the entities to be recognized, and the recognition model is based on manually crafted parsing rules created by a domain expert.

Although not addressing the same type of data as we do, we can find approaches used in other contexts that also perform NER in text containing little or no lexical evidence. In [12], an approach is described for performing information extraction on a particular kind of unstructured and ungrammatical text posted on the World Wide Web, such as item auction posts or online classifieds. The aim of this approach however is to extract a structured data record from each post, assuming that each post contains multiple attributes' values of one entity, making the approach not applicable to our scenario.

Other works, addressing NER in text without lexical evidence, focused on search engine queries [13][14]. In this work the problem is defined assuming the existence of one main entity per query, and adopted a specific technique for such cases, based on query logs [13] or user sessions [14] and topic models. We find the topic model approach to be not generally applicable for NER in to the data we are studying, since it assumes the existence of only one main entity per data element value.

3 Approach

We aimed at developing a general NER approach which could be systematically applied to any poorly structured data set. This section starts by presenting our analysis of the NER problem in structured data, and the general design decisions behind our approach. The description of the approach follows, and finalizes with the description of relevant implementation details.

3.1 Analysis

From our analysis of named entities found in structured data sets, we can highlight the following points:

- Availability of lexical evidence varies in many cases. In some data elements we found grammatically well-structured text, in other elements we found short sentences, containing very limited lexical evidence, or plain expression with practically non-existing lexical evidence. We also observed that in some cases, analysis of the same field across several records, revealed a mix of all cases.
- Instead of lexical evidence, we observed that, in some cases, textual patterns are often available and could be explored as evidence for NER. For example, punctuation marks play an important role, but its use may differ from how they are used in natural language text.
- These data elements are typically modeled with general semantics. The semantics associated with each element influences the type of named entities found in the actual records. Therefore, we observed different probability distributions for each entity type across data elements.
- One of the major sources of evidence is the actual name of the entities. Each entity type presents names with different words and lengths, and also with different degrees of ambiguity with other words and entity types.

From this analysis we believe that a generic approach must be highly adaptable, not only to the data set under consideration but also to each data element. Text found in each element across the whole data set is likely to be associated with particular patterns and degrees of available lexical evidence.

On a more generic level, the approach should have a strong focus on the disambiguation of the names between the supported entity types, and be able to disambiguate between entity names and other nouns/words.

3.2 Entity Types

We studied the three entity types on which most NER research has been focused, and which are commonly known as *enamex* [15]: person, location and organization. In addressing these three entity types, we wanted to design an approach that was not limited to a set of known entity names, but could recognize any named entity of the supported entity types, as usually done in NER in grammatically well-structured text.

As mentioned in the previous section, in structured data the characteristics of the names of persons, organizations, and places are a strong evidence for recognizing the named entities and determining their entity type. Therefore, in order to allow the predictive model to use the likelihood of a token being part of a named entity, we have collected name usage statistics from comprehensive data sets of persons, organizations and locations.

Person and organization name statistics were extracted from VIAF - Virtual International Authority File [16]. VIAF is a joint effort of several national libraries from

all continents towards a consolidated data set gathered for many years about the creators of the bibliographic resources held at these libraries.

Location name statistics were extracted from Geonames [17], a geographic ontology that covers all countries and contains over eight million locations.

A description of how the statistics were extracted, and used in the predictive model, is presented in Section 3.4.

3.3 Predictive Model

Our analysis suggested that a flexible approach with the capacity to adapt to the data set would be necessary for performing NER in structured data. This suggested the application of a machine learned model, an option also supported by the literature review of state of the art NER approaches.

The NER problem can be formulated as follows. Given a text string x and a set of entity types Y , where x consists of a sequence of tokens $x_1 \dots x_n$, and each token is a word or a punctuation mark, the entity recognition task consists in segmenting x into a sequence s of non-overlapping segments $s_1 \dots s_p$ where each segment s_j is associated with a $y_j \in Y$, and a start position t_j , and an end position u_j (for notation readability purposes we assume Y to also contain a *non_entity* type). All segments of s are non-overlapping and fully encompass all tokens of x , therefore for all x_i exists one and only one s_j that satisfies $s_{ij} <= i$ and $s_{ij} >= i$.

We use as a basis the conditional models of conditional random fields (CRF) [18]. CRFs define a conditional probability $p(y|x)$ over label sequences given a particular observation sequence x . These models allow the labelling of an arbitrary sequence x' by choosing the label sequence y' that maximizes the conditional probability $p(y'|x')$. The conditional nature of these models allows arbitrary characteristics of the sequences to be captured by the model, without requiring previous knowledge, by the modeller, about how these characteristics are related [19].

In order to find the sequence s that correctly recognizes the entity names from the observation sequence x , evidence is extracted or calculated. This evidence consists in a set of features which capture those characteristics of the empirical distribution of the data that support the recognition of names. Many different methods have been used to calculate and use features in a combined manner. Features may be calculated from natural language processing of the source text, by rules defined by domain experts, by lookups in lists of entity names and ontologies, from syntactical characteristics of the tokens, etc. The following section presents the set of features that we defined for our particular predictive model.

3.4 Features

Several features were defined to give the predictive model the capability to capture distinct aspects of the text, such as locating potential names, disambiguate between entity types and other words, or detecting textual patterns from syntactical and lexical evidence. This section presents the definition of these features.

A set of features were defined to provide the predictive model with some evidence for locating potential names of entities in the text. These features were created based on data or statistics taken from the comprehensive listings of names described in Section 3.2. Each entity type has different characteristics in the way entities are named, so we defined the features in different ways for each entity type.

The features for person names explore how frequent a word was found in person names, making a distinction between first names, surnames and names that appear in lowercase. Let F denote a bag built from all first names found in VIAF, and let S denote a bag built from all surnames found in VIAF, and let C be a bag built from all names found non-capitalized in VIAF. We define the following real valued features:

$$personFirstName(x, i) = \log \left(1 + \frac{F_{\#x_i}}{\left(\frac{\sum_{j=0}^{\#F} F_{\#j}}{\#F} \right)} \right)$$

$$personSurname(x, i) = \log \left(1 + \frac{S_{\#x_i}}{\left(\frac{\sum_{j=0}^{\#S} S_{\#j}}{\#S} \right)} \right)$$

$$personNoCapitalsName(x, i) = \log \left(1 + \frac{C_{\#x_i}}{\left(\frac{\sum_{j=0}^{\#C} C_{\#j}}{\#C} \right)} \right)$$

For organizations, only one feature was defined. Let C be a bag built from all words and punctuation marks found in the names of organizations in VIAF, we define the following real valued feature:

$$organizationName(x, i) = \log \left(1 + \frac{C_{\#x_i}}{\left(\frac{\sum_{j=0}^{\#C} C_{\#j}}{\#C} \right)} \right)$$

For places, the diversity of the names makes the frequency of use of the words not effective, so one feature was defined, using the type of geographic entity and the highest population known for a place on whose name the word appears in. Let C denote a bag built from all tokens found in the names of continents and countries. Similarly let D , E , F and G denote bags built from all tokens found in the names of cities, administrative divisions or islands, natural geographic entities, and other geographic features, respectively. Also let $population(t) \mapsto \mathbb{N}$ denote a function that returns the maximum population found in a location name with token t . We defined the following real valued feature:

$$locationName(x, i) = \begin{cases} 1, & \text{if } x_i \in C \\ \frac{\min(100000, population(x_i))}{100000}, & \text{if } x_i \in D \\ 0.7, & \text{if } x_i \in E \\ 0.6, & \text{if } x_i \in F \\ 0.1, & \text{if } x_i \in G \\ 0, & \text{otherwise} \end{cases}$$

Some features are based on data extracted from the WordNet [20] of the language matching the language of the source text, which in the case we studied was English. These features provide evidence to disambiguate between named entities of the target types and other words.

With the aim to disambiguate between proper nouns referring to other entity types, and proper nouns referring to persons, locations and organizations, we define the feature $properNoun(x, i) \mapsto \{0,1\}$. Let P denote the set of all variants in synsets which have a part-of-speech value of *proper noun*, and let G, H, I, J, K, L denote the sets of variants in synsets which are hyponyms, either directly or transitively, of one of the synsets² *geographic area#noun#1*, *landmass#noun#1*, *district#noun#1*, *body of water#noun#1*, *organization#noun#5*, and *person#noun#1*, respectively. The feature is defined as:

$$properNoun(x, i) = \begin{cases} 1, & \text{if } x_i \in P \setminus (G \cup H \cup I \cup J \cup K \cup L) \\ 0, & \text{otherwise} \end{cases}$$

We also use the Wordnet to capture the possible part-of-speech of some tokens. We defined the feature $posNoun(x, i) \mapsto \{0,1\}$, which indicates if the token exists in a synset with part-of-speech *noun*. Let A denote the set of variants in synsets which have a part-of-speech value of *noun*, we define the feature as:

$$posNoun(x, i) = \begin{cases} 1, & \text{if } x_i \in A \\ 0, & \text{otherwise} \end{cases}$$

Similar features were defined for other parts-of-speech: $posVerb(x, i) \mapsto \{0,1\}$, $posAdjective(x, i) \mapsto \{0,1\}$, $posAdverb(x, i) \mapsto \{0,1\}$, and $posPreposition(x, i) \mapsto \{0,1\}$.

We also defined features to capture syntactical characteristics of the text and the tokens. The features $startOfElement(x, i) \mapsto \{0,1\}$ and $endOfElement(x, i) \mapsto \{0,1\}$ indicate if token x_i is at the start or at the end of the value of the data element. The case of the token is captured through the features $isCapitalized(x, i) \mapsto \{0,1\}$ and $isFullCaps(x, i) \mapsto \{0,1\}$, which indicate if the token is a word and contains the first letter in uppercase, or all letters in uppercase, respectively. The token's character length is captured by the feature $tokenLength(x, i) \mapsto \mathbb{N}$.

The tokens are also used in a nominal feature $token(x, i) \mapsto T$, where T denotes the set of tokens built from the three preceding tokens, and the two following tokens, of every named entity found in the training data:

² To refer to Princeton WordNet synsets, we use the notation $w\#p\#i$ where i corresponds to the i -th sense of a literal w with part of speech p .

$$token(x, i) = \begin{cases} x_i, & \text{if } x_i \in T \\ \emptyset, & \text{otherwise} \end{cases}$$

Capitalization statistics of words in the data set are extracted and used in a feature. Let C denote the bag of capitalized words in the data set, and let D denote a bag of the non-capitalized words in data set, we define the following real valued feature:

$$capitalizedFrequency(x, i) = \log\left(\frac{C_{\#x_i}}{(1 + D_{\#x_i})}\right)$$

Since typically each data element will have values with different characteristics, a feature is necessary to capture the data element where the text is contained. We defined the feature $dataElement(x, i) \mapsto D$, where D denotes the set of data element identifiers of the data model (for example, in data encoded in XML, these identifiers consist of the xml element's namespace and element's name).

Additional features are defined in similar way, but they refer to the three previous tokens and the two following tokens, instead of the current one.

3.5 Implementation Details

In this section we provide some relevant details of the implementation of our approach, in particular we address text tokenization and the CRF implementation and configuration.

Tokenization of the text inside the data elements is performed only at word level. No sentence or paragraph tokenization is performed, since in many cases well-structured sentences are not present in the data and the results of sentence and paragraph tokenization could invalidate the detection of patterns in the data.

Word tokenization is performed in a language independent way. We also justify this option to avoid the breaking of patterns in the data, in particular in cases where punctuation is used in the data with different meanings than it is has in natural language text. We have applied the word breaking rules of UNICODE [22].

The CRF implementation used was provided by the Java implementation in the MALLET - Machine Learning for Language Toolkit [21]. The CRF was configured to use the three previous states in the sequence in the labelling of the sequence, and was trained using an objective function for CRFs that consists in the label likelihood plus a Gaussian prior on parameters.

4 Evaluation

The evaluation of our approach was performed in the data sets from Europeana³, which consist in descriptions of digital objects of cultural interest. This data set follows a data model using mainly Dublin Core elements, and named entities appear in

³ <http://www.europeana.eu/>

data elements for titles, textual descriptions, tables of contents, subjects, authors and publication.

The data set contains records originating from several European providers from the cultural sector, such as libraries, museums and archives. Several European languages are present, even within the description of the same object, for example when the object being described is of a different language than the one used to create its description.

Providers from where this data originates follow different practices for describing the digital objects, which causes the existence of highly heterogeneous data. Lexical evidence is very limited in this data set, so it provides a good scenario for the evaluation of the evidence made available by the structure and textual patterns of the data.

This section describes the experimental setup and its results. It will follow with the description of the data set used for evaluation, and then describe the evaluation procedure. Results of the evaluation are presented afterwards, and it finalizes with the results of the evaluation of individual features.

4.1 Evaluation Data Set

An evaluation of our approach was performed on a selected collection of metadata records from Europeana. This collection was created by randomly selecting records in the English language. The selection process was done in two steps: first, all records in the English language were selected from all Europeana data providers; and second, a random selection of records was performed, balancing the number of records chosen across different providers.

In total, the evaluation data set⁴ consisted in 120 records containing in its elements 584 references to persons, 457 to locations and 153 to organizations, as shown in Table 1.

Table 1. Data elements studied in the data set and total annotated named entities

Data element	Element definition ⁵	Pers.	Locat.	Organiz.
Title	A name given to the resource.	142	86	26
creator / contributor	An entity primarily responsible for making the resource / An entity responsible for making contributions to the resource.	156	0	27
Subject	The topic of the resource.	60	136	16
Coverage	The spatial or temporal topic of the resource, the spatial applicability of the resource, or the jurisdiction under which the resource is relevant.	0	79	0
Description	An account of the resource.	199	75	33
table of contents	A list of subunits of the resource.	10	29	3
Publisher	An entity responsible for making the resource available.	17	52	48
	Total:	584	457	153

⁴ The data set is available for research use at <http://web.ist.utl.pt/~nuno.freire/ner/>

⁵ Element definitions were taken from the Dublin Core Metadata Terms.

The evaluation data set was manually annotated. In very few cases, the manual annotation was uncertain, because the data records may not contain enough information to support a correct annotation. For example, some sentences with named entities were too small and no other information was available in the record to support a decision on the classification of the named entities to their entity type. Named entities were annotated with their *enamex* type. If the annotator was unsure of the *enamex* type of a named entity, he would annotate it as *unknown*. These annotations were not considered for the evaluation of the results, and any recognition made in these entities was discarded.

4.2 Evaluation Procedure

The accuracy of the results of our approach was compared with that of other two approaches: one was the implementation of a conditional maximum entropy model [25], taken from the OpenNLP package; the other was based on conditional random fields [26], from the Stanford Named Entity Recognizer (Stanford NER). For both cases, we used the respective predictive models trained on the CoNLL 2003 English training data [27]. However, since in all tests the Stanford NER performed better than OpenNlp, for readability, we only present the results of Stanford NER as our baseline for comparison.

Since our predictive model was trained on the evaluation data set, all the measurements were obtained using cross-validation tests, which has been widely accepted as a reliable method for calculating generalization accuracy [24]. Cross-validation involves partitioning the evaluation data set into complementary subsets, testing on one subset, while training on the remaining subset. Ten-fold cross-validation was performed using different partitions, and the validation results were averaged over the ten runs.

As the NER evaluation method, we have used the *exact-match* method. This method has been used in several named entity recognition evaluation tasks [23] [27]. In the *exact-match* method, an entity is only considered correctly recognized if it is exactly located as in the manual annotation. Recognition of only part of the name, or with words that are not part of the name, is not considered correct. In combination with the *exact-match* method, we used the metrics of precision⁶, recall⁷ and F₁-measure⁸.

To evaluate on the balance between results in precision and recall, we have taken measures at several minimum confidence thresholds. For both our approach and the baseline, we only consider a named entity recognized if the joint probability of the corresponding segment is equal or above the minimum confidence threshold.

4.3 Results

The overall results of the evaluation of all entity types are presented in Fig. 2, and the results of each entity type are presented in Fig. 1. The results of our approach were

⁶ The percentage of correctly identified named entities in all named entities found.

⁷ The percentage of named entities found compared to all existing named entities.

⁸ The weighted harmonic mean of precision and recall (equal weights for recall and precision).

higher for all entity types, metrics and confidence levels. The differences between our approach and the baseline were statistically significant with $P>0.001$ for all measurements except for the entity type location where, in the lowest confidence threshold, we obtained $P>0.01$ on the three metrics.

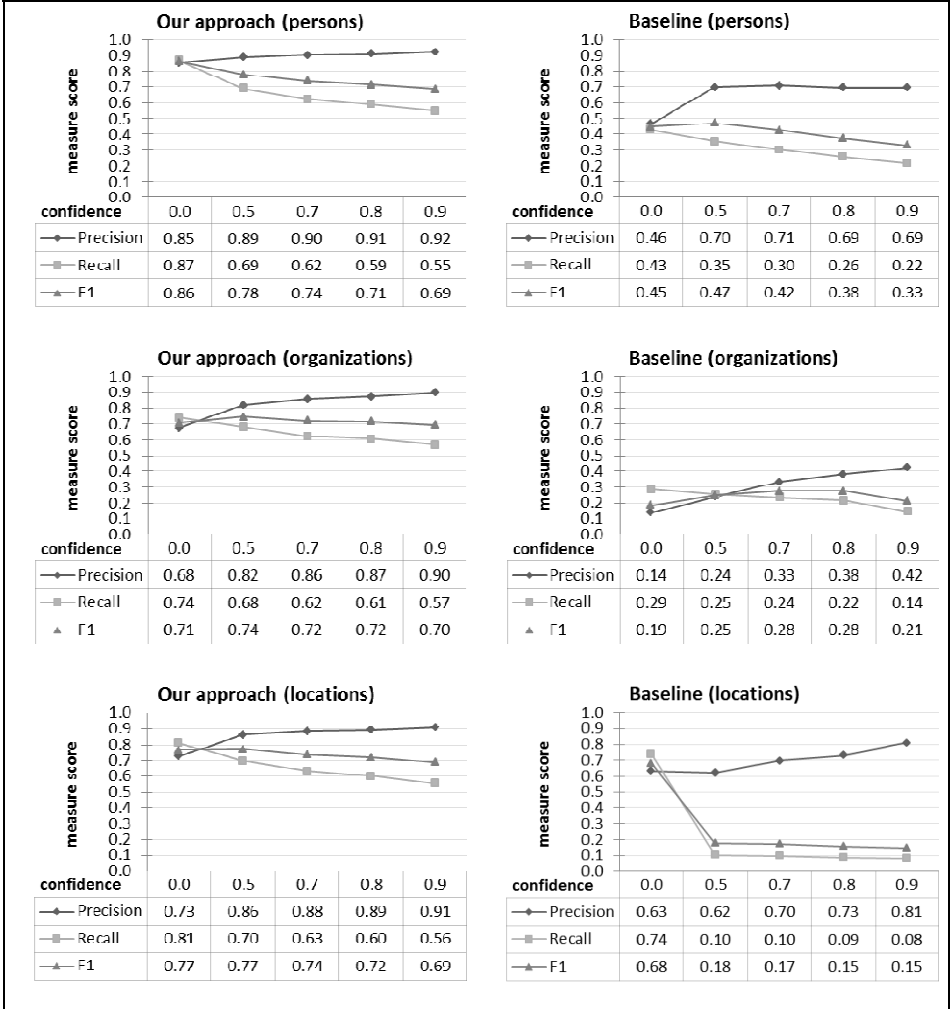


Fig. 1. Precision, recall and F₁ results of the three *enamex* entity types measured on the evaluation data set

Both Stanford NER and our approach are based on CRFs. Although the implementation of CRFs used was not the same, and other differences exist on how CRFs are used, we believe that the difference in the results obtained with both approaches is due to the different features used, therefore supporting our initial hypothesis that the semantic context of the data structure, and non-lexical features, could support NER.

An interesting result can be observed at the lowest confidence threshold result for the entity location, where Stanford NER was actually able to achieve a F_1 of 0.68, but the probability given by the CRF predictive model was close to zero for more than 70% of the recognized named entities. This observation suggests that the lack of lexical evidence had a major impact in its results.

Results of both approaches generally showed lowest values for recall than for precision. In our approach overall recall ranged from 0.55 to 0.82 while overall precision ranged from 0.77 to 0.91. Given the importance of the features based on the names of entities, as show in the next section, we believe that the lower recall is mainly caused by names that had no presence in the entity names data sets. However, we were not able to empirically support this conclusion.

Our approach was able to achieve a high precision of 0.91 at 0.55 recall, or reach a recall of 0.82 at 0.77 precision. We believe these values reached levels high enough to support a wide range of information extraction applications, which may have different requirements for recall or precision.

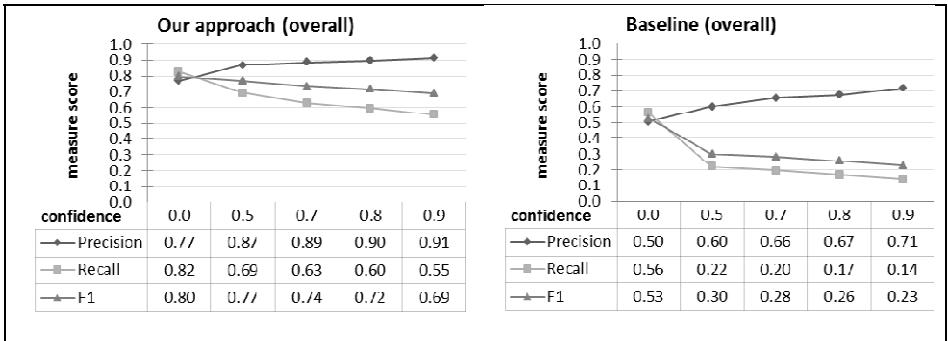


Fig. 2. Precision, recall and F_1 results of all entity types measured on the evaluation data set

4.4 Feature Evaluation

In order to evaluate the contribution of each feature for the quality of the NER results, we have performed a feature selection evaluation based on the wrapper methodology [28]. This method employs cross-validation using the actual target learning algorithm to estimate the accuracy of subsets of features.

To conduct this evaluation, we have grouped related features (for example, all features related with part-of-speech were considered one group), and performed an exhaustive evaluation for all combinations of groups of features.

In total, we formed 10 groups of features and tested all combinations of 7 groups. Each feature combination was evaluated by a 10-fold cross-validation test and the best performing feature combination, measured by the F_1 , of each fold was noted.

Table 2 summarizes the results, by showing how often each feature was present in the best performing combination of the 10 folds, for all entity types, and for each type individually. Since the features related with the names of the entities were essential

for the overall results, on the evaluation on the individual entity types, we always used combinations including these three groups of features, so that the results could be more easily compared and analyzed.

Table 2. Results of the evaluation of the features

Feature groups	Included in best combination			
	all types	persons	locations	organizations
<i>personFirstName(x, i)</i> <i>personSurname(x, i)</i> <i>personNoCapitalsName(x, i)</i>	100%	100%	100%	100%
<i>organizationName(x, i)</i>	100%	100%	100%	100%
<i>locationName(x, i)</i>	100%	100%	100%	100%
<i>token(x, i)</i> <i>startOfElement(x, i)</i> <i>endOfElement(x, i)</i>	90%	50%	70%	70%
<i>isCapitalized(x, i)</i> <i>isUppercased(x, i)</i>	80%	50%	100%	60%
<i>posNoun(x, i)</i> <i>posVerb(x, i)</i> <i>posAdjective(x, i)</i> <i>posAdverb(x, i)</i> <i>posProperNoun(x, i)</i> <i>posPreposition(x, i)</i>	70%	60%	70%	50%
<i>properNoun(x, i)</i>	60%	70%	50%	50%
<i>tokenLength(x, i)</i>	60%	60%	30%	50%
<i>dataElement(x, i)</i>	20%	60%	10%	20%
<i>capitalizedFrequency(x, i)</i>	20%	50%	70%	80%

All features contributed to the best performing combination, for all entity types, in at least two of the cross-validation folds. The features which were used the least for the best overall results, *dataElement(x, i)* and *capitalizedFrequency(x, i)*, were often used when evaluated on the results of the individual entity types. Therefore we believe that all features should be used when applying this approach to other data sets.

We can also observe that the features that detected the names of the entities were always used in the overall results. And, in addition, the features *token(x, i)*, *startOfElement(x, i)*, *endOfElement(x, i)*, *isCapitalized(x, i)*, and *isUppercased(x, i)* were used very often. This seems to indicate that textual patterns were very relevant for providing evidence for NER.

In the results of the feature *dataElement(x, i)*, it is worth noting that it was used only in 10% or 20% of the folds in the overall results for locations and organizations, but for persons it was used in 60% of the folds. This indicates that the textual patterns where persons are referenced were distinct across data elements, while for the other entity types the patterns were more uniform across data elements. Our analysis

pointed that, in the data elements for creators and contributors, the names for persons often appeared in inverse order (that is, *surname, first_names*), while in the other elements they appeared in direct order (that is, *first_names surname*). We therefore conclude that the semantic context given by the data structure is generally not required to allow the recognition of the entities, but in some cases, it can provide importance evidence for the predictive model.

5 Conclusion and Future Work

We presented an approach for the task of named entity recognition in structured data containing free text as the values of its elements. This approach is based on the extraction of features from the text, which allows the predictive model to operate with more independent of lexical evidence than named entity recognition systems developed for grammatically well-formed text.

Our approach was able to achieve a maximum precision of 0.91 at 0.55 recall, and a maximum recall of 0.82 at 0.77 precision. The achieved results were significantly higher than those obtained with the baseline. We believe this level of quality in named entity recognition allows the use of this approach to support a wide range of information extraction applications in digital library metadata.

Although we have specifically studied metadata from the cultural heritage sector, we believe our approach has general applicability to any poorly structured data model.

In future work we will explore the use of ontologies for creating features to improve the recognition of named entities. We will also address the resolution of the recognized named entities in linked data contexts and ontologies.

References

1. Seth, G.: Unstructured Data and the 80 Percent Rule: Investigating the 80%. Technical report, Clarabridge Bridgepoints (2008)
2. Shilakes, C., Tylman, J.: Enterprise Information Portals. Merrill Lynch Report (1998)
3. Sarawagi, S.: Information Extraction. Found. Trends Databases 1, 261–377 (2008)
4. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. *Linguisticae Investigationes* 30 (2007)
5. McCallum, A., Freitag, D., Pereira, F.: Maximum entropy Markov models for information extraction and segmentation. In: International Conference on Machine Learning (2000)
6. Martins, B., Borbinha, J., Pedrosa, G., Gil, J., Freire, N.: Geographically-aware information retrieval for collections of digitized historical maps. In: 4th ACM Workshop on Geographical Information Retrieval (2007)
7. Freire, N., Borbinha, J., Calado, P., Martins, B.: A Metadata Geoparsing System for Place Name Recognition and Resolution in Metadata Records. In: ACM/IEEE Joint Conference on Digital Libraries (2011)
8. Sporleder, C.: Natural Language Processing for Cultural Heritage Domains. *Language and Linguistics Compass* 4(9), 750–768 (2010)
9. King, P., Poulouvassilis, A.: Enhancing database technology to better manage and exploit Partially Structured Data. Technical report, University of London (2000)

10. Williams, D.: Combining Data Integration and Information Extraction. PhD thesis, University of London (2008)
11. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M., Saggion, H., Petrak, J., Li, Y., Peters, W.: Text Processing with GATE (Version 6). University of Sheffield Department of Computer Science (2011) ISBN 978-0956599315
12. Michelson, M., Knoblock, C.: Creating Relational Data from Unstructured and Ungrammatical Data Sources. *Journal of Artificial Intelligence Research* 31, 543–590 (2008)
13. Guo, J., Xu, G., Cheng, X., Li, H.: Named Entity Recognition in Query. In: 32nd Annual ACM SIGIR Conference (2009)
14. Du, J., Zhang, Z., Yan, J., Cui, Y., Chen, Z.: Using Search Session Context for Named Entity Recognition in Query. In: 33rd Annual ACM SIGIR Conference (2010)
15. Grishman, R., Sundheim, B.: Message Understanding Conference - 6: A Brief History. In: Proc. International Conference on Computational Linguistics (1996)
16. Bennett, R., Hengel-Dittrich, C., O’Neill, E., Tillett, B.B.: VIAF (Virtual International Authority File): Linking Die Deutsche Bibliothek and Library of Congress Name Authority Files. In: 72nd IFLA General Conference and Council (2006)
17. Vatant, B., Wick, M.: Geonames Ontology (2006), <http://www.geonames.org/ontology/>
18. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282–289. Morgan Kaufmann Publishers Inc. (2001)
19. Wallach, H.: Conditional Random Fields: An Introduction. Technical Report MS-CIS-04-21. Department of Computer and Information Science, University of Pennsylvania (2004), http://www.cs.umass.edu/~wallach/technical_reports/wallach04conditional.pdf
20. Miller, G.A., Beckwith, R., Fellbaum, C.D., Gross, D., Miller, K.: WordNet: An online lexical database. *Int. J. Lexicograph.* 3(4), 235–244 (1990)
21. McCallum, A.: MALLET: A Machine Learning for Language Toolkit (2002), <http://mallet.cs.umass.edu>
22. The Unicode Consortium: Unicode Text Segmentation (2010), <http://www.unicode.org/reports/tr29/>
23. Sekine, S., Isahara, H.: IREX: IR and IE Evaluation project in Japanese. In: Proc. Conference on Language Resources and Evaluation (2000)
24. Michie, D., Spiegelhalter, D.J., Taylor, C.C.: Machine learning, neural and statistical classification. Prentice Hall, Englewood Cliffs (1994)
25. Goodman, J.: Sequential Conditional Generalized Iterative Scaling. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pp. 9–16 (2002)
26. Finkel, J.R., Grenager, T., Manning, C.: Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In: 43rd Annual Meeting of the Association for Computational Linguistics (2005)
27. Sang, T.K., Erik, F., De, F.: Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In: Conf. on Natural Language Learning (2003)
28. Kohavi, R., John, G.: Wrappers for feature selection. *Artificial Intelligence* 97(1-2), 273–324 (1997)

Supporting Linked Data Production for Cultural Heritage Institutes: The Amsterdam Museum Case Study

Victor de Boer¹, Jan Wielemaker¹, Judith van Gent², Michiel Hildebrand¹,
Antoine Isaac¹, Jacco van Ossenbruggen¹, and Guus Schreiber¹

¹ Department of Computer Science, VU University, Amsterdam, The Netherlands
{v.de.boer, j.wielemaker, a.isaac,
m.hildebrand, j.r.van.ossenbruggen, guus.schreiber}@vu.nl

² Amsterdam Museum, Amsterdam, The Netherlands
j.vangent@amsterdammuseum.nl

Abstract. Within the cultural heritage field, proprietary metadata and vocabularies are being transformed into public Linked Data. These efforts have mostly been at the level of large-scale aggregators such as Europeana where the original data is abstracted to a common format and schema. Although this approach ensures a level of consistency and interoperability, the richness of the original data is lost in the process. In this paper, we present a transparent and interactive methodology for ingesting, converting and linking cultural heritage metadata into Linked Data. The methodology is designed to maintain the richness and detail of the original metadata. We introduce the XMLRDF conversion tool and describe how it is integrated in the ClioPatria semantic web toolkit. The methodology and the tools have been validated by converting the Amsterdam Museum metadata to a Linked Data version. In this way, the Amsterdam Museum became the first ‘small’ cultural heritage institution with a node in the Linked Data cloud.

1 Introduction

Cultural heritage institutions such as museums, archives or libraries typically have large databases of metadata records describing the objects they curate as well as thesauri and other authority files used for these metadata fields. At the same time, in the Linked Data cloud a number of general datasets exist, such as GeoNames, VIAF or DBPedia. Importing the individual cultural heritage metadata into the Linked Data cloud and linking to these general datasets improves its reusability and integration.

While larger cultural heritage institutions such as the German National Library¹ or British National Library² have the resources to produce their own Linked Data, metadata from smaller institutions is currently only being added through large-scale aggregators. A prime example is Europeana, whose goals are to serve as an aggregator for cultural heritage institution data. This is to be achieved through a process of ingesting the metadata records, restructuring it to fit the Europeana Data Model and publishing it

¹ <http://permalink.gmane.org/gmane.culture.libraries.ngc4lib/7544>

² <http://www.bl.uk/bibliographic/datafree.html>

as Linked Data on Europeana servers. This approach ensures a level of consistency and interoperability between the datasets from different institutions. The automatic ingestion process, conversion into new dataformats and external hosting by the aggregator however creates the problem of a disconnect between the cultural heritage institute original metadata and the Linked Data version.

Rather than having Linked Data ingestion being done automatically by large aggregators, we present a methodology that is both *transparent* and *interactive*. The methodology covers data ingestion, conversion, alignment and Linked Data publication. It is highly modular with clearly recognizable data transformation steps, which can be evaluated and adapted based on these evaluations. This design allows the institute's collection managers, who are most knowledgeable about their own data, to perform or oversee the process themselves. We describe a stack of tools that allow collection managers to produce a Linked Data version of their metadata that maintains the richness of the original data including the institute-specific metadata classes and properties. By providing a mapping to a common schema interoperability is achieved. This has been previously investigated by Tordai et al. [9], of which this work is a continuation. We provide a partial validation of both these tools and the general methodology by using it to convert the metadata from the Amsterdam Museum to RDF and serving it as Linked Data.

2 Methodology Overview

To convert collection metadata into Linked Data, we here describe the general methodology. The input is the original collection metadata as provided by aggregators or individual cultural heritage institutions. The result of the workflow process is the collection metadata in semantic format (RDF). Links are established between vocabulary terms used in the collections.

Figure 1 shows the general workflow for the conversion and linking of the provided metadata. The methodology is built on the ClioPatria semantic server [11]. ClioPatria provides feedback to the user about intermediary or final RDF output in the form of an RDF browser and by providing statistics on the various RDF graphs. This feedback is crucial for the intended interactivity. The approach takes the form of a modular workflow, supported by two tools. Both the XMLRDF and Amalgame are packages of the ClioPatria semantic web toolkit. ClioPatria itself is based on SWI-Prolog and XML-RDF can therefore use its expressiveness for more complex conversions. ClioPatria and its packages are available from <http://ClioPatria.swi-prolog.org/>.

In the first step of this workflow, we ingest the XML into the ClioPatria environment. This can be either a static XML file with the metadata or the result of OAI harvesting operation. We give an example in the case study in Section 6.3. In the second step, the XML is converted to crude RDF format. This is done using the XMLRDF tool, which is documented in Section 3. This crude RDF is then rewritten in RDF adhering to the chosen metadata format, which is done using graph rewrite rules. These rules are executed by the XMLRDF tool to produce the final RDF representation of the collection metadata. An example of ingested XML, crude RDF and rewritten RDF is presented in Section 6.3 and Figure 3. Next, the user can provide an RDFS metadata schema which relates the produced classes and properties to the metadata schema of choice.

The XMLRDF tool provides support for this by presenting the user with a schema template based on the RDF data loaded. In Step 5, links are established between vocabulary concepts that are used in the collection metadata and other vocabularies. This is done using the Amalgame tool, which is documented in van Ossenburg et al. [7]. In Section 5, we describe Amalgame and its methodology insofar it is part of the more general conversion strategy.

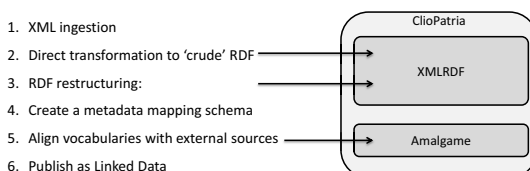


Fig. 1. General workflow for converting and linking metadata. The figure lists the various steps and relates them to either the XMLRDF or Amalgame tool or to the ClioPatria server itself.

The RDF data can be served as Linked Open Data using ClioPatria. The server performs HTTP content negotiation. If the HTTP request asks for ‘text/html’, the server responds with an HTML page showing the (object) metadata in human-readable form. When ‘application/rdf+xml’ is requested, the server responds by providing the Concise Bounded Description in RDF triples³. This adheres to the Linked Data requirements [1]. Additionally, a SPARQL endpoint is provided, where the data can be queried.

3 XMLRDF Conversion Tool

We here describe the XMLRDF tool, argue why it is specifically useful for converting cultural heritage data and provide a number of ‘recipes’ for converting such data into RDF. A more complete documentation is available [10].

When rewriting XML source data to an RDF datamodel, two extreme cases can be identified. In one case, the original metadata is forced into a target metamodel such as Dublin Core. This means that specific metadata values are copied to the new model, deleting other values. While this produces ‘clean’ data in the new metamodel, the original complexity and detail of the data may be lost. On the other end of the spectrum is a purely syntactic transformation to RDF, where each XML construct is converted to RDF. For datasets that are more or less ‘flat’ record structures, the produced RDF is usually of sufficient quality. However, cultural heritage datasets usually are inherently graph-like and will consist of more complex XML. They often include implicit links to vocabulary terms and other records, for which RDF provides a way to make these explicit. On the other hand, the XML often contains purely syntactical constructs such as some sub-trees used for meaningless grouping of elements that should not be maintained in the RDF version. In many cases, it is unclear whether a construct is meaningless or meaningful. For example, in most cases, XML element ordering is meaningless,

³ <http://www.w3.org/Submission/CBD/>

but for some archival data, the ordering is important and should be explicitly modeled in the target RDF.

The XMLRDF tool is designed to translate each of these syntactic artifacts into a proper semantic model where objects and properties are typed and semantically related to a semantic metamodel such as the widely used SKOS and Dublin Core vocabularies. It does so in two main steps, shown in Figure 1. First, a syntactic conversion of the source XML into crude RDF is made. The produced RDF is as close as possible to the source metadata. This step is described in Section 3.1. The RDF produced in this way can be re-structured and enriched in the second step, which is in turn subdivided into multiple sub-steps. We describe this in Section 3.2.

The interactivity in XMLRDF stems from the fact that individual rules, or combinations of rules, can be run independently of each other and the resulting intermediate RDF can be quickly inspected through the ClioPatria browser/visualization. This allows the user to write rules, evaluate the result and adapt the rule if necessary.

3.1 Step 2: Syntactic Conversion to RDF

The core idea behind converting ‘data-xml’ into RDF is that every complex XML element maps to a resource (often a blank node) and every atomic attribute maps to an attribute of this bnode. Such a translation gives a valid RDF document, which can be processed using XMLRDF’s graph-rewrite rules. There are a few places where we must be more subtle in the initial conversion: The `xml:lang` attribute is kept around and if we create an RDF literal, it is used to create a literal in the current language and `xmlns` declarations are ignored (they make the XML name space declarations available to the application, but the namespaces are already processed by the XML parser).

Second, we may wish to map some of our properties into `rdfs:XMLLiteral` or RDF dataTypes. In particular the first must be done in the first pass to avoid all the complexities of turning the RDF back into XML. In the conversion configuration, the user can specify properties and classes that are to be preserved and mapped to either an `XMLLiteral` or a typed RDF Literal.

The initial XML to RDF mapper uses the XML attribute and tag-names for creating RDF properties. A namespace prefix is optionally added to each XML name to create a fully qualified URI. It also ‘restyles’ XML identifiers: notably identifiers that contain a dot (.) are hard to process using Turtle. In the top part of Figure 3, we show the an example XML snippet and its crude RDF form.

3.2 Step 3: Enriching the RDF

The RDF produced in the steps is generally rather crude and the graph can be rewritten using rules in the XMLRDF rewrite language. The rewrite language is a production-rule system, where the syntax is modelled after Constraint Handling Rules, a committed-choice language for constraint programming [3] and the triple notation is based on Turtle/SPARQL. The transformation process for actual data however can be complicated. For these cases the rule-system allow mixing rules with arbitrary Prolog code, providing an unconstrained transformation system. We provide a (dynamically extended) library

```

title_nl @@
{ S, am:title, TitleNL }
<=>
{ S, am:title, TitleNL@nl}.

content_person @@
{PersonURI, am:name, CP},
{PersonURI, rdf:type, am:'Person'}\
{S, am:contentPersonName, CP}
<=>
{S, am:contentPersonName, PersonURI}.

use_to_altlabel @@
{UseUri, am:term, UseTerm},
{S, am:term, AltLab}\
{S, am:use, UseTerm}
<=>
{UseUri, skos:altLabel, AltLab@nl}.

dimensions @@
{ _S, am:dimension, B},
{ B, am:dimensionValue, literal(Val)},
{ B, am:dimensionUnit, literal(Unit)}?,
{ B, am:dimensionType, literal(Type)}?,
{ B, am:dimensionPrecision, literal(Prec)}?,
{ B, am:dimensionPart, literal(Part)}?,
{ B, am:dimensionNotes, literal(Notes)}?
==>
rdf_is_bnode(B),
concat_m([Type,Val,Unit,Prec,Part,Notes], ConcatVal),
{B, rdfs:label, literal(ConcatVal)}.

assign_uris @@
{S, am:preref, literal(Preref)}\
{ S } <=>
literal_to_id(['proxy-', Preref], am, URI),
{ URI }.

clean_empty @@
{ _, _, "" } <=> true.

```

Fig. 2. Examples of XMLRDF rules used in the Amsterdam Museum conversion

of Prolog routines for typical conversion tasks. In some cases these will not satisfy, in which case expertise in programming Prolog becomes necessary. We here give an overview of the XMLRDF rewriting rules and provide a number of RDF transformation recipes. There are 3 types of production rules:

1. *Propagation rules* add triples
2. *Simplication rules* delete triples and add new triples.
3. *Simpagation rules* are in between. They match triples, delete triples and add triples,

The overall syntax for the three rule-types is (in the order above):

```

<name>? @@ <triple>* ==> <guard>? , <triple>*.
<name>? @@ <triple>* <=> <guard>? , <triple>*.
<name>? @@ <triple>* \ <triple>* <=> <guard>? , <triple>*.

```

Here, <guard> is an arbitrary Prolog goal. <triple> is a triple in a Turtle-like, but Prolog native, syntax: { <subject>, <predicate>, <object> }. Any of these fields may contain a variable, written as a Prolog variable: an uppercase letter followed by zero or more letters, digits or the underscore. Resources are either fully (single-)quoted Prolog atoms (E.g. 'http://example.com/myResource', or terms of the form <prefix>:<local>, e.g., vra:title or ulan:'Person' (note the quotes to avoid Prolog interpretation as a variable). Figure 2 shows six rules that were used in the Amsterdam Museum example, the rules relate to recipes that will be discussed in the following sections. The rules can be run all consecutively or they can be executed one at a time allowing the user to evaluate the intermediary results. In the bottom part of Figure 3 we also show the RDF graph resulting from applying these rules to the example crude RDF.

Fixing the Node-Structure. In some cases, we might want to add triples by concatenating multiple values. The **dimensions** rule in Figure 2 is an example of this, where we add a concatenation of dimension values as an additional triple, using a new predicate. Since we do not delete the original metadata, some data duplication occurs. In addition to this, some literal fields need to be rewritten, sometimes to (multiple) new literals and sometimes to a named or bnode instance.

The simplification rules can be used to map the record-based structure to the desired structure. An example is the **use_to_altlabel** rule in Figure 2, which converts the ISO term-based thesaurus constructs to the SKOS variant. This rule takes the alternative term and re-asserts it as the `skos:altLabel` for the main concept.

Another use for the simplification rules is to delete triples, by having an empty action part (Prolog ‘true’), as shown by the **clean_empty** rule in Figure 2. This can be used to delete triples with empty literals or otherwise obsolete triples.

Some blank nodes provide no semantic organization and can be removed, relating its properties to the parent node. At other places, intermediate instances must be created (as blank nodes or named instances).

Re-establish Internal Links. The crude RDF often contains literals where it should have references to other RDF instances. Some properties represent links to other works in the collection. The property value is typically a literal representing a unique identifier to the target object such as the collection identifier or a database key. This step replaces the predicate-value with an actual link to the target resource. The rewrite rules can use the RDF background knowledge to determine the correct URI.

Re-establish External Links. This step re-establishes links from external resources such as vocabularies which we know to be used during the annotation. In this step we only make mapping for which we are absolutely sure. I.e., if there is any ambiguity, we maintain the value as a blank node created in the previous step. An example of such a rule is the **content_person** rule shown in Figure 2.

Assign URIs to Blank Nodes Where Applicable. Any blank node we may wish to link to from the outside world needs to be given a real URI. The record-URIs are typically created from the collection-identifier. For other blank nodes, we look for distinguishing (short) literals. The construct $\{X\}$ can be used on the condition and action side of a rule. If used, there must be exactly one such construct, one for the resource to be deleted and one for the resource to be added. All resources for which the condition matches are renamed. The **assign_uris** rule in Figure 2 is an example of this. The $\{S\}$ binds the (blank node) identifier to be renamed. The Prolog guard generates a URI (see below) which replaces all occurrences of the resource.

Utility Predicates. The rewriting process is often guided by a guard which is, as already mentioned, an arbitrary Prolog goal. Because translation of repositories shares a lot of common tasks, we developed a library for these. An example of such a utility predicate is the `literal_to_id` predicate, which generates an URI from a literal by

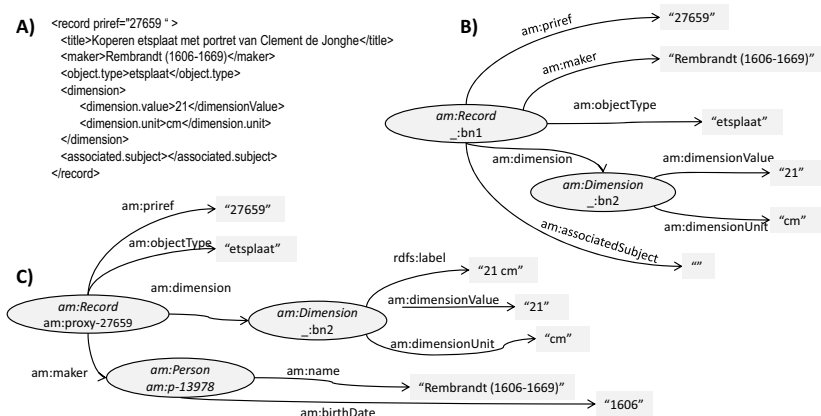


Fig. 3. Example of the different steps of XMLRDF. A) shows an XML sample snippet describing a single record is shown. B) is the result of the direct conversion to crude RDF is displayed. C) shows the graph after the rules from Figure 2 have been applied. In that final graph, the URI for the creator is used instead of the literal; a concatenated dimension label is added to the blank node; the empty ‘associated subject’ triple is removed and the record has a proxy-based URI.

mapping all characters that are not allowed in a (Turtle) identifier to `_`, as shown in the `assign_uris` rule in Figure 2.

4 Step 4: Mapping to Interoperability Layer

It is advised to maintain the original property- and type-names (classes) in the RDF because this allows to reason about possible subtle differences between the source-specific properties and properties that come from generic schemas such as Dublin Core. E.g., a creator as listed for a work in a museum for architecture is typically an architect and the work in the museum is some form of reproduction on the real physical object. If we had replaced the original creator property by `dcterms:creator`, this information is lost. A second reason to maintain the original property- and type-names makes it much easier to relate the RDF to the original collection data. One of the advantages of this is that it becomes easier to reuse the result of semantic enrichment in the original data-source. This implies that the converted data is normally accompanied by a schema that lists the properties and types in the data and relates them using `rdfs:subPropertyOf` or `rdfs:subClassOf` to one or more generic schemas (e.g., Dublin Core). ClioPatria provides a facility to generate a schema for a graph from the actual data, which can be used as a starting point. An example is shown in Figure 4.

ClioPatria supports RDFS entailment and any application that is built on top of its infrastructure is able to exploit the subproperty mappings that have been added in the metadata schema. In this way, the EDM datasets are integrated.

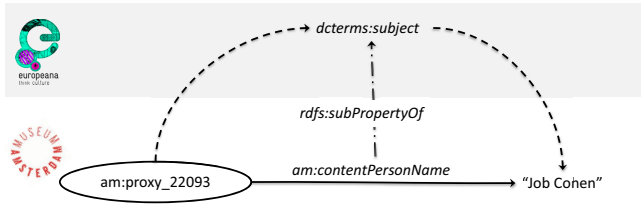


Fig. 4. RDF fragment showing how metadata mapping ensures interoperability. The bottom part of the figure shows an example triple relating an object to the name of a depicted person. Dublin Core (the metadata standard used in Europeana for object descriptions) only has a single notion of the subject of a work. By mapping the specific properties to the more general property using the `rdfs:subPropertyOf` in the metadata schema, an application capable of RDFS reasoning can infer that the object has “Job Cohen” as its subject. We therefore achieve interoperability without discarding the complexity of the original data.

5 Step 5: Vocabulary Alignment Using Amalgame

Amalgame (AMsterdam ALignment GenerAtion MEtatool) is a tool for finding, evaluating and managing vocabulary alignments. The explicit goal for this platform is not to produce ‘yet another alignment method’ but rather support an interactive approach to vocabulary alignment where the user can combine existing matching techniques into an alignment workflow targeted to the data set at hand using a workflow setup [7]. Amalgame is freely available and documented at <http://semanticweb.cs.vu.nl/amalgame/>.

Which blocks to use and in what order or combination is fully controlled by the user. Furthermore, produced alignments (both intermediate and end results) can be easily evaluated to give insight in their quality. It is designed for the large but shallow vocabularies typical in the cultural heritage domain and to provide support to analyze large sets of correspondences.

Alignment within Amalgame is a process where the user iteratively applies matchers, partitions the result set, and applies new matchers or a filter. After each step the user typically analyzes the results to determine the next step. User can quickly test various matching techniques (based on labels, structure etc.) with different settings on their specific source data. Through this testing, the user gains insight into which techniques perform well, both with regards to precision (what percentage of the produced correspondences are correct) as well as coverage (how many source concepts are mapped). High-precision result sets of correspondences can be consolidated and published alongside the source RDF. To this end, Amalgame features:

- An interactive workflow composition functionality. Using this setup, the user can iteratively select various actions on vocabularies or intermediate mapping results. Filters can be applied to select subsets of concepts or of mapping results. By concatenating these actions, an alignment workflow emerges.
- A statistics function, where statistics for intermediate and end-result alignment sets are shown.
- An evaluation view, where subsets of alignments can be evaluated manually.

6 Case Study: Amsterdam Museum

In this section, we describe how we used the above described methodology to convert the Amsterdam Museum metadata and vocabularies to five-star Linked Data that is compatible with the Europeana Data Model (EDM). This includes linking the vocabularies used in the metadata values to external sources. The input files, the intermediary and converted RDF, the schema mapping files as well as the alignment strategies are all available online at <http://semanticweb.cs.vu.nl/lod/am/data.html>, where they are listed for each step. We here present an overview.

6.1 Amsterdam Museum Metadata

The Amsterdam Museum⁴ is a Dutch museum hosting cultural heritage objects related to Amsterdam and its citizens. Among these objects are paintings, drawings, prints, glass and silver objects, furniture, books, costumes, etc. all linked to Amsterdam's history and modern culture. At any given moment, around 20% of the objects are on display in the museum's exhibition rooms, while the rest is stored in storage depots.

As do many museums, the Amsterdam Museum uses a digital data management system to manage their collection metadata and authority files, in this case the proprietary Adlib Museum software⁵. As part of the museum's policy of sharing knowledge, in 2010, the Amsterdam Museum made their entire collection available online using a creative commons license. The collection can be browsed through a web-interface⁶. Second, for machine consumption, an XML REST API was provided that can be used to harvest the entire collection's metadata or retrieve specific results based on search-terms such as on creator or year. The latter API has been used extensively in multiple Cultural Heritage-related app-building challenges.

6.2 The Europeana Data Model

Europeana enables people to explore the digital resources of Europe's museums, libraries, archives and audio-visual collections. Among its goals, Europeana will act as an aggregator for European Linked Cultural Data. The idea is that the data from individual cultural heritage institutions can be integrated by mapping them to a common metadata model: the Europeana Data Model (EDM) [5].

EDM adheres to the principles of the Web of Data and is defined using RDF. The model is designed to support the richness of the content providers metadata but also enables data enrichment from a range of third party sources. EDM supports multiple providers describing the same object, while clearly showing the provenance of all the data that links to the digital object. This is achieved by incorporating the *proxy-aggregation* mechanism from the Object Re-use and Exchange (ORE) mode⁷. For our purpose, this means that an Amsterdam Museum metadata record gives rise to both a

⁴ <http://amsterdammuseum.nl>

⁵ <http://www.adlibsoft.com/>

⁶ <http://collectie.ahm.nl>

⁷ <http://www.openarchives.org/ore/>

proxy resource as well as an aggregation resource. The RDF triples that make up the object metadata (creator, dimensions etc.) have the proxy as their source while the triples that are used for provenance (data provider, rights etc.) as well as alternate representation (e.g. digital thumbnails) have the aggregation resource as their source.

For its actual metadata, the EDM builds on proven metadata standards that are used throughout the cultural heritage domain. Dublin Core is used to represent object metadata and SKOS to represent thesauri and vocabularies. Next to these properties, a limited number of EDM-specific properties are introduced, including predicates that allow for event-centric metadata.

6.3 Producing RDF Using XMLRDF

We here report on the XML to RDF conversion, give examples of rewrite rules and discuss specific issues. For the first step of our conversion effort, we used the data as harvested through the OAI-PMH interface. This resulted in three separate XML datasets: The *object metadata*, a *thesaurus* with concepts used in the object metadata fields and a *person authority file*. The three datasets were first transformed into three RDF datasets using the pure syntactic rewriting facility of XMLRDF (step 2). Then each set was restructured using separate rewriting rules (step 3) and separate schema mapping files (step 4). We report on each of these in the next sections.

For all three datasets, we mapped the XML attributes to RDF using a base namespace <http://purl.org/collections/nl/am/>. We used these purl.org URIs since for this conversion we were not in the position to use the Amsterdam Museum namespace for our ClíoPatria server.

Object Metadata. The object metadata consist of metadata records for the 73.447 objects including creator, dimensions, digital reproductions, related exhibitions etc. This dataset was converted to 6,301,012 triples in the crude RDF transform. 669,502 distinct RDF subjects were identified as well as 94 distinct properties. 497,534 of the RDF objects were identified as literals.

To enrich the crude RDF, a total of 58 XMLRDF rewrite rules were made. Of these rules, 24 were used to re-establish links to the thesaurus and 5 rules reestablished links to persons. An additional 4 rules made inter-object relations explicit. 10 rules were ‘clean-up’ rules. The remaining rules include rules that provide URIs to resources, rules that rewrite untyped literals into language-typed RDF literals, rules re-ifying nested blank nodes and rules combining literal values in one human-readable literal. Examples of these rules are shown in Figure 2 specifically **clean_empty**, **assign_uris**, **title_nl**, **content_person** and **dimensions**. The rules shown there are relatively simple for the sake of clarity.

The 55 rules that are executed first are not EDM-specific, as they translate the XML record structure into their RDF equivalent. The three rules that are executed last map the data to the EDM. These rules explicitly build the aggregation-proxy construct for each of the records and moves the record properties to the appropriate resource (object metadata to the proxy, provenance data and reproduction info to the aggregation).

In total, executing the rewriting rules resulted in 5,700,371 triples with 100 predicates and 933,891 subjects, of which 566,239 are blank nodes.

We constructed an RDFS mapping file relating the 100 Amsterdam Museum properties to the EDM properties through the `rdfs:subPropertyOf` construct. Seven properties were mapped to EDM-specific properties (`ens:hasMet`, `ens:happenedAt`, etc.) and three properties were defined as subproperties of `rdfs:label`, the rest of the properties are defined as subproperties of Dublin Core properties. Two Amsterdam Museum classes ‘`am:Exhibition`’ and ‘`am:Locat`’ were defined as `rdfs:subClassOf` of the EDM class ‘`ens:Event`’.

Thesaurus Metadata. This dataset consists of 28.000 concepts used in the object metadata fields, including geographical terms, motifs, events etc. In the crude RDF transformation step, this was converted to 601,819 RDF triples about 160,571 distinct subjects, using 19 distinct properties. 55,780 RDF objects are literals.

Most term-based thesauri, including the AM thesaurus, have a more or less uniform structure (ISO 25964) for which the standard RDF representation is SKOS. We therefore chose to rewrite the AM thesaurus directly to SKOS format. For this purpose, we constructed 23 rewriting rules. 6 rules establish links by mapping literal values to URIs resulting in the SKOS object relations `skos:broader`, `skos:narrower` and `skos:related`. 4 rules mapped the original thesaurus’ USE/USEFOR constructs to `skos:altLabels` (cf. the **use.to.altlabel** in Figure 2), 6 rules were clean-up rules. The remaining rules include rules that give URIs, give type relations and relate the `skos:Concepts` to a concept scheme. In total after the rewrite, 160,701 RDF triples remain, describing 28,127 subjects using 13 properties. Since the conversion already produced most of the SKOS properties, the RDFS mapping file only contains the (new) `skos:ConceptScheme` triples and mappings that relate the Amsterdam Museum notes to from the `skos:notes`.

Person Authority File. This dataset contains biographical information on 66.968 persons related to the objects or the metadata itself. This relation includes creators, past or present owners, depicted persons etc. In the crude RDF transformation step, the person authority file was converted to 301,143 RDF triples about 66,968 distinct subjects, using 21 distinct properties. 143,760 RDF objects are literals.

Since the crude RDF was already well structured and no additional literal rewriting or mapping to URIs was required, only 2 rules are needed for the people authority file. One changes the type of the records to ‘Person’, while the second one gives URIs to the persons. These minor translations did not change the above statistics.

Since the current version of the EDM does not specify how biographical metadata is to be represented, we mapped to properties from the RDA Group 2 metadata standard⁸. These properties include given and family names, birth and death dates etc. As a side note, informed by this conversion, this metadata set is currently considered as the EDM standard for biographical information. In total 20 `rdfs:subProperty` relations were defined. The `am:Person` class was also mapped as a `rdfs:subClassOf` `ens:Agent`.

Discussion. Of course, any (semi-)automatic transformation using rules produces some erroneous results. For example, the rules re-establishing links such as the **content.person** link in Figure 2 assume unique property values (in this case the name of the person) and

⁸ <http://rdvocab.info/ElementsGr2>

that those values exist. Although in this case the name should be available and unique, there were three unmapped values after this rule was applied (for the remaining 2775 values, the correct URI was found). ClioPatria allows us to identify the erroneous values quickly. A new rule can be constructed for these, either rewriting them or removing the unmapped triples. Alternatively, the triples can be maintained, as was done in the Amsterdam Museum case.

Another example where the method is only partially successful is for two AM properties `am:contentSubject` and `am:contentPersonName`. These relate an object to either a concept or a person (for example a painting depicting a nobleman and a building). Dublin Core provides the `dcterms:subject` property which does not differentiate between the types. In the schema, we defined the AM properties as `rdfs:subProperty` of `dcterms:subject`. An application capable of RDFS reasoning can infer that the object has `dcterms:subject` both the person and the concept. We therefore achieve some interoperability without discarding the complexity of the original data as expressed using the properties of the ‘am’ namespace.

6.4 Producing Links to External Sources Using Amalgame

To illustrate step 5, we aligned the thesaurus and person authority file with a number of external sources using Amalgame. We report on the final alignment strategies.

Thesaurus. We mapped the thesaurus partly to the Dutch AATNed⁹ thesaurus and partly to GeoNames¹⁰. The thesaurus was first split into a geographical and a non-geographical part consisting of 15851 and 11506 concepts respectively. We then aligned the Dutch part of the geographic part (953 concepts with a common ancestor “Netherlands”) to the Dutch part of GeoNames using a basic label-match algorithm. This resulted in 143 unambiguous matches. We performed an informal evaluation by manually assessing a random sample of the mappings. This resulted in indicated a high quality of the matches (90%+ precision). The AM concepts for which no match was found include Amsterdam street names or even physical storage locations of art objects, which are obviously not in GeoNames.

The non-geographic part of AM was aligned with the AATNed using the same basic label match algorithm. Here, 3820 AM concepts were mapped. We then split the mapping in an unambiguous (one source is mapped to one target) and an ambiguous part (one-to-many or many-to-one). The unambiguous part was evaluated as having a high precision, the ambiguous mappings could be further disambiguated but still have a good precision of about 75%. The coverage for the non-geographic part is about 33%.

Person Authority File. The person authority file was aligned to a subset of DBpedia¹¹ containing persons using only the target `skos:prefLabels`. This resulted in 34 high quality mappings. The unmapped concepts were then aligned using the `skos:altLabels` as well, and then split in 453 unambiguous and 897 ambiguous matches, with estimated

⁹ <http://www.aat-ned.nl>

¹⁰ <http://www.GeoNames.org>

¹¹ <http://dbpedia.org>

precisions of 25% and 10% respectively. These could also be further filtered by hand. The people database was also aligned with Getty Union List of Artist Names (ULAN)¹², resulting in 1078 unambiguous matches with a high precision (100%) and an additional 348 ambiguous matches, with a slightly lower estimated precision. Although ULAN itself is not in the Linked Data Cloud, it is incorporated in VIAF, which is in the Linked Data cloud.

The main reason for the low coverage is that a very large number of AM-persons are not listed in ULAN as they are relatively unknown local artists, depicted or related persons, museum employees, or even organizations. Mapping to alternative sources can increase coverage here.

Produced Alignments. We identify three categories of mappings: high precision (100% correct), mid-precision (80-90% correct) and low precision (<80%). The first category can be added to a semantic layer without any further processing, whereas the second and third category might require more filtering and processing. For the Amsterdam Museum, we found high-precision mappings for $143 + 2498 + 34 + 1078 = 3753$ concepts. Although this is only a fraction of the total number of concepts, the usefulness of these mappings is much greater as they represent the part of the concepts with which the most metadata is annotated. In total, 70.742 out of the 73.447 (96%) objects are annotated with one or more concepts or persons that have been linked, with an average of 4.3 linked concepts per object. A relatively low number of high-precision mappings were found in this case study. Current work includes raising this number by mapping to other sources and using more sophisticated Amalgame alignment strategies. We here present the results mainly as an illustration of the overall methodology.

6.5 Serving Amsterdam Museum Linked Open Data

The Amsterdam museum data, consisting of the converted datasets, the schema mapping files and the high-quality mapping files are served as Linked Open Data on the Europeana Semantic Layer (ESL)¹³. The ESL is a running instance of ClíoPatria that houses other datasets that have been mapped to EDM. More information, including how to access or download the data is found at <http://semanticweb.cs.vu.nl/lod/am>.

7 Related Work

In this paper, we presented a methodology for transforming legacy data into RDF format, restructuring the RDF, establishing links and presenting it as Linked Data. A set of tools developed at the Free University Berlin provides similar functionalities. Their D2R server is a tool for publishing relational databases on the Semantic Web, by allowing data providers to construct a wrapper around the database [2]. This makes the database content browsable for both RDF and HTML browsers as well as queryable by SPARQL. The R2R tool can be used to restructure the RDF and finally the Silk tool

¹² <http://www.getty.edu/research/tools/vocabularies/ulan>

¹³ <http://semanticweb.cs.vu.nl/europeana>

is used to generate links to other data sources. One difference between D2R and our approach described here is that we assume an XML output of the original data, whereas D2R acts directly as a wrapper on the data. In the cultural heritage domain, many institutes already publish their data as XML using the OAI-PMH protocol¹⁴ as part of their normal workflow. This XML can therefore be considered their 'outward image' of the internal database and is an ideal starting place for our Linked Data conversion. A second difference is that we explicitly provide tools for interactive conversion and alignment of the data. XMLRDF is part of the ClioPatria RDF production platform, allowing for rapid assessment and evaluation of intermediary RDF.

Other tools that can be used to produce RDF include tools based on XSL transformations (XSLT). An example of such a tool is the OAI2LOD Server, which also starts from an OAI-PMH input, converts this to RDF using XSLT and provides RDF-browser and SPARQL access to the data [4]. Another example is the OAI-PMH RDFizer which is one of Simile's RDF transformation tools [8]. Such tools make use of the fact that RDF can be serialized as XML and do the conversion by restructuring the XML tree. A lot of cultural heritage institutions have relatively complex datastructures and will therefore need more complex operations in parts of the conversion [6]. Even though XSLT as a Turing-complete language has the same level of expressivity as Prolog, a number of common rewriting operations are better supported by Prolog and its rewriting rule language. Specifically, the XMLRDF rules can use Prolog and ClioPatria's RDF reasoning ability, taking existing triples into account when converting new triples.

8 Discussion and Future Work

In this paper, we presented an interactive methodology to ingesting and converting cultural heritage metadata as well as linking it to external data sources and publishing it as Linked Open Data. We described how this is supported by the ClioPatria semantic server and more specifically by the XMLRDF tool for conversion and the Amalgame tool for alignment. We illustrated this through a case study where the entire collection of the Amsterdam Museum was converted using these tools. The tools are designed to support domain experts in the conversion in such a way that they will be able to convert the legacy XML data to RDF using as much of the original ontological commitments as possible. In other words, as explained in Sections 1 and 3, we aim to retain as much as possible of the richness (and semantic choices) of the original metadata.

The tools assume knowledge of XML and RDF. For basic data conversion, (Prolog) programming skills are not necessary. However, for more complex transformations, some programming will be required. In general, transformation of rich data requires technical skills to solve the more complex cases. As described in Section 4, we use a metadata mapping to a given interoperability level and through this adhere to that set of ontological commitments. This being said, in every individual conversion, there will be semantic choices have to be made by the data expert.

Although the tools are designed to be interactive and transparent and therefore usable by the institutions' collection managers, for this case study the authors performed the

¹⁴ <http://www.openarchives.org/OAI/openarchivesprotocol.html>

conversion themselves, to showcase the tools and present a number of re-usable XML-RDF conversion recipes. In the context of Europeana, the XMLRDF tool has been used by the authors, as well as by external parties to convert archival, museum and library data. A number of these converted datasets are presented in the ESL. The Amalgame tool has also been used by external parties and we are currently in the process of having alignments done by actual collection managers.

Acknowledgements. We like to thank Marijke Oosterbroek from Amsterdam Museum and Steffen Henniecke from Humboldt University for their feedback and assistance. This work was partially supported by the PrestoPRIME and EuropeanaConnect projects.

References

1. Berners-Lee, T.: Linked data - design issues (2006), <http://www.w3.org/DesignIssues/~LinkedData.html>
2. Bizer, C., Cyganiak, R.: D2r server – publishing relational databases on the semantic web (poster). In: International Semantic Web Conference (2003)
3. Frühwirth, T.: Introducing simplification rules. Tech. Rep. ECRC-LP-63, European Computer-Industry Research Centre, Munchen, Germany (October 1991); Presented at the Workshop Logisches Programmieren, Goosen/Berlin, Germany, and the Workshop on Rewriting and Constraints, Dagstuhl, Germany (October 1991)
4. Haslhofer, B., Schandl, B.: Interweaving oai-pmh data sources with the linked data cloud. *Int. J. Metadata, Semantics and Ontologies* 1(5), 17–31 (2010), <http://eprints.cs.univie.ac.at/73/>
5. Isaac, A.: Europeana data model primer (2010), <http://version1.europeana.eu/web/europeana-project/technicaldocuments/>
6. Omelayenko, B.: Porting cultural repositories to the semantic web. In: Proceedings of the First Workshop on Semantic Interoperability in the European Digital Library (SIEDL 2008), pp. 14–25 (2008)
7. van Ossenbruggen, J., Hildebrand, M., de Boer, V.: Interactive Vocabulary Alignment. In: Gradmann, S., Borri, F., Meghini, C., Schuldt, H. (eds.) TPD 2011. LNCS, vol. 6966, pp. 296–307. Springer, Heidelberg (2011)
8. Project, T.S.: Oai-pmh rdfizer (2012), http://simile.mit.edu/wiki/OAI-PMH_RDFizer (retrieved December 2011)
9. Tordai, A., Omelayenko, B., Schreiber, G.: Semantic Excavation of the City of Books. In: Proceedings of the Semantic Authoring, Annotation and Knowledge Markup Workshop (SAAKM 2007), pp. 39–46. CEUR-WS (2007), <http://ftp.informatik.rwthachen.de/Publications/CEUR-WS/Vol-289/>
10. Wielemaker, J., Hildebrand, M., van Ossenbruggen, J., Schreiber, G.: Thesaurus-Based Search in Large Heterogeneous Collections. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 695–708. Springer, Heidelberg (2008), http://dx.doi.org/10.1007/978-3-540-88564-1_44
11. Wielemaker, J., de Boer, V., Isaac, A., van Ossenbruggen, J., Hildebrand, M., Schreiber, G., Henniecke, S.: Semantic workflow tool available. EuropeanaConnect Deliverable 1.3.1 (2011), http://www.europeanconnect.eu/documents/D1.3.1_eConnect_Workflow_automation_method_implementation_v1.0.pdf

Curate and Storyspace: An Ontology and Web-Based Environment for Describing Curatorial Narratives

Paul Mulholland, Annika Wolff, and Trevor Collins

Knowledge Media Institute, The Open University, Walton Hall,
Milton Keynes, MK7 6AA, UK
{p.mulholland,a.l.wolff,t.d.collins}@open.ac.uk

Abstract. Existing metadata schemes and content management systems used by museums focus on describing the heritage objects that the museum holds in its collection. These are used to manage and describe individual heritage objects according to properties such as artist, date and preservation requirements. Curatorial narratives, such as physical or online exhibitions tell a story that spans across heritage objects and have a meaning that does not necessarily reside in the individual heritage objects themselves. Here we present *curate*, an ontology for describing curatorial narratives. This draws on structuralist accounts that distinguish the narrative from the story and plot, and also a detailed analysis of two museum exhibitions and the curatorial processes that contributed to them. *storyspace*, our web based interface and API to the ontology, is being used by curatorial staff in two museums to model curatorial narratives and the processes through which they are constructed.

Keywords: Cultural heritage, story, plot, narrative, ontology, museum.

1 Introduction

Current museum metadata schemes and content management systems focus on the description and management of the individual heritage objects that the museum holds in its collection. An important responsibility for museums, as well as preserving the collection, is to communicate to the public. One key form of communication is through the development of curatorial narratives. These curatorial narratives may take the form of physical museum exhibitions (possibly supplemented by other materials such as audio guides and booklets) or online presentations. Curatorial narratives express meaning across a number of heritage objects. The meaning of the narrative cannot be expressed or derived purely from the metadata of the heritage objects that it contains. Currently, there is therefore no support for the description and search of museum narratives based on their meaning rather than the objects that they contain.

This work is being conducted as part of DECIPHER, an EU Framework Programme 7 project in the area of Digital Libraries and Digital Preservation. A key aim of DECIPHER is to allow users interactively to assemble, visualize and explore, not just collections of heritage objects, but the knowledge structures that connect and give them meaning. As part of this work, the curate ontology has been developed in order that we can understand and describe the reasoning behind a curatorial narrative.

This can be used to describe and search of narratives based on their meaning rather than just the heritage objects that they contain. The ontology will also be used to drive computational assistance for the human construction of narratives.

The rest of this paper is structured as follows. The next section describes related research in the formal description of events and its use in providing navigation across heritage objects. Section 3 describes the curate ontology and how it can be used to describe curatorial narratives. It draws on structuralist theories that distinguish the narrative presentation from the conceptualization of the story and plot. Section 4 describes storyspace, an API and web interface to the ontology. Section 5 describes the use of storyspace to model curatorial narratives on the conceptual, story level. Section 6 summarises the findings of a structured interview with two members of museum curatorial staff that used storyspace to model curatorial narratives over a two month period. Section 7 presents conclusions and ongoing work.

2 Related Work

Although there has not been any previous attempt to develop an ontology of curatorial narrative, some previous research has used metadata to generate or describe presentations that include multiple heritage objects. These have made use of event-based ontologies and metadata schemes to conceptually interconnect heritage objects.

Bletchley Park Text [1, 2] uses historical interviews described according to CIDOC CRM [3] event-based metadata to assemble an online newspaper in response to a query. Interviews are grouped according to the common people, places and objects mentioned in their constituent events. Hyvonen et al. [4, 5] used event-based metadata to assemble further heritage objects around another that acted as a hub or backbone to the presentation. In one case a movie about the ceramics process was represented as events and linked to other resources related to concepts (e.g. people objects) featured in the events [5]. In the other, events were used to generate links within a poem and to external resources giving additional information [5].

Wang et al. [6, 7] use content metadata and user preferences to suggest related heritage objects of interest. van Hage et al. [8] combine this with a real-time routing system to provide a personalized museum tour guide creating a conceptual path across a number of heritage objects. The personalized tour guide developed by Lim and Aylett [9] associated heritage objects with a metadata structure they termed a story element that comprised events, people, objects, museum location and causal relationships to other story elements. Recommendations were made based on casual relationships and shared items contained in story elements.

Finally, van Erp et al. [10] describe a prototype system for event-driven browsing. The system suggests related heritage objects based on their associated events. By selecting related heritage objects the user can create a pathway through the heritage objects.

Research related to the interconnection of heritage objects based on event metadata has made use of a number of ways of formally representing events. CIDOC CRM is an upper level ontology for the cultural heritage sector [3]. CIDOC CRM affords an event-based representation of metadata. This provides a way of representing the

changing properties of a heritage object over time (for example the changing ownership of a painting). Another particular advantage of the CIDOC CRM ontology is that it facilitates interoperability among museum metadata schemes. Other approaches to the formal representation of events have been proposed such as LODE [11] and SEM [12]. These aim to limit ontological commitment in order to broaden the range of events that can be represented and are not focused specifically on the heritage domain.

Other work has looked at separating the interpretation of events from the representation of the events themselves. This simplifies the properties of the event and allows multiple (possibly conflicting) interpretations of the same event to be modeled, for example alternative perspectives on the cause-effect relationship between events. The Descriptions and Situations (DnS) ontology design pattern applied to events [13] supports this by distinguishing a situation (e.g. two events) from its description (e.g. cause-effect relationship between them).

3 The Curate Ontology

Development of the curate ontology drew on an analysis of the curatorial processes as conducted in museums. We analyzed in detail the curatorial processes involved in the development of two exhibitions held by museum partners in the DECIPHER project. The two exhibitions were Gabriel Metsu [14] held at the National Gallery of Ireland and The Moderns [15] held at the Irish Museum of Modern Art (IMMA). The Gabriel Metsu exhibition focused on the work of a single artist. The Moderns presented an Irish perspective on Modernism in art and covered a range of artists from around 1900 to 1970. The analysis involved visiting the exhibitions (in the case of the Moderns) and analysis of associated resources (e.g. transcript of the audio guide, museum panels, booklets). A one-day workshop was held focused on each exhibition. The first half of the workshop was devoted to presentations by staff involved in the curatorial process. This included curators of the physical exhibition and others involved in interpretation and producing narratives around the exhibition such as staff working on the education programme. The second half was focussed on discussions motivated by scenarios and paper prototypes of what types of interaction could be envisaged from tools resulting from the project. Further details on the analysis of curatorial practice can be found elsewhere [16, 17].

This work produced a rich description of the curatorial process. In developing the ontology we proposed two working hypotheses to guide our interpretation of the data, and that could also be tested as part of the interpretation process. Our first hypothesis was that a curatorial narrative should have the generic properties found in other types of narrative such as a novel or a film. Structuralist theories can be used to distinguish story, plot and narrative [18]. The story is the set of events that can be told. The plot imposes a network of relationships on the events of the story signifying their roles and importance in the overall story and how they are interrelated (e.g. a causal relationship between two events). The plot therefore turns a chronology of events into a subjective interpretation of those events. The narrative is a particular telling of that story and plot in some particular media.

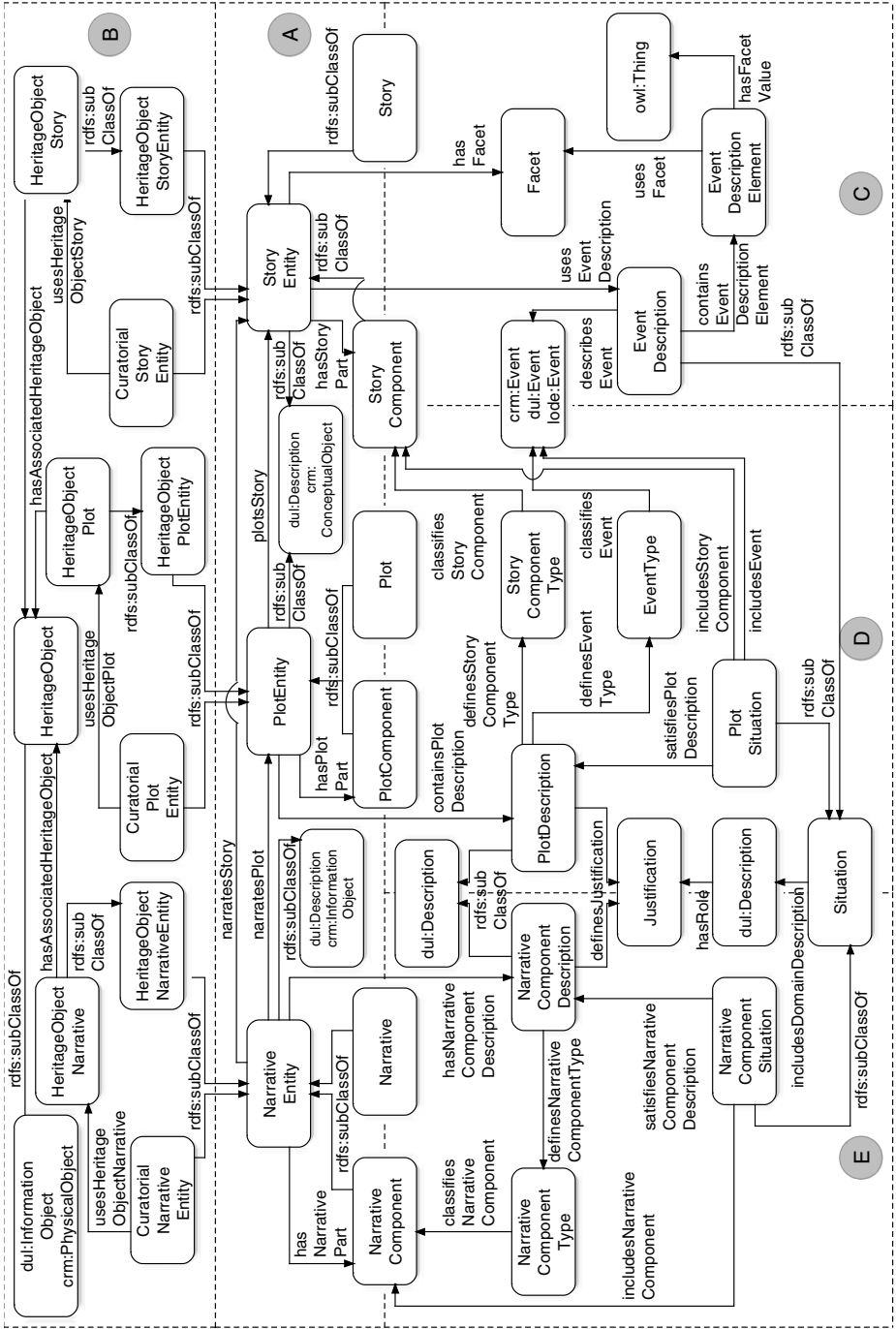


Fig. 1. Overview of the curate ontology

Second, we hypothesized that curatorial narratives are not only presentations but the product of a process of inquiry in which the heritage objects provide a source of evidence. Narrative inquiry [19] is a methodology in which research can be conducted by selecting or constructing a story of events, interpreting these by proposing and testing a plot and then presenting this as a narrative to the research community. Narrative inquiry can be contrasted with the scientific method as a research methodology. In narrative inquiry the plot can be thought of as essentially a hypothesis that is tested against the story, being the data of the experiment. Story, plot and narrative therefore constitute a process rather than only associated types of description.

These hypotheses, in combination with an iterative design process in participation with the two museums, led to the construction of the curate ontology¹. An overview is shown in figure 1. CIDOC CRM [3] and DOLCE+DnS Ultralite (DUL)² are used as upper level ontologies for curate. There are five main components to the ontology, indicated by the areas A to E in figure 1. These will be described in the following five subsections.

3.1 Story, Plot and Narrative

Part A of the ontology describes the concepts of story, plot and narrative and how they are related (see figure 1). A narrative presents both a story and a plot. A story is interpreted by a plot. A number of plots may be created for the same story. In some cases a narrative may present a story but have no associated plot. This would indicate that the narrative is recounting a chronology of events (i.e. a chronicle) but offers no interpretation of them. Figure 2 shows the relationship between a story of Gabriel Metsu and an associated plot and story. The story itself contains events. The events of a story will be considered further in section 3.3.

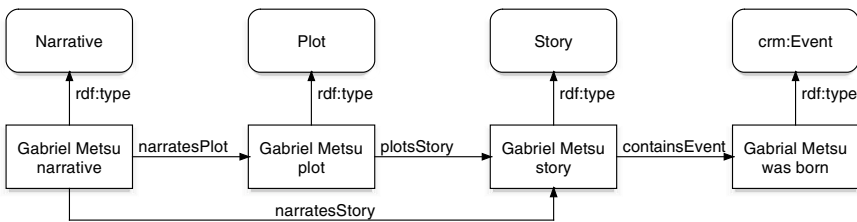


Fig. 2. Modeling narrative, plot, story and event

3.2 Stories and Heritage Objects

The relationship between heritage objects and the story, plot and narrative is illustrated in part B of figure 1. Discussions with museum partners made clear that we needed to distinguish two types of narrative. A heritage object narrative tells a story

¹ <http://decipher.open.ac.uk/curate>
² <http://ontologydesignpatterns.org/ont/dul/DUL.owl>

about a heritage object. A heritage object may have multiple heritage object narratives. These heritage object narratives may draw on different aspects of the heritage object such as how the object was created, some insight it gives about the life of the artist, what is depicted in the heritage object or who has owned it. A curatorial narrative threads across a number of heritage object narratives. It makes conceptual relationships across a set of exhibits, yielding more complex insights than could be made from the exhibits individually.

This approach to modeling has two advantages. First, it allows us to distinguish alternative stories of the same heritage object. Second it allows us to model, through the heritage object story, what contribution a heritage object brings to a curatorial story. The relationship between a heritage object and an event, mediated by the heritage object story, plays the role of the illustrate property in the LODE ontology [11] that associates an object with an event. The mediating role of the heritage object story though allows us to represent through which story the event is associated with the object.

Figure 3 shows two heritage object stories of the painting “A Woman Reading a Letter”. One is concerned with how the painting illustrates the brush technique of the artist. The other is concerned with a more recent incident in which the painting was stolen and recovered. The story about brush technique is relevant to the curatorial story. The curatorial story shows how Metsu’s technique changed over time, drawing on a number of heritage object stories illustrating technique at a particular point in time.

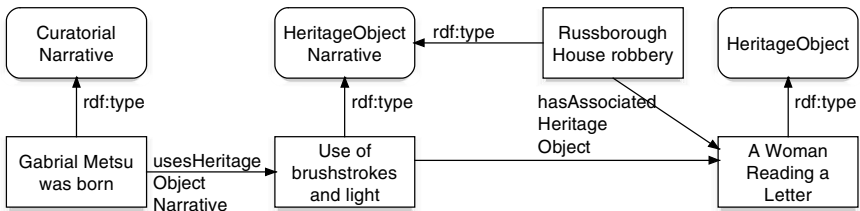


Fig. 3. Modeling curatorial stories, heritage object stories and heritage objects

3.3 Facets, Events and Event Descriptions

The relationship of stories to events and their description within a story is represented in part C of figure 1. An event included in a story has an associated event description. This describes the event according to the facets of the story. The facets are dimensions according to which the event can be described. Following [11], this interpretation of an event in the context of a story is modeled as a DUL:Situation, the story itself constituting a DUL:Description. From a narrative inquiry perspective [19] this allows us to move from a chronicle, a set of events that just have a position in time, to what is called a storyline. A storyline also describes the events in other ways relevant to the investigation. For example, if changing patterns in the location of events over time was of interest then location would be a facet of the story used to describe the events. Similarly, if the investigation was considering the incidence of certain types of activity then activity type of the event would be a facet of the story.

This type of organization was identified in the exhibitions and recognized as a general organizational principle by curatorial staff. In the case of Gabriel Metsu, the first part of the exhibition illustrated the development of his technique early in his career. Here, time was the primary organizing principle. In the later components of the exhibition, which focus on when his technique has fully developed, organization was thematic, indicating whether the work was related to, for example, family, reputation or religion.

Figure 4 shows the association of an event with the theme of religion. Within the context of this story, the event has an event description. The event description has event description elements that assign one or more values for a defined facet of the story. The event description is used to reify the interpretative elements of the event description.

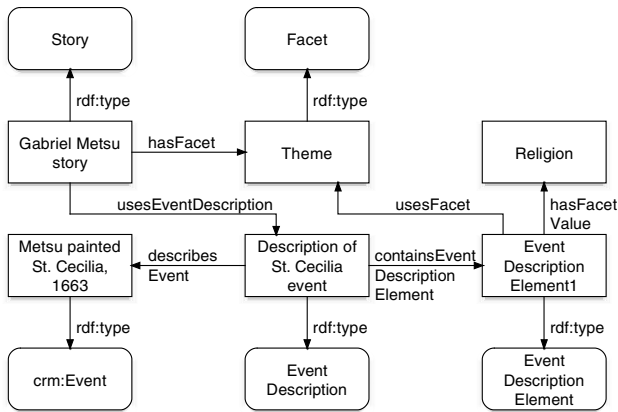


Fig. 4. Modeling events and event descriptions

3.4 Plots, Events and Story Components

The emplotment of a story (i.e. its association with a plot) is represented in part D of figure 1. The approach taken to modeling plot makes use of the Descriptions and Situations (DnS) pattern applied to events [13]. The ontology supports the definition of plot relationships across events, story components or both. For example, a plot relationship may define that one event causes another. In practice, the relationships found between events tend to be subtler, for example the specification of an influence between events. This is not only a feature of curatorial narratives. For example, in an analysis of novels, Chatman [20] highlights “happenings” that have no cause within the narrative. Similarly, plot relationships may be specified between story components (e.g. this area in space and time is more peaceful than another) or between both events and story components (e.g. this event was pivotal between two areas in space and time).

Figure 5 shows an example in which one event (Metsu drew ‘Sketch of a female figure’) is classified as being preparatory for another event. As in the DnS ontology design pattern a justification can be added in support of the defined relationship. Here, visual similarity is used to justify the proposed plot relationship.

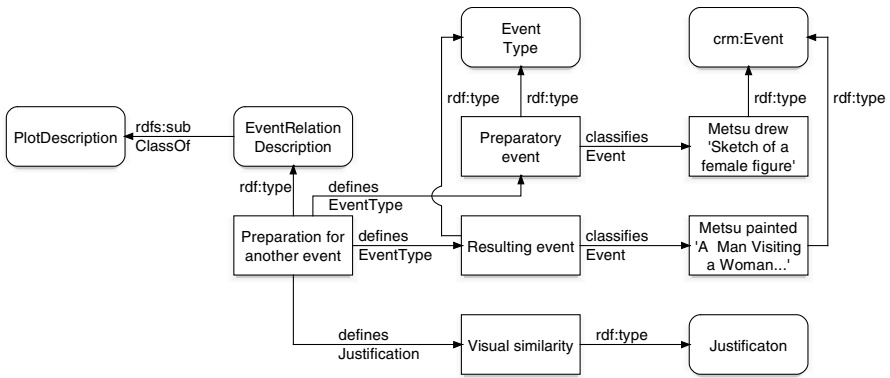


Fig. 5. Modeling plot relations between events

3.5 Narrative Components

The narrative presentation of a story and plot is represented in part E of figure 1. This also makes use of the Descriptions and Situations (DnS) pattern to specify structural relationships between components of the narrative. A curatorial narrative within a physical museum space may vary considerably from the underlying story due to different types of physical constraint. First, differences may be due to the fixed structure of the museum space. For example, the exhibition space at IMMA is made up of a number of relatively small rooms and interconnecting doors and corridors. This can result in a story component spanning a number of physical spaces, with the organization of heritage objects and interpretation panels across those spaces being as much determined by aesthetic and size constraints as the conceptual organization of the story.

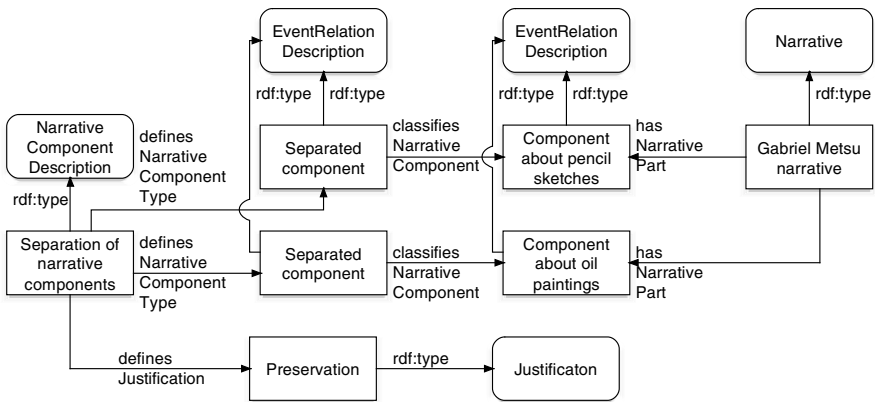


Fig. 6. Modeling relationships between narrative components

Some differences between story and narrative organization may result from preservation constraints of the exhibits. For example, pencil sketches need to be displayed in darker conditions than are used for displaying paintings, therefore need to be separated in a physical museum space though not on the conceptual space of the story. Figure 6 represents this example in which a narrative has been broken down in order to separate components for preservation reasons.

4 Storyspace

Storyspace is an API to the curate ontology and currently a web interface to the story and heritage object components of the ontology (i.e. sections B and C of figure 1). The decision was taken to develop a web interface in order that museum participants in the design process could try to model aspects of curatorial narratives for themselves, understand the implications of the ontology and provide feedback on both the ontology and web interface. The web interface was developed using the Drupal CMS³. In Drupal, pieces of content (which may be rendered as a whole or part of a web page) are represented as nodes. A Drupal node is of a node type that defines the content fields of the node. For example, a node type for representing film reviews may have fields for the film title, a textual review of the film and integer representing a rating of the film. Corlosquet et al. [21] drew on the parallel between Drupal content type, fields and nodes and the classes, properties and individuals of an ontology and knowledge base. They developed support for Drupal content to be published semantically according to this mapping.

Building on this idea we developed a set of Drupal content types for representing the story and heritage object parts of the curate ontology. The content types developed were: story, event, facet, heritage object, data and reference. The story content type is used to represent both heritage object and curatorial stories. The reference content type does not map to the curate ontology and represents a bibliographic source for a curatorial or heritage object story. This was added at the request of the museums and can be represented formally using existing bibliographic ontologies such as BIBO⁴. The data content type represents additional metadata associated with an individual of the curate ontology represented in storyspace. For example this is used to represent additional metadata associated with an event imported into storyspace. An `rdfs:seeAlso` property is defined between the entity (e.g. the event) and the additional metadata.

Storyspace required a slightly more flexible mapping to the ontology than demonstrated by Corlosquet [21] as, for example, heritage object and curatorial stories and their components are all represented by the same Drupal content type but map to different classes in the ontology. A Drupal module was developed in storyspace to allow the `rdf:type` of a node to be defined according to any combination of fields and values of the Drupal node. A formal description of the storyspace

³ <http://drupal.org>

⁴ <http://bibliontology.com>

content is held in a Sesame⁵ triple store using the ARC2 library⁶. The ontology API is tied to the creation, update and deletion functions of the Drupal nodes and can be triggered programmatically or via the web-based user interface. Also, similar to Corlosquet et al. [21], appending rdfxml to a Drupal path presents the metadata of the node. Storyspace also makes use of Simile Exhibit⁷ to allow the user to visualize events of the story according to selected facets. The next section describes how storyspace has been used by curatorial staff at the museums to model curatorial stories. For this exercise the curatorial staff took the two exhibitions that had been studied previously (The Moderns and Gabriel Metsu) and modeled the underlying story of the exhibitions and related resources. The longer term intended use for storyspace is to model a future exhibition and use it in working toward the narrative presentation. However, the reverse engineering of the stories from the final narratives provided a good test case and access to a number of existing resources that could be managed in storyspace.

5 Modeling Stories with Storyspace

Although storyspace is being used to model a number of classes within the curate ontology (e.g. storyspace, heritage object, facet, event, event description) in the interface we chose to emphasise the story and subjugate the role of the other components in the interface as elements of a story. In the primary menu (the dark band toward to top of the screen in figure 7) curatorial and heritage object stories are therefore represented in the primary menu and other entities are accessible either through the stories in which they are contained or through the Resources menu.

In figure 7 a curatorial story has been selected entitled “Our collection: IMMA publication of art packs for children aged six to twelve years old”. In the left hand menu the Drupal fields (i.e. ontological properties) of the story can be accessed. These correspond to components of the story, the events it contains, its facets, references and also Simile Exhibit visualisations that can be used to visualize the story’s events according to its defined facets.

Figure 7 is showing the heritage object stories of a curatorial story developed to communicate an exhibition to schoolchildren. The heritage object stories contain a view of the heritage object comprising its title in bold, a thumbnail and standard collection information. The heritage object may participate in multiple heritage object stories. Each of these heritage object stories may themselves be used in multiple curatorial stories.

Figure 8 shows a list of the heritage object stories of a particular heritage object. This list is accessed by selecting Heritage Objects from the Resources menu, and then selecting the heritage object, in this case Sounion by Cecil King, and then selecting the Object stories for that heritage object from the left hand menu.

⁵ <http://www.openrdf.org>

⁶ <http://arc.semsol.org>

⁷ <http://www.simile-widgets.org/exhibit>

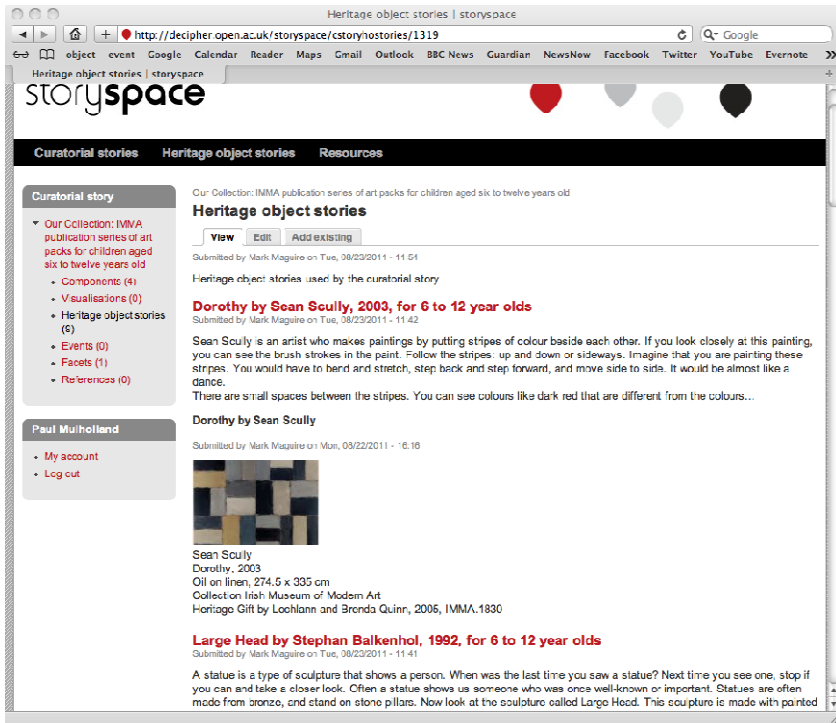


Fig. 7. Heritage object stories within a curatorial story aimed at children

When an event is included in a story it can be described according to the facets of the story. In figure 9 the event of Gabriel Metsu painting a self-portrait is being viewed. Values have been specified for the three facets of the story. In storyspace, facets can be defined as accepting manually entered values or mapped to a property. In this second case, object values from the event metadata triples that have the event as a subject for that property are displayed. Simile exhibit visualisations can be generated for a story from a forms-based interface in which the facets to be visualised, colour-coded and included in the lens are specified. Figure 10 shows a visualisation of events from the Gabriel Metsu story.

Currently, the events of a story can be either entered manually or imported from Freebase⁸. Text associated with a story can be selected to display a list of events using the Freebase search API. In figure 11, George Bernard Shaw has been selected from the story text (left) and a list of events are displayed related to this text string. These events can be viewed and added into the story. Metadata related to a selected event is imported as a related data node. Facets associated with Freebase properties (e.g. /time/event/start_date) can be used to automatically add facet values to the story.

⁸ <http://www.freebase.com>

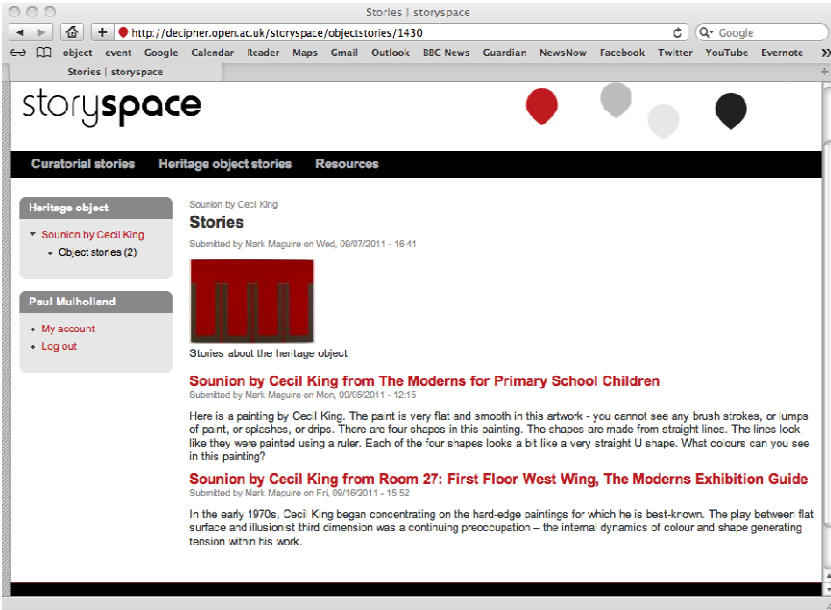


Fig. 8. Heritage object stories of the same heritage object

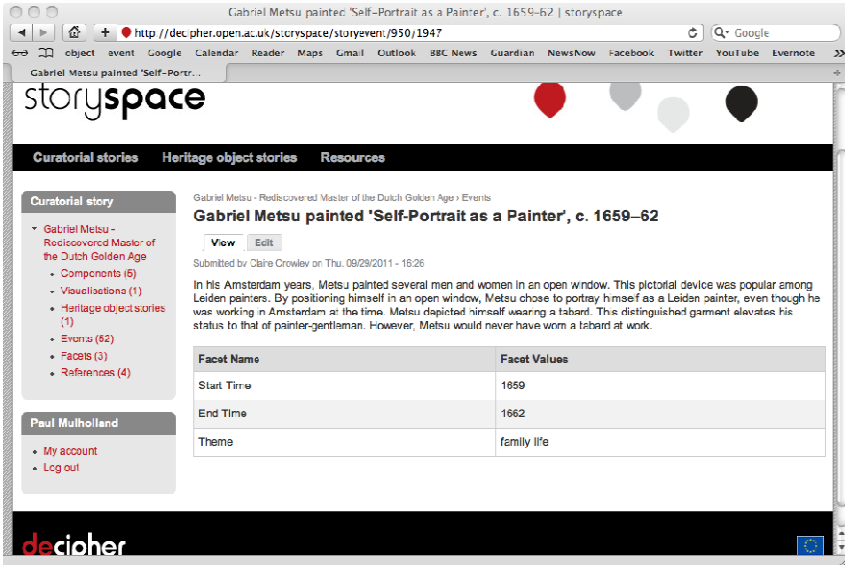


Fig. 9. Facet values of an event described within a story

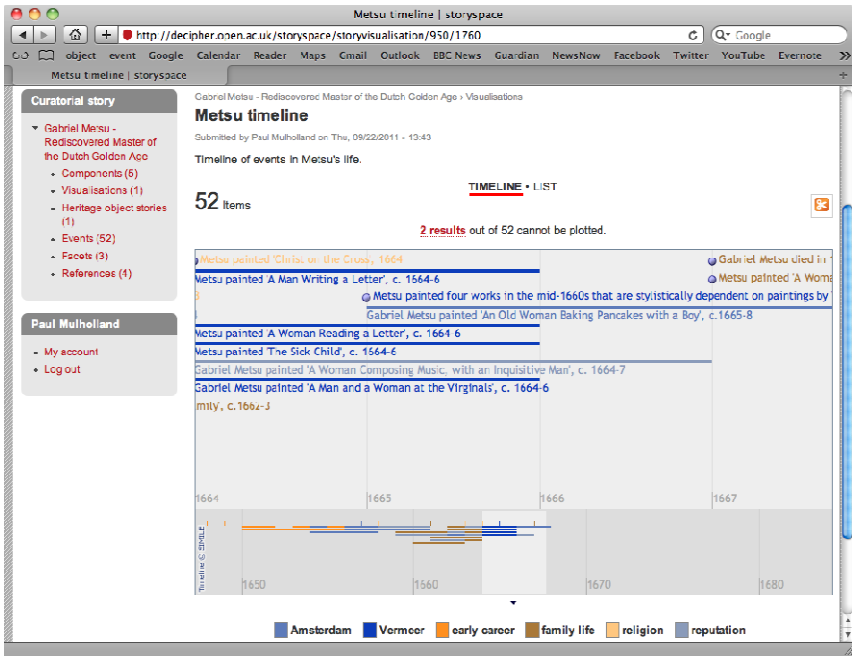


Fig. 10. Timeline visualization of events in the Gabriel Metsu story

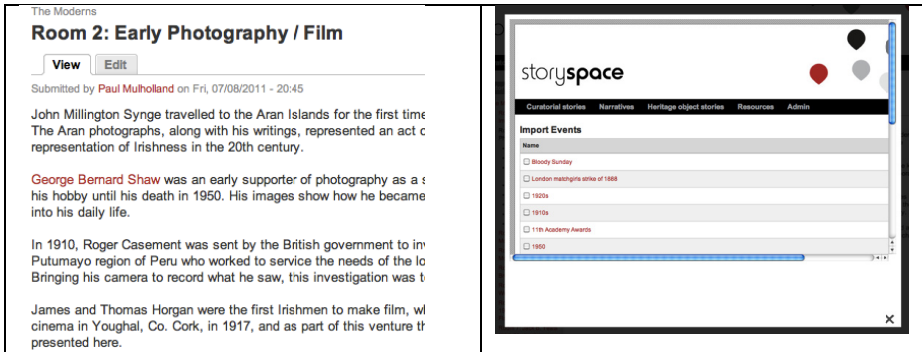


Fig. 11. Adding events from Freebase to a story

6 Structured Interview

A structured interview was conducted with the two members of curatorial staff who had contributed the most to storyspace over a two month period. One was from each of the participating museums. Questions covered the main concepts of curate and how they are navigated and authored in storyspace. Overall they reported finding it easy to navigate storyspace and add the main types of content. This was evidenced by the content that had been added related to the two exhibitions. The modelling of heritage objects, heritage object stories and curatorial stories was found to be relatively straightforward. It was also found to be useful to navigate the different heritage object

stories that had been told related to a heritage object and the curatorial stories in which they had been included.

The representation of events within a story was found to be more difficult with contemporary art as found in The Moderns exhibition. For the stories in the Gabriel Metsu exhibition and even the earlier parts of The Moderns exhibition, the events in the story were easier to identify. For the more recent works (e.g. from the 1960s or 1970s) the events were less clear. For living artists and in situations where many sources of evidence are not in the public domain, a historical perspective is harder to establish and therefore key events of the story are harder to identify.

For both exhibitions the most problematic concept was facet. These could be used to model various themes of the events but it was not always clear how best to model this. For example, a single theme facet could be defined with a number of possible values or it could be broken down into a number of facets each representing a sub-theme.

A further issue discussed in the interviews was alternative nomenclature for some of the concepts, such as heritage object stories and facets, when used in storyspace by specific audiences. Although the concepts were found to enable the successful representation of museum storytelling there are no existing, generally used terms in museum practice that can be mapped to them. The option of specialized labels in storyspace for particular museum groups was discussed as a possible extension. This will be considered further during wider trials with museum staff.

7 Conclusions and Future Work

We have developed curate, an ontology for describing curatorial narratives. This draws on structuralist theories that distinguish story, plot and narrative. Storyspace has been developed as an API and web interface to the ontology. Two exhibitions and their related stories have been used to motivate development of the ontology and also help validate it by modelling the stories using storyspace. Current work is focussed on using the curate ontology to provide assistance in the construction of curatorial narratives on each of the story, plot and narrative levels. On the story level, we are investigating how the author can be assisted in organising the story in terms of its components and their organising facets. On the plot level we are investigating how the plot descriptions can be suggested for a given story. Finally, on the narrative level, we are looking at how the composition of the narrative structure can be suggested, given a story and plot.

Acknowledgements. This work was supported by the DECIPHER project (270001), funded by the EU 7th Framework Programme in the area of Digital Libraries and Digital Preservation.

References

1. Collins, T., Mulholland, P., Zdrahal, Z.: Semantic Browsing of Digital Collections. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 127–141. Springer, Heidelberg (2005)
2. Mulholland, P., Collins, T., Zdrahal, Z.: Bletchley Park Text. *Journal of Interactive Media in Education* (2005), <http://jime.open.ac.uk/2005/24>
3. Crofts, N., Doerr, M., Gill, T., Stead, S., Stiff, M. (eds.): Definition of the CIDOC Conceptual Reference Model (2010), http://www.cidoc-crm.org/official_release_cidoc.html

4. Hyvönen, E., Mäkelä, E., Kauppinen, T., Alm, O., Kurki, J., Ruotsalo, T., Seppälä, K., Takala, J., Puputti, K., Kuittinen, H., Viljanen, K., Tuominen, J., Palonen, T., Frosterus, M., Sinkkilä, R., Paakkanen, P., Laitio, J., Nyberg, K.: CultureSampo: A National Publication System of Cultural Heritage on the Semantic Web 2.0. 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. 851–856. Springer, Heidelberg (2009)
5. Hyvönen, E., Palonen, T., Takala, J.: Narrative semantic web - Case National Finnish Epic Kalevala. In: Poster papers, Extended Semantic Web Conference, Heraklion, Greece (2010)
6. Wang, Y., Aroyo, L.M., Stash, N., Rutledge, L.: Interactive User Modeling for Personalized Access to Museum Collections: The Rijksmuseum Case Study. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511, pp. 385–389. Springer, Heidelberg (2007)
7. Wang, Y., Aroyo, L., Stash, N., et al.: Cultivating Personalized Museum Tours Online and On-site. *Interdisciplinary Science Reviews* 32(2), 141–156 (2009)
8. van Hage, W.R., Stash, N., Wang, Y., Aroyo, L.: Finding Your Way through the Rijksmuseum with an Adaptive Mobile Museum Guide. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010. LNCS, vol. 6088, pp. 46–59. Springer, Heidelberg (2010)
9. Lim, M.Y., Aylett, R.: Narrative Construction in a Mobile Tour Guide. In: Cavazza, M., Donikian, S. (eds.) ICVS-VirtStory 2007. LNCS, vol. 4871, pp. 51–62. Springer, Heidelberg (2007)
10. van Erp, M., Oomen, J., Segers, R., van den Akker, C., et al.: Automatic Heritage Metadata Enrichment with Historic Events. In: *Museums and the Web*, Philadelphia, PA (2011)
11. Shaw, R., Troncy, R., Hardman, L.: LOD: Linking Open Descriptions of Events. In: *Asian Semantic Web Conference*, pp. 153–167 (2009)
12. van Hage, W.R., Malaise, V., Segers, R., Hollink, L., Schreiber, G.: Design and use of the Simple Event Model (SEM). *Journal of Web Semantics* 9(2) (2011)
13. Scherp, A., Franz, T., Saathoff, C., Staab, S.: F—A Model of Events based on the Foundational Ontology DOLCE+DnSULtralite. In: *International Conference on Knowledge Capture*, pp. 137–144 (2009)
14. Waiboer, A.E.: Gabriel Metsu, Rediscovered Master of the Dutch Golden Age. National Gallery of Ireland (2010)
15. Arnold, B., Cass, B., Dorgan, T., et al.: *The Moderns - The Arts in Ireland from the 1900s to the 1970s*. Irish Museum of Modern Art (2011)
16. Maguire, M.: Museum practices report. Public DECIPHER Deliverable D2.1.1 (2011), <http://www.decipher-research.eu>
17. Mulholland, P., Wolff, A., Collins, T., Zdrahal, Z.: An event-based approach to describing and understanding museum narratives. In: *Detection, Representation, and Exploitation of Events in the Semantic Web. Workshop in Conjunction with the International Semantic Web Conference* (2011)
18. Hazel, P.: *Narrative and New Media*. In: *Narrative in Interactive Learning Environments*, Edinburgh, UK (2008)
19. Polkinghorne, D.: *Narrative knowing and the human sciences*. State Univ. NY Press (1988)
20. Chatman, S.: *Story and Discourse: Narrative structure in fiction and film*. Cornell U. (1980)
21. Corlosquet, S., Delbru, R., Clark, T., Polleres, A., Decker, S.: Produce and Consume Linked Data with Drupal! In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 763–778. Springer, Heidelberg (2009)

Bringing Mathematics to the Web of Data: The Case of the Mathematics Subject Classification^{*}

Christoph Lange^{1,2}, Patrick Ion^{3,4}, Anastasia Dimou⁴, Charalampos Bratsas⁴,
Wolfram Sperber⁵, Michael Kohlhase¹, and Ioannis Antoniou⁴

¹ Computer Science, Jacobs University Bremen, Germany

{ch.lange,m.kohlhase}@jacobs-university.de

² SFB TR/8, University of Bremen, Germany

³ Mathematical Reviews, American Mathematical Society
ion@ams.org

⁴ Web Science, Aristotle University Thessaloniki, Greece
{andimou,cbratsas,iantonio}@math.auth.gr

⁵ FIZ Karlsruhe, Germany
wolfram@zentralblatt-math.org

Abstract. The Mathematics Subject Classification (MSC), maintained by the American Mathematical Society’s Mathematical Reviews (MR) and FIZ Karlsruhe’s Zentralblatt für Mathematik (Zbl), is a scheme for classifying publications in mathematics. While it is widely used, its traditional, idiosyncratic conceptualization and representation did not encourage wide reuse on the Web, and it made the scheme hard to maintain. We have reimplemented its current version MSC2010 as a Linked Open Dataset using SKOS, and our focus is concentrated on turning it into the new MSC authority. This paper explains the motivation and details of our design considerations and how we realized them in the implementation, presents use cases, and future applications.

1 Introduction: The MSC and Its Applications

Classification schemes – “descriptive information for an arrangement or division of objects into groups based on characteristics, which the objects have in common” [\[9\]](#) – are in broad use in digital libraries. Due to the long history of library sciences, existing classification schemes cover a wide range of subjects of interest. Pre-existing subject schemes have proven to be a starting point for developing corresponding formal representations of domains in the form of ontologies.

There are general purpose classification schemes such as the Dewey Decimal System (DDC; see <http://www.oclc.org/dewey/> and p. [770](#)) and the Library of

^{*} We would like to thank Joseph Corneli from the PlanetMath encyclopedia for providing us with information on the PlanetMath access statistics. The first author has been supported by DFG project I1-[OntoSpace] of SFB/TR 8 “Spatial Cognition” and EPSRC grant “EP/J007498/1”. The second author thanks the Mathematics Department of the University of Michigan for its support.

Congress Subject Headings (LCSH; cf. sec. 7), as well as domain-specific ones such as ACM's Computing Classification System (CCS [23]) and the Physics and Astronomy Classification Scheme (PACS [17]). The Mathematics Subject Classification (MSC [14]) is the most common point of reference in mathematics. It has been used to classify mathematical documents of all types, ranging from lecture notes to journal articles and books. The MSC is maintained for the mathematical community by Mathematical Reviews (henceforth abbreviated as MR) and Zentralblatt Math (henceforth abbreviated as Zbl). The MSC is a three-layer scheme using alphanumeric codes, for example: **53** is the classification for Differential Geometry, **53A** for Classical Differential Geometry, and **53A45** for Vector and Tensor Analysis.

The present version MSC2010, released in January 2010, is now in production use at MR and Zbl. Fixes of simple factual and conceptual errors are still possible, whereas larger changes will be deferred to the next major revision to guarantee a period of stability to developers of applications and services.

Current Usage All major mathematical journals and digital libraries make use of the MSC. Examples range from the services of the AMS Mathematical Reviews, online as MathSciNet, and FIZ Karlsruhe's ZBMATH[1], through almost all the publishers of mathematics (Elsevier, Springer, etc.), to the arXiv.org pre-print server and the PlanetMath free encyclopedia [18]. The MSC is mainly used as a means of structuring mathematics literature in libraries and for the purposes of retrieving information by topic. For example, a recent analysis of the PlanetMath server logs[2] shows that accesses of PlanetMath's "browse by subject" pages[3], whose structure corresponds to the MSC2000, constitute 5 to 6 percent of all accesses of PlanetMath pages. Taking, furthermore, into account that these pages are much less often linked to from external sites (such as Wikipedia) and change less frequently, one can assume that they are less frequently visited by users coming from a search engine's results page and thus constitute a significant fraction of PlanetMath's "intra-site" traffic.

When an author, an editor, or a librarian classifies a publication, he or she typically identifies the right class(es) by consulting a human-readable version of the MSC. Web forms for creating a mathematical publication or uploading an existing one to a digital library typically require manual input of the MSC classes; the same holds for the search forms e.g. of MR or Zbl. Assistance is not provided – neither to authors, who could particularly benefit from an automatic suggestion of appropriate MSC classes based on the contents of an article, nor to users searching for articles, who could, e.g., benefit from the ability to select an MSC class without knowing its alphanumeric code, and from an automatic suggestion of related classes.

Maintenance and Revision So Far. The master source of the MSC has until recently been maintained in one plain \TeX file, using a set of custom macros

¹ <http://www.ams.org/mathscinet/> and <http://www.zentralblatt-math.org/zblmath/>

² Personal communication with Joseph Corneli from PlanetMath, 2011-10-31.

³ <http://planetmath.org/browse/objects/>

which have been developed around 1984, and had no major changes since then. The marked-up source of the example given above looks as follows:

```
\MajorSub 53-++\SubText Differential geometry
  \SeeFor{For differential topology, see \SbjNo 57Rxx.
  For foundational questions of differentiable manifolds, see \SbjNo 58Axx}
...
\SecndLvl 53Axx\SubText Classical differential geometry
...
\ThirdLvl 53A45\SubText Vector and tensor analysis
```

It is obvious that the $\text{T}_{\text{E}}\text{X}$ code is not useful for web-scale machine processing and linking. A new approach for web applications seems necessary for several reasons. Specialized subject classification schemes tend to be maintained by only a few experts in the arcane ways of the art, and the intellectual capital of a classification scheme does not necessarily become more obvious from merely reimplementing in a more standard format. However, the additional possibilities for accessing the scheme and producing different views tailored to specific audiences or purposes, which standard formats enable, may lead not only to wider adoption but even to better quality control. This is because more opportunities in distinguishing new aspects of the classification scheme arise, thus uncovering issues that may not have been identified by the few expert maintainers only.

The remainder of this section reviews the recent maintenance of the MSC implementation and points out problems we encountered. From 2006 to 2009 the MSC2000 version, then in current use, underwent a general revision, done publicly by the editorial staffs of MR and Zbl. This revision included additions and changes, and corrections of known errors and resulted in MSC2010. The editors took into consideration comments and suggestions from the mathematical public, of which there were on the order of a thousand recorded in a MySQL database. This was done using a standard installation of MediaWiki which was, and still is viewable by all, but was only editable by about 50 staff members. Each change from the previous version can be clearly seen (additions in green, deletions in red, on a yellow background)⁴.

Once the intellectual content had been finalized in this process, the new MSC2010 $\text{T}_{\text{E}}\text{X}$ master file in the format described above had to be produced, as well as derived and ancillary documents in various formats. These included: a table of changes, a KWIC index, PDF files for printing, as well as further variant forms useful to MR and Zbl. Furthermore, a TiddlyWiki edition (a single-user wiki in one HTML file; cf. <http://www.tiddlywiki.com>) was provided to enable users to download a personal copy of the MSC2010, which they could browse and annotate. The $\text{T}_{\text{E}}\text{X}$ master was obtained from the MediaWiki using a custom Python script; most of the derived files were constructed from the $\text{T}_{\text{E}}\text{X}$ master using custom Perl scripts. Obviously, all of these scripts were specific to the custom MSC $\text{T}_{\text{E}}\text{X}$ format; therefore, it would neither have been possible to reuse existing scripts from the maintenance of other subject classification schemes, nor will it be possible to use our scripts for a scheme other than the MSC.

⁴ see, for example, <http://msc2010.org/mscwiki/index.php?title=13-XX>

2 Requirements for a Reimplementation

The previous section outlined the current state of the MSC2010 and its limitations; an account of further problems beyond the scope of this paper can be found in [20]. To do better MR and Zbl decided to reimplement the MSC2010 after its release, i.e., after the mathematical domain knowledge had been settled. Concrete requirements and desirable goals for MSC2010 were:

1. The new implementation should **facilitate use and reuse** of the MSC:
 - (a) It should give convenient access to all the capabilities in the MSC that MR and Zbl depend on in producing their services and allow development.
 - (b) It should allow content providers, such as scientific publishers, to easily offer searching and querying publications using MSC classes in their digital libraries.
 - (c) It should enable making tools and interfaces, and reusing existing tools, that will help authors to classify their documents in a way that can easily be processed automatically.
2. The new implementation should **facilitate maintenance** of the MSC:
 - (a) It should be **complete** in that it **preserves all information** that was present in the existing implementation – preferably in a **semantically faithful** way, and leave room for **subsequent semantic refinements**.
 - (b) It should be maintainable with **standard tools**, minimizing the need for custom programming.
 - (c) It should enable a closer integration of maintenance-related information, e.g. changes from MSC2000 to MSC2010, which were previously recorded in separate XML files having a custom schema.
3. While the core concept scheme of the MSC is maintained through an editorial process, the new implementation should **enable knowledge workers and service developers** in mathematics and related fields to **adapt and extend** the MSC for their purposes:
 - (a) To connect mathematical subjects to related subjects in other domains (e.g. science).
 - (b) To enhance the MSC for their custom use cases, e.g. to add unofficial Greek class labels when using the MSC to structure a Greek lecture note repository
 - (c) However, such customizations should not affect the core scheme.
4. The new implementation should not only allow defining connections to related subjects, but it should **allow end users to explore such connections** and **discover new relevant knowledge**.

We chose RDF Linked Data, using the W3C-standardized SKOS vocabulary (Simple Knowledge Organization System [13]), for the reimplementation – hoping that our commitment to a standard will not only facilitate *our* maintenance but also foster wide adoption. Historically, the choice was motivated when a library asked for a MARC (Machine-Readable Cataloging) version of the MSC, and experts suggested to start with SKOS, as, e.g., the Library of Congress had done for its

Subject Headings (cf. sec. 7). Moreover, we knew we could rely on existing best-practice recommendations for modeling classification systems in SKOS, such as 16.

3 Design of the MSC/SKOS Concept Scheme

This section discusses design decisions we made while we were implementing the MSC2010 in SKOS, as to satisfy as many of the given requirements as possible. We roughly divide the different aspects of the MSC by the complexity of their representation in SKOS. This section focuses on structures that could be implemented in SKOS Core in a straightforward way, or by relatively straightforward extensions, the latter being implemented as an OWL ontology whose namespace we abbreviate with *mscvocab:*. In contrast, sec. 5 discusses points where we reached the limits of SKOS, or even of RDF.

The Basic Hierarchy (SKOS Core). The MSC presents a simple tree graph with 63 first-level nodes below the top root element, and 528 nodes as children of those, with 5606 final leaves. Implementing the basic concept hierarchy required a straightforward application of the following SKOS vocabulary terms:

- *skos:ConceptScheme* – for the whole concept scheme
- *skos:Concept* – for each MSC class. In contrast, the traditional terminology explicitly represented the level of a class in the hierarchy (“Major Subject”, “Second Level”, “Third Level”), but the level is deducible by the link structure expressed by *skos:narrower*.
- *skos:hasTopConcept* – for linking the scheme to the top level of the concept hierarchy
- *skos:narrower* – for links from the top level to the second level, and from the second level to the third level
- *skos:broader* – for backlinks in the opposite direction
- *skos:inScheme* – for backlinks from each class to the scheme

The following listing shows (in Turtle serialization) the SKOS implementation of these basic properties of the MSC class 53A45, plus two properties covered by the following subsections (notation and label):

```
msc2010:53A45 a skos:Concept; skos:inScheme msc2010:; skos:broader msc2010:53Axx;
  skos:prefLabel "Vector_and_tensor_analysis"@en ;
  skos:notation "53A45"^^mscsmpl:MSCNotation ;
```

The choice of appropriate URIs for the concepts required some more considerations and is therefore covered separately on p. 771.

Notations (SKOS Core) The 5-character class number (e.g. 53A45) could be represented as a notation⁵ (*skos:notation*), for which, for the purpose of enabling MSC-specific validation, we implemented our own datatype *mscvocab:MSCNotation*.

⁵ “a string of characters [...] used to uniquely identify a concept within the scope of a given concept scheme [which is] different from a lexical label in that a notation is not normally recognizable as a word or sequence of words in any natural language” 13

Multilingual Labels (SKOS Core). About each concept, the \TeX source provided as further information a descriptive English text (`\SubText` in the \TeX source), which could be represented as a preferred label, except that mathematical content requires separate treatment (see p. 768). Choosing SKOS allowed us to go beyond just representing the information given in the \TeX sources. Independently from the \TeX source, several trusted sources had contributed translations of the descriptive texts to further languages: Chinese, Italian, and Russian⁶. SKOS, thanks to its RDF foundation, not only facilitates handling accented characters (which also occur in the English-language descriptions) and non-Latin alphabets, but allows for multilingual labels, for example:

```
msc2010:53A45 skos:prefLabel "Vector_and_tensor_analysis"@en, "向量与张量分析"@zh .
```

We expect this to facilitate maintenance of the translated descriptions, as they are now part of the master source. Plus, RDF gives external developers speaking further languages the possibility to attach unofficial labels in, say, Greek, to the MSC/SKOS dataset by maintaining a separate graph containing triples such as

```
msc2010:53A45 skos:prefLabel "Διανυσματική και τανυστική ανάλυση"@el .
```

and then, for the desired application, merging it into the graph given by the official dataset.

Mathematical Markup in Labels (SKOS Core). The subject of mathematics involves formulas that need special markup and symbols, even within the descriptive labels of the MSC2010. Most of them, merely consisting of numbers, variable names or operator symbols, are sufficiently simple to be represented as plain Unicode text, but the semantics of mathematical expressions is often encoded in a two-dimensional layout and some labels make use of that. A detailed analysis of the \TeX source shows that 215 out of 6198 labels contain mathematical markup. While the real complexity of two-dimensional markup, e.g. fractions or matrices, does not occur in these labels, and while recent Unicode versions cover most mathematical symbols (including Latin and Greek letters in various scripts such as bold or italic, sub- and superscript digits, operators and other symbols), 23 labels remain that cannot be represented in Unicode. These include: expressions in a sub-/superscript (e.g., S^{n-1} or ${}_2F_1$), non-standard sub-/superscript letters (e.g., 1^k , H^p , or v_n), sub-/superscript symbols (e.g. C^∞), and overlined operators ($\overline{\partial}$). With MathML [3], whose recent inclusion in HTML5 is expected to lead to a more widespread adoption, there is an XML markup language that is capable of expressing such layout schemata, even with fallback alternative texts for applications that do not fully support MathML. RDF supports XML literals; these are literals of datatype *rdf:XMLLiteral*. Unfortunately, this approach is not compatible with multilingual labels, for reasons we will discuss on p. 773.

```
msc2010:26E10 skos:prefLabel "<math>\infty</math>">
<math>\infty</math><math>C</math><math>\infty</math><math>-</math>functions, quasi-analytic functions"^^rdf:XMLLiteral .
```

⁶ The sources were: Tsinghua University for Chinese, the Russian Academy of Sciences for Russian, and Alberto Marinari for Italian (MSC2000 only)

Linked Partitively Related Concepts (Extension). In addition to links to broader and narrower concepts, the \TeX source contained three further types of “see also” links from MSC concepts to other related MSC concepts. We introduced custom properties for each of these link types to capture their specific semantics. Note that these links are not symmetric; therefore, we did not make these properties subproperties of *skos:related*, but of a custom property *mscvocab:relatedPartOf*⁷ (to express that it is not an overall matching but a partitive relationship), which we declared a subproperty of the generic *skos:semanticRelation*. “See also” and “See mainly” could then be represented in a straightforward way, using the custom properties *mscvocab:seeAlso* and *mscvocab:seeMainly*. Note that it was easy to identify links having MSC classes as their targets, as such targets were preceded by $\backslash\text{SbjNo}$ (“subject number”) in the \TeX source. The trickier “See for” link represents conditional pointers, as can, e.g., be seen in the case of 53-+⁸ in the listing on p. 765. The relationship of a source class to a target class is not universally asserted but restricted to a certain aspect of the concept. As SKOS does not offer built-in support for such links, we chose a twofold approach of (1) establishing such links unconditionally for ease of traversing (but with another *mscvocab:relatedPartOf* subproperty to avoid confusion), and (2) to reify them into resources that point to their source and their target and carry the condition as a property, to fully capture their semantics:

```

msc:53-XX a skos:Concept ;                               msc:53-XXto57Rxx-seeFor
  mscvocab:seeConditionally msc:57Rxx ;                 mscvocab:forTarget msc:57Rxx ;
  mscvocab:seeFor                                         mscvocab:scope
  msc:53-XXto57Rxx-seeFor .                             "for_differential_topology" .

```

We introduce *mscvocab:scope* as a subproperty of the SKOS annotation property *skos:scopeNote*. For now, we chose this custom approach to reification, given that (1) RDF’s built-in reification support (assigning an ID to the triple *msc:53-XX msc:seeConditionally msc:57Rxx* and then stating further properties about the subject having that ID) is not recommended for use in Linked Datasets for lack of convenient SPARQL querying support [8, sec. 2.4] and is scheduled for deprecation in RDF 1.1⁹, and (2) other alternatives, such as Named Graphs [5], have not yet been standardized and are therefore not yet universally supported.

Our custom approach has the disadvantage of requiring both the direct link and the reified link to be expressed redundantly. However, the effort of manually expressing this can be saved by automatically inferring the direct link from its reified representation, taking advantage of the fact that the composition of *msc:seeFor* and *msc:forTarget* implies a conditional link. In fact we do so, using rules (cf. p. 772), or alternatively the OWL 2 axiomatization *mscvocab:seeFor* \circ *mscvocab:forTarget* \sqsubseteq *mscvocab:seeConditionally*.

⁷ Earlier SKOS versions included such properties in a “SKOS Extensions Vocabulary” (<http://www.w3.org/2004/02/skos/extensions/spec/2004-10-18.html>), which, however, has not been adopted as a standard so far.

⁸ The actual MSC code is 53-XX; it is encoded as 53-++ for historical reasons.

⁹ <http://www.w3.org/2011/01/rdf-wg-charter>

Linking Across MSC Versions and Other Concept Schemes (SKOS Core). While our main focus was on implementing the MSC2010 in SKOS, we also applied our $\text{T}_{\text{E}}\text{X}\rightarrow\text{SKOS}$ translation script (see sect. 4) to the older versions MSC2000 and MSC1991. Particularly the MSC2000 is still widely in use; therefore, making explicit how closely classes match across MSC versions will aid automated migration of existing digital libraries or at least be able to assist semi-automatic migration. SKOS offers a set of different properties to express the closeness of matching across subject classification schemes. Frequently occurring cases in the MSC include concepts **unchanged** across versions (\Rightarrow *skos:exactMatch*), **reclassifications** within an area, e.g. 05E40 “Combinatorial aspects of commutative algebra” partly replacing the MSC2000 classes 05E20 and 05E25 (\Rightarrow *skos:relatedMatch*), and **diversification** of areas, e.g. within the area 97-XX “Mathematics education”, which had 49 concepts in 2000 and 160 concepts in 2010 (\Rightarrow *skos:broadMatch*). While we have so far only used these mapping properties across MSC versions, SKOS implementations of further subject classification schemes in related domains are to be expected soon (cf. sec. 8). In this setting, these properties can be applied analogously.

Linking to non-SKOS Concepts (Extension). The adoption of SKOS as a W3C Recommendation and the increasing awareness of the potential benefits of Linked Open Data are good reasons to expect further classification schemes to become available as SKOS datasets in the near future (cf. sec. 8). However, many relevant schemes are not currently available in a full SKOS implementation. At <http://dewey.info>, for example, there is an experimental implementation covering the top three levels of the DDC, which includes the class 510 “Mathematics” and its 8 subclasses. For the MSC, however, more fine-grained mappings to the DDC Revision 21 have already been identified. For the time being, we represent them by incorporating local placeholders for the relevant DDC concepts into our implementation, and linking to them, for example:

```
msc:53A45 skos:relatedMatch [ a skos:Concept ; dcterms:isPartOf ddc:, msc: ;
    skos:notation "515.63"^^<http://dewey.info/schema-terms/Notation> ;
    skos:prefLabel "Vector, Tensor, Spinor Analysis" ] .
```

In this listing, the DDC concept appears as a blank node; in any case we refrained from assigning URIs in the DDC namespace to them. While the URI scheme for the deeper levels has already been decided upon (having URIs such as <http://dewey.info/class/515.63>), they are not currently dereferenceable.

Collections of Concepts Besides the Main Hierarchy (SKOS Core). Some of the links within the MSC do not have single classes as their targets, but groups of classes, which do not have a common superconcept that one could instead link to. The most frequently used group of such concepts is the group of all subclasses covering historical works related to an area. In the numeric scheme, these subclasses end in -03; for instance, 53-03 is the class of historical works about differential geometry. We have grouped them as *skos:members* of a *skos:Collection*, a

¹⁰ <http://oclc.org/developer/documentation/dewey-web-services/using-api>

semantically weaker notion than *skos:Concept* – but that choice demands awareness of the fact that SKOS keeps collections and concepts disjoint. Similar groupings include general reference works (-00), instructional expositions (-01), and works on computational methods (-08).

```
msc:HistoricalTopics a skos:Collection ;
  skos:prefLabel "Historical_topics"@en ;
  skos:member msc:01-XX, ..., msc:03-03, ..., msc:97-03 .
```

In addition, the MSC implicitly contains cross-area concepts such as “stability”, a property that a number of different mathematical structures may have [20]. The MSC classes related to stability are not currently systematically grouped; we just have the word “stability” to be found explicitly in the labels. Our implementation does not yet make such groupings explicit, but *skos:Collection* provides the necessary tools for doing so, after a careful conceptual analysis.

Co-Classification Policies. A further complication for modeling is introduced by the MSC specification prescribing that any resource classified with a -03 code (see above) be additionally classified with one class of the 01-XX section so as to express the specific historical aspect (e.g. “19th century” or “bibliography”). In the T_EX source, the descriptive label of each -03 class contained a remark about that. While our current implementation does not yet represent that policy in a fully machine-comprehensible way, SKOS allowed us to move forward in two regards: (1) We attached the information to the collection of historical topics, i.e. in one central place, to facilitate maintenance. On p. 772 we explain how the information can be propagated to the members of the collection. (2) The *skos:note* property allows keeping the information in a dedicated place, separate from the labels of concepts.

```
msc:HistoricalTopics skos:note "Any_resource_classified_as_-03_must_also
___be_assigned_at_least_one_classification_number_from_Section_01." .
```

URI Syntax. Deploying a Linked Dataset requires thinking about a URI syntax [8]. In the SKOS implementation described so far, the MSC2010 dataset has around 92,000 triples (in the expanded version; see below); the RDF/XML serialization is around 7 MB large. We expect that information about few MSC-classified resources will be required in typical Linked Data scenarios, such as looking up information about an MSC-classified resource. Publications in paper-based and digital libraries are typically classified with two MSC classes; in addition to these, the superclasses may be of interest. As such applications should not be burdened with a 7 MB download, a “hash” namespace does not make sense. Conversely, applications that require full access to the MSC, such as annotation services that suggest MSC classes whose labels match a given text (as shown in fig. 11), or browser frontends to digital libraries, would rather benefit from querying a SPARQL endpoint, or their developers would preload them with a downloaded copy of the MSC dataset anyway – a possibility that is independent from the choice of namespace URI. Thus, we chose

<http://msc2010.org/resources/MSC/2010/> as namespace URI for the MSC2010. For the older MSC versions, the last path components contain the respective years. The MSC-specific SKOS extension vocabulary and the MSC-specific datatype library reside in separate namespaces, as they are conceptually separate from the MSC classes and as we expect different (slower) maintenance cycles for them.

4 Deployment and Publication

We generated the new SKOS master source of the MSC2010 by a script (described below) in an iterative process while deciding on the design issues detailed above. After that, we published the data in four complementary ways, aiming to address a large audience of users and developers and enabling them to link their data to the MSC and to use the MSC in their services. All publications are available from the project homepage <http://msc2010.org/mscwork/>.

We implemented the script for the original translation of the old \TeX master source of the MSC2010 to the new SKOS master source (one RDF/XML file) in Perl. It has now served its purpose and was never prepared or intended to be applicable to any other source representation of a classification scheme.

For querying the dataset, we expose it through a **SPARQL endpoint**. The **MSC Linked Wiki** frontend aims at providing easy and user-friendly navigation through the MSC. It uses SPARQL queries to the endpoint to present the classification on its pages.

We maintain **different RDF versions for different demands**. For ease of maintenance, the SKOS master source is restricted to a semantic core of RDF triples that avoids redundancy (e.g. only modeling the *skos:narrower* direction of the hierarchy). When an OWL reasoner is available, the *skos:broader* direction can be inferred automatically; however: In a Linked Data setting, where clients hop through the dataset from resource to resource in a “follow-your-nose” manner [21], it is essential to provide as many explicit links as possible. Secondly, inference support may not always be available in applications, depending on their scalability-related performance constraints.

Therefore, we have implemented an automatic expansion of the core dataset to an enriched “convenience” version, which we expose through the SPARQL endpoint and as Linked Open Data (LOD). So far, the latter is served from static RDF/XML files (one per MSC class), into which we split the one-file enriched version, but we are planning to serve the dataset through a SPARQL endpoint at <http://msc2010.org>. Additionally, we offer both versions of the dataset for download, so that application developers can import them into their triple stores. We have implemented the expansion as N3 rules. The following rule, for example, infers *skos:broader* from *skos:narrower*, effectively hard-coding the semantics of *owl:inverseOf*:

```
{ ?conc skos:narrower ?narrowerConc } => { ?narrowerConc skos:broader ?conc }.
```

Further rules infer *skos:topConceptOf* from *skos:hasTopConcept*, generate backlinks from reified “see for” links to their sources, un-reify the “see for” links into

mscvocab:seeConditionally, dumb down all MSC-specific links to *skos:semanticRelation*, and further dumb down *skos:semanticRelation* to *rdfs:seeAlso* for off-the-shelf linked data browsers. These rules can be applied to the core dataset using an N3 reasoner such as *cwm* [2], which increases the number of triples by more than 16% (from 79,000 core triples to 92,000).

```
cwm --rdf msc2010-core.skos --n3 expand-skos-rules.n3 --think
```

5 Benefits Experienced and Difficulties Encountered

SKOS was designed to be simple and thus powerful in deployment. Its authors thought of it as much simpler than full OWL and very suitable for such cases of knowledge organization as thesauri. However, in the research described here, SKOS had to pass the reality check of a large classification scheme that had grown up in a specialized field over a long time. This section summarizes the benefits we experienced and the difficulties we encountered.

Relying on SKOS satisfies the requirements to “facilitate use, reuse, and maintenance” in that RDF in general and SKOS in particular enjoy wide tool support. There are tools for searching and querying, for editing, for consistency checking, and for annotating documents; for an overview, see <http://www.w3.org/2001/sw/wiki/SKOS> and [19]. Concerning reuse, the Linked Data principles provide a straightforward way of making SKOS/RDF accessible on the Web not only for browsing and for download, but also as a target for linking. Furthermore, we expect the wide availability of RDF parsers to facilitate implementing conversions into forms that are used by library management systems. In regard to maintenance, SKOS proved able to capture large parts of the structural semantics of the MSC. In particular, it supports maintaining links to other concept schemes and translations together with the core scheme.

Our implementation is complete in that it preserves all information that was present in the old T_EX source – often making the semantics more explicit and thus more easily accessible to automated processing. Making the semantics explicit was partly supported by SKOS itself, but most of it had to be done by extensions – mostly using extension points SKOS or RDF provided for.

Multilingual Labels vs. Mathematical Markup. One particular problem remains, for which neither SKOS nor RDF provided a sufficient solution. On p. 768 we pointed out the importance of formulas in labels in the mathematical domain. XML literals using MathML seem to be the solution, but for the following reasons they conflict with multilingual labels, which are also highly relevant in the MSC setting: (1) The SKOS recommendation states that “by convention, *skos:prefLabel* [is] only used with plain literals” [13, sec. 6.5.4], i.e. with non-datatype literals. (2) Datatype literals may not carry a language tag. The language of an XML literal may be indicated *inside* the XML, e.g. by enclosing the whole literal into an element that carries an XML language tag (e.g. `<element xml:lang="en">`) – but these are not part of the RDF graph and therefore not accessible from SPARQL queries. (3) An English and a Greek

skos:prefLabel with mathematical markup for the same MSC class, marked up as shown in (2), would count as two *skos:prefLabels* without an (RDF) language tag. While that would not explicitly violate SKOS integrity condition S14, which demands that “a resource has no more than one value of *skos:prefLabel* per *language tag*”, it would contradict convention (1), leaving little hope for tool support. Carroll and Philipps proposed an extension to the RDF semantics in [4] that would allow for indicating the language of an XML literal in 2005, but that idea has never been adopted. Therefore, our current SKOS implementation of the MSC2010 leaves this problem unsolved for now. Note that separating the mathematical expressions in labels from the surrounding text would not qualify as a workaround, as (1) expressions can be scattered over multiple places in a text sentence, e.g. in the role of an adjective qualifying a noun, and as (2) the structure and presentation of mathematical expressions may vary depending on the language – not in the concrete case of the MSC labels but in general.

6 Use Case: The Linked Universities Initiative

This section presents a use case for the Linked Data implementation of the MSC, highlighting its relevance for electronic publishing and education. *Linked Universities* (<http://linkeduniversities.org>) is an alliance of European universities engaged in exposing their public data as linked data. The School of Mathematics at Aristotle University Thessaloniki (AUTH), in conjunction with “Semantic AUTH”, AUTH’s contribution to the Linked Universities initiative, has one such semantic portal at <http://www.math.auth.gr>. The courses offered in the school are semantically annotated using appropriate ontologies such as the Academic Institution Internal Structure Ontology (AIISO), Bowlogna and Bibliographic Ontology (BIBLIO), and published according to the Linked Data principles. Furthermore, the scientific fields covered by courses, as well as the faculty’s research interests, are annotated using MSC/SKOS. They will also include references to other Linked Data entities inside or outside the website.

7 Related Work

Besides following the best practices established for SKOSifying the DDC [16], our work was inspired from the LCSH dataset [22]. Both are comprehensive, general-purpose classification schemes in contrast to the domain-specific MSC. The LCSH was converted from a MARCXML representation to SKOS, using custom scripts similar to ours. Similar to our approach, the authors developed custom extensions to SKOS (e.g. structured change descriptions similar Panzer’s and Zeng’s, which we reused), and finally evolved them into the MADS/RDF data model, which can be thought of a superset of SKOS “designed specifically to support authority data as used by and needed in the LIS [library and information science] community and its technology systems” [12]. They also experienced limitations of SKOS, concretely concerning the representation of “pre-coordinated concepts”, i.e. subject headings combined from other headings. While our Web

Section of Computer Science and Numerical Analysis

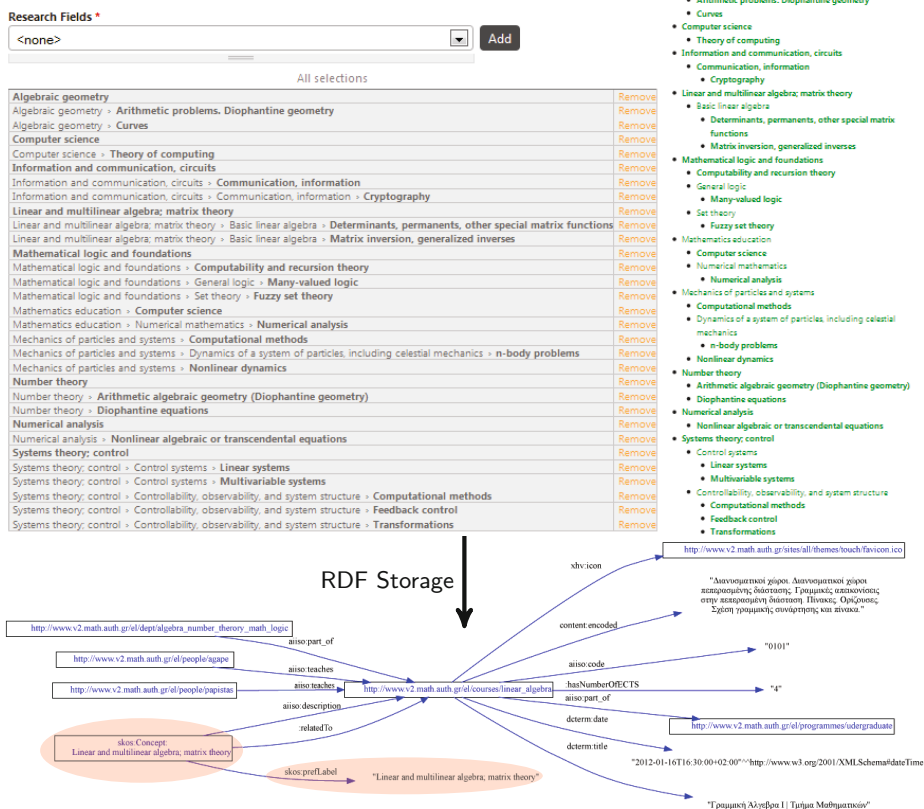


Fig. 1. Annotating the scientific fields of a course on the new AUTH School of Mathematics site, using MSC/SKOS and Drupal 7 semantic mappings (cf. [6])

frontend to the MSC is in an early stage, the LCSH dataset is served via a comprehensive frontend that offers each record for download in different formats (including full MADS/RDF vs. plain SKOS), a graph visualization, and a form for reporting errors. Limitations of SKOS and the possibility of extending SKOS have also been reported for domain-specific classification schemes; see, e.g., van Assem’s case studies with three different thesauri [1]. In domains closely related to mathematics, we are not aware of completed SKOS implementations, but of work in progress, for example on the ACM CCS [23] for computer science [11].

8 Conclusion; Roadmap Towards a Math. Web of Data

With this work we delivered the first complete LOD implementation of the MSC. This brought us closer to satisfying our original requirements, and the rigorous

¹¹ Personal comm. with Bernard Rous, ACM Director of Publications, 2011-06-08.

conceptual modeling approach helped to uncover new issues in the MSC conceptualization. While this paper focuses on preserving all information from the previous \TeX master sources (plus translated labels), we have also, in previous work [20], identified directions for *enhancing* the conceptual model by precise *definitions* of the MSC classes, adding index terms to classes (for which Panzer and Zeng provide a SKOS design pattern [16]) and a *faceted structure* (which the collections introduced on p. 770 only partly address).

The MSC/SKOS dataset is also one of the first Linked Datasets in mathematics. Our previous work has laid the conceptual and technical foundations for integrating mathematics into the Web of Data [11]; we believe that the availability of the central classification scheme of this domain as LOD will encourage further progress. Deploying the MSC as LOD makes it more easily reusable and enables classification of smaller resources of mathematical knowledge (e.g. blog posts, or figures or formulas in larger publications), instead of the traditional approach of assigning few MSC classes to a whole article. For a closer integration of mathematical resources with those from related domains, we plan to establish links from and to the ACM CCS [23], once available in SKOS, and with the PACS [17], which we expect to reimplement ourselves. As further deployment targets, we envision the European Digital Math Library [7], whose developers are starting to work on Linked Data publishing, as well as the PlanetMath encyclopedia, which is being reimplemented using the Planetary social semantic web portal [10]. With this deployment strategy and the increased ability to classify fine-grained mathematical resources over the Web, we also believe that the MSC/SKOS dataset may support a democratization of scientific publishing, and, by taking away some of the control from the big publishing companies and giving it back to the authors, encourage the rise of networked science that depends on collaborative intelligences [15].

References

- [1] van Assem, M.F.J.: Converting and Integrating Vocabularies for the Semantic Web. PhD thesis, Vrije Universiteit Amsterdam (2010), <http://hdl.handle.net/1871/16148>
- [2] Berners-Lee, T.: Cwm. A general purpose data processor for the semantic web (2009), <http://www.w3.org/2000/10/swap/doc/cwm.html>
- [3] Mathematical Markup Language (MathML) 3.0. W3C Recommendation (2010), <http://www.w3.org/TR/MathML3>
- [4] Carroll, J.J., Phillips, A.: Multilingual RDF and OWL. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 108–122. Springer, Heidelberg (2005)
- [5] Carroll, J.J., et al.: Named Graphs, Provenance and Trust. In: WWW, pp. 613–622. ACM (2005)
- [6] Corlosquet, S., Delbru, R., Clark, T., Polleres, A., Decker, S.: Produce and Consume Linked Data with Drupal! In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 763–778. Springer, Heidelberg (2009)
- [7] EuDML – European Digital Mathematics Library, <http://eudml.eu>

- [8] Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Morgan & Claypool (2011), <http://linkeddatabook.com>
- [9] Information technology – Metadata registries (MDR) – Part 1: Framework. Tech. Rep. 11179-1, ISO/IEC (2004)
- [10] Kohlhase, M., et al.: The Planetary System: Web 3.0 & Active Documents for STEM. *Procedia Computer Science* 4, 598–607 (2011), <https://svn.mathweb.org/repos/planetary/doc/epc11/paper.pdf>
- [11] Lange, C.: Enabling Collaboration on Semiformal Mathematical Knowledge by Semantic Web Integration. *Studies on the Semantic Web*, vol. 11. IOS Press, Amsterdam (2011)
- [12] MADS/RDF primer. Status: Final Public Review Document (2011), <http://www.loc.gov/standards/mads/rdf/>
- [13] Miles, A., Bechhofer, S.: SKOS Simple Knowledge Organization System Reference. W3C Recommendation (2009), <http://www.w3.org/TR/skos-reference>
- [14] MSC 2010 (2010), <http://msc2010.org>
- [15] Nielsen, M.: *Reinventing Discovery: The New Era of Networked Science*. Princeton University Press (2011)
- [16] Panzer, M., Zeng, M.L.: Modeling Classification Systems in SKOS: Some Challenges and Best-Practice Recommendations. In: *International Conference on Dublin Core and Metadata Applications* (2009), <http://dcpapers.dublincore.org/index.php/pubs/article/view/974/0>
- [17] Physics and Astronomy Classification Scheme, PACS (2010), <http://aip.org/pacs/>
- [18] PlanetMath.org, <http://planetmath.org>
- [19] Solomou, G., Papatheodorou, T.: The Use of SKOS Vocabularies in Digital Repositories: The DSpace Case. In: *Semantic Computing (ICSC)*, pp. 542–547. IEEE (2010)
- [20] Sperber, W., Ion, P.: Content analysis and classification in mathematics. In: *Classification & Ontology, Intern. UDC Seminar*, pp. 129–144
- [21] Summers, E.: Following your nose to the Web of Data. *Information Standards Quarterly* 20(1) (2008), <http://inkdroid.org/journal/following-your-nose-to-the-web-of-data>
- [22] Summers, E., et al.: LCSH, SKOS and Linked Data. In: *Dublin Core* (2008), arXiv:0805.2855v3 [cs.DL]
- [23] The 1998 ACM Computing Classification System (1998), <http://www.acm.org/about/class/ccs98>

A Publishing Pipeline for Linked Government Data

Fadi Maali¹, Richard Cyganiak¹, and Vassilios Peristeras²

¹ Digital Enterprise Research Institute, NUI Galway, Ireland
{fadi.maali,richard.cyganiak}@deri.org

² European Commission, Interoperability Solutions for European Public Administrations
vassilios.peristeras@ec.europa.eu

Abstract. We tackle the challenges involved in converting raw government data into high-quality Linked Government Data (LGD). Our approach is centred around the idea of *self-service LGD* which shifts the burden of Linked Data conversion towards the data consumer. The self-service LGD is supported by a publishing pipeline that also enables sharing the results with sufficient provenance information. We describe how the publishing pipeline was applied to a local government catalogue in Ireland resulting in a significant amount of Linked Data published.

1 Introduction

Open data is an important part of the recent open government movement which aims towards more openness, transparency and efficiency in government. Government data catalogues, such as data.gov and data.gov.uk, constitute a corner stone in this movement as they serve as central one-stop portals where datasets can be found and accessed. However, working with this data can still be a challenge; often it is provided in a haphazard way, driven by practicalities within the producing government agency, and not by the needs of the information user. Formats are often inconvenient, (e.g. numerical tables as PDFs), there is little consistency across datasets, and documentation is often poor [6].

Linked Government Data (LGD) [2] is a promising technique to enable more efficient access to government data. LGD makes the data part of the web where it can be interlinked to other data that provides documentation, additional context or necessary background information. However, realizing this potential is costly. The pioneering LGD efforts in the U.S. and U.K. have shown that creating high-quality Linked Data from raw data files requires considerable investment into reverse-engineering, documenting data elements, data clean-up, schema mapping, and instance matching [8,16]. When data.gov started publishing RDF, large numbers of datasets were converted using a simple automatic algorithm, without much curation effort, which limits the practical value of the resulting RDF. In the U.K., RDF datasets published around data.gov.uk are carefully curated and of high quality, but due to limited availability of trained staff and contractors, only selected high-value datasets have been subjected to the Linked

Data treatment, while most data remains in raw form. In general, the Semantic Web standards are mature and powerful, but there is still a lack of practical approaches and patterns for the publishing of government data [16].

In a previous work, we presented a contribution towards supporting the production of high-quality LGD, the “self-service” approach [6]. It shifts the burden of Linked Data conversion towards the data consumer. We pursued this work to refine the self-service approach, fill in the missing pieces and realize the vision via a working implementation.

The Case for “Self-service LGD”

In a nutshell, the self-service approach enables consumers who need a Linked Data representation of a raw government dataset to produce the Linked Data themselves without waiting for the government to do so. Shifting the burden of Linked Data conversion towards the data consumer has several advantages [6]: (i) there are more of them; (ii) they have the necessary motivation for performing conversion and clean-up; (iii) they know which datasets they need, and don’t have to rely on the government’s data team to convert the right datasets.

It is worth mentioning that a self-service approach is aligned with civic-sourcing, a particular type of “crowd sourcing” being adopted as part of Government 2.0 to harness the wisdom of citizens [15].

Realizing the Self-service LGD

Working with the authoritative government data in a crowd-sourcing manner further emphasizes managing the tensioned balance between being easy to use and assuring quality results. A proper solution should enable producing useful results rather than merely “triple collection” and still be accessible to non-expert users. We argue that the following requirements are essential to realize the self-service approach:

Interactive approach it is vital that users have full control over the transformation process from cleaning and tidying up the raw data to controlling the shape and characteristics of the resulting RDF data. Full automatic approaches do not always guarantee good results, therefore human intervention, input and control are required.

Graphical user interface easy-to-use tools are essential to making the process swift, less demanding and approachable by non-expert users.

Reproducibility and traceability authoritative nature of government data is one of its main characteristics. Cleaning-up and converting the data, especially if done by a third party, might compromise this authoritative nature and adversely affect the data perceived value. To alleviate this, the original source of the data should be made clear along with full description of all the operations that were applied to the data. A determined user should be able to examine and re-produce all these operations starting from the original data and ending with an exact copy of the published converted data.

Flexibility the provided solution should not enforce a rigid workflow on the user. Components, tools and models should be independent from each other, yet working well together to fit in a specific workflow adopted by the user.

Decentralization there should be no requirement to register in a centralized repository, to use a single service or to coordinate with others.

Results sharing it should be possible to easily share results with others to avoid duplicating work and efforts.

In this paper, we describe how we addressed these requirements through the “LGD Publishing Pipeline”. Furthermore, we report on a case study in which the pipeline was applied to publish the content of a local government catalogue in Ireland as Linked Data.

The contributions of this paper are:

1. An end-to-end publishing pipeline implementing the self-service approach. The publishing pipeline, centred around Google Refine¹, enables converting raw data available on government catalogues into interlinked RDF (section 2). The pipeline also enables sharing the results along with their provenance description on CKAN.net, a popular open data registry (section 2.5).
2. A formal machine-readable representation of full provenance information associated with the publishing pipeline. The LGD Publishing Pipeline is capable of capturing the provenance information, formally representing it according to the Open Provenance Model Vocabulary (OPMV)² and sharing it along with the data on CKAN.net (section 2.5).
3. A case study applying the publishing pipeline to a local government catalogue in Ireland. The resulting RDF, published as linked data as part of data-gov.ie, is linked to existing data in the LOD cloud. A number of widely-used vocabularies in the Linked Data community — such as VoID³, OPMV and Data Cube Vocabulary⁴ — were utilised in the data representation. The intermix of these vocabularies enriches the data and enables powerful scenarios (section 3).

2 LGD Publishing Pipeline

The LGD Publishing Pipeline is outlined in figure 1. The proposed pipeline, governed by the requirements listed in the previous section, is in line with the process described in the seminal tutorial “How to publish Linked Data?”⁴ and with various practices reported in literature^{7,11}.

We based the pipeline on Google Refine, a data workbench that has powerful capabilities for data massaging and tidying up. We extended Google Refine with Linked Data capabilities and enabled direct connection to government catalogues from within Google Refine. By adopting Google Refine as the basis of the pipeline we gain the following benefits:

¹ <http://code.google.com/p/google-refine/>

² <http://code.google.com/p/opmv/>

³ <http://www.w3.org/TR/void/>

⁴ <http://bit.ly/data-cube-vocabulary>

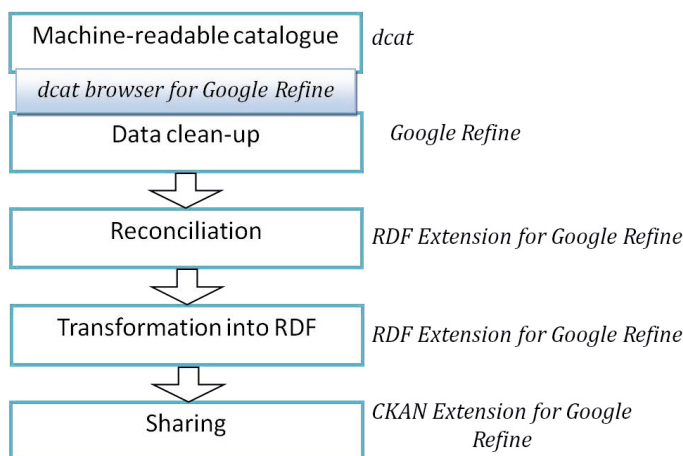


Fig. 1. Linked Data publishing pipeline (pertinent tool is shown next to each step)

- Powerful data editing, transforming and enriching capabilities.
- Rich import capabilities e.g. JSON, Excel, CSV, TSV, etc.
- Support of full and persistent undo/redo history.
- Popular in open data community.
- Extensible and under active development.
- Free and open source.

All the involved functionalities are available through a single workbench which not only supports transforming raw data into RDF; but also enables interlinking the data, capturing and formally representing all the applied operations (i.e. provenance information). The steps involved are independent from each other, yet seamlessly integrated from the user point of view. In the following subsections, we describe the involved steps outlined in figure 1.

2.1 Machine Readable Catalogues

Increasingly, governments are maintaining data catalogues listing the datasets they share with the public⁵. These catalogues play a vital role in enhancing the visibility and findability of the government datasets. However, catalogues' data is often only available through the catalogues web sites. Even when catalogues make their data available in a machine-readable format, they still use proprietary APIs and data formats. This heterogeneity hinders any effort to build tools that fully utilise and reliably access the available data.

We developed *Dcat*, an RDF vocabulary to represent government data catalogues [13]. *Dcat* defines terms to describe catalogues, datasets and their distributions (i.e. accessible forms through files, web services, etc.).

⁵ <http://datacatalogs.org/> lists 200 catalogues as of 06/12/2011.

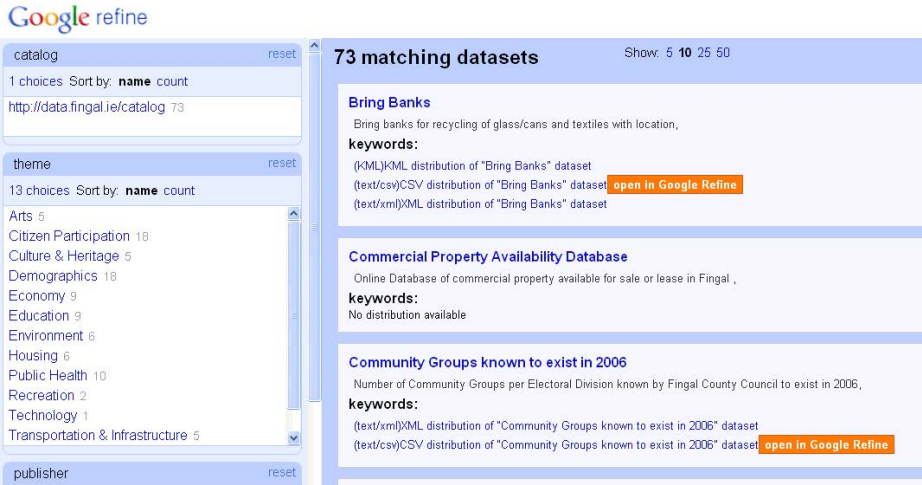


Fig. 2. Dcat Browser - navigating catalogues from within Google Refine

Dcat has been adopted by a number of government catalogues. Prominent examples of Dcat adopters include data.gov.uk and semantic.ckan.net. Currently, Dcat development is pursued under the W3C Government Linked Data Working Group⁶. Therefore, a growing adoption of it is plausible.

Our first extension to Google Refine, Dcat Browser, utilises Dcat to enable browsing government catalogues from within Google Refine. Feeding the Dcat Browser with Dcat data, via a SPARQL endpoint URL or an RDF dump, results in a faceted browser of the available datasets (figure 2). Datasets that have distributions understandable by Google Refine (e.g. CSV, Excel, TSV, etc.) can be directly opened as Google Refine project. The extension takes care of fetching files and opening them in Google Refine. Imported files can then be scrutinized and subjected to all Google Refine editing and transformation functionalities.

2.2 Data Clean-up

A stage of data preparation is necessary to fix errors, remove duplicates and prepare for transformation. Google Refine has powerful data cleaning and transformation capabilities. It also has an expressive expression language called GREL. The built-in clustering engine facilitates identifying duplicates. Additionally, facets, which are at the heart of Google Refine, help understanding the data and getting it into a proper shape before converting to RDF⁷.

⁶ http://www.w3.org/egov/wiki/Data_Catalog_Vocabulary

⁷ Full documentation of Google Refine is available at:

<http://code.google.com/p/google-refine/wiki/DocumentationForUsers>

2.3 Converting Raw Data into RDF

We developed the RDF Extension for Google Refine⁸ to enable modelling and exporting tabular data in RDF format. The conversion of tabular data into RDF is guided through a template graph defined by the user. The template graph nodes represent resources, literals or blank nodes while edges are RDF properties (see figure 3). Nodes values are either constants or expressions based on the cells contents. Every row in the dataset generates a subgraph according to the template graph, and the whole RDF graph produced is the result of merging all rows subgraphs. Expressions that produce errors or evaluate to empty strings are ignored.

The main features of the extension are highlighted below (interested readers are encouraged to check 12):

- RDF-centric mapping: From information integration point of view, mapping can be source-centric or target-centric. In our case it can be spreadsheet-centric or RDF-centric, respectively. RDF Extension uses the RDF-centric approach i.e. the translation process will be described in terms of the intended RDF data. RDF-centric is more expressive than the spreadsheet-centric approach 11. Furthermore, it is closer to the conceptual model of the data rather than the representation model as expressed in the particular tabular structure of the spreadsheet.
- Expression language for custom expressions: Google Refine Expression Language GREL is used for defining custom values. GREL uses intuitive syntax and comes with a fairly rich set of functions. It also supports if-else expressions, which means that the exported RDF data can be customised based on cells' content (e.g. defining different classes based on cell content).
- Vocabularies/ontologies support: defining namespace prefixes and basic vocabulary management (add, delete and update) are supported. The RDF Extension is able to import vocabularies available on the web regardless of the format used (e.g. RDFa, RDF/XML and Turtle) as long as their deployment is compatible with the best practices recommended by the W3C in 3. This makes it easier to reuse existing vocabularies. Such reuse not only saves effort and time but also assures that the data is more usable and not isolated. When no existing terms are suitable, users can forge their own.
- Graphical User Interface (GUI): The design of the template graph –the graph that defines the mapping– is supported by a graphical user interface where the graph is displayed as a node-link diagram. Autocomplete support for imported ontologies is also provided.
- Debugging: instant preview of the resulting RDF data is provided to enable quick debugging of the mapping. The preview is the RDF data generated from the first ten rows and serialised in Turtle syntax⁹. Turtle syntax is chosen because of its readability and compactness.

⁸ <http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/>

⁹ <http://www.w3.org/TR/turtle/>

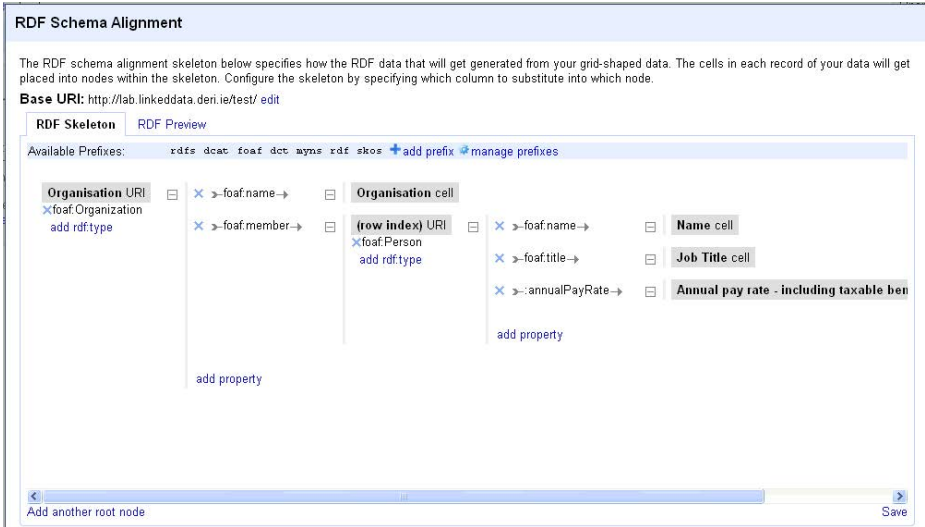


Fig. 3. RDF Extension user interface - graph template design

It is worth mentioning that in addition to the graphical representation of the mapping, users are able to access a text-based representation that can be reused, exchanged or directly edited by advanced users.

2.4 Interlinking

Linking across dataset boundaries turns the Web of Linked Data from a collection of data silos into a global data space [5]. RDF Links are established by using the same URIs across multiple datasets.

Google Refine supports data reconciliation i.e. matching a project's data against some external reference dataset. It comes with a built-in support to reconcile data against Freebase. Additional reconciliation services can be added via implementing a standard interface¹⁰. We extended Google Refine to reconcile against any RDF data available through a SPARQL endpoint or as a dump file. Reconciling against an RDF dataset makes URIs defined in that dataset usable in the RDF export process. As a result, interlinking is integrated as part of the publishing pipeline and enabled with a few clicks.

For example, to reconcile country names listed as part of a tabular data against DBpedia all is needed is providing Google Refine with DBpedia SPARQL endpoint URL. The reconciliation capability of the RDF Extension, will match the country names against labels in DBpedia. Restricting matching by type and adjacent properties (i.e. RDF graph neighbourhood) is also supported. In [14] we provided the full details and evaluated different matching approaches.

¹⁰ <http://code.google.com/p/google-refine/wiki/ReconciliationServiceApi>

2.5 Sharing

The last step in the LGD Publishing Pipeline is sharing the RDF data so that others can reuse it. However, the authoritative nature of government data increases the importance of sharing a clear description of all the operations applied to the data. Ideally, provenance information is shared in a machine-readable format with a well-defined semantics to enable not only human users but also programs to access the information, process and utilise it.

We developed “CKAN Extension for Google Refine”^[11] that captures the operations applied to the data, represents them according to the Open Provenance Model Vocabulary (OPMV) and enables sharing the data and its provenance on [CKAN.net](http://ckan.net).

OPMV is a lightweight provenance vocabulary based on OPM [18]. It is used by data.gov.uk to track provenance of data published by the U.K. government. The core ontology of OPMV can be extended by defining supplementary modules. We defined an OPMV extension module to describe Google Refine *workflow* provenance in a machine-readable format. The extension is based on another OPMV extension developed by Jeni Tennison^[12]. It is available and documented online at its namespace: <http://vocab.deri.ie/grefine#>

Google Refine logs all the operations applied to the data. It explicitly represents these operations in JSON and enables extracting and (re)applying them. The RDF related operations added to Google Refine are no exception. Both the RDF modelling and reconciling are recorded and saved in the project history. The JSON representation of the history in Google Refine is a full record of the information provenance. The extension OPMV module enables linking together the RDF data, the source data and the Google Refine operation history. Figure 4 shows an example representation of the provenance of RDF data exported using Google Refine RDF Extension. In the figure `ex:rdf_file` is an RDF file derived from `ex:csv_file` by applying operations represented in `ex:json_history_file`.

Lastly, we enabled sharing the data on CKAN.net from within Google Refine with a few clicks. CKAN.net is an “open data hub” i.e. a registry where people can publicly share datasets by registering them along with their metadata and access information. CKAN.net can be seen as a platform for crowdsourcing a comprehensive list of available datasets. It enjoys an active community that is constantly improving and maintaining dataset descriptions. CKAN Storage^[13], a recent extension of CKAN, allows files to be uploaded to and hosted by CKAN.net.

A typical workflow for a CKAN contributor who wants to share the results of transforming data into RDF using Google Refine might be: (i) exporting the data from Google Refine in CSV and in RDF (ii) extracting and saving Google Refine operation history (iii) preparing the provenance description (iv) uploading the files to CKAN Storage and keeping track of the files URLs (v)

¹¹ <http://lab.linkeddata.deri.ie/2011/grefine-ckan>

¹² <http://purl.org/net/opmv/types/google-refine#>

¹³ <http://ckan.org/2011/05/16/storage-extension-for-ckan/>

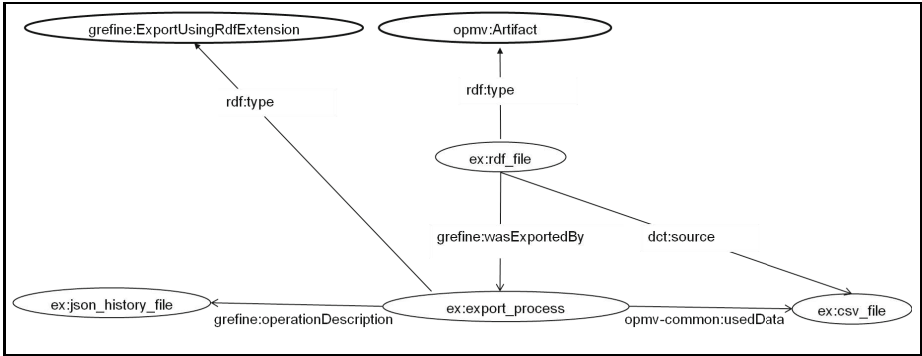


Fig. 4. RDF representation of provenance information of Google Refine RDF

updating the corresponding package on CKAN.net. CKAN Extension for Google Refine automates this tedious process to save time and efforts and to reduce errors (figure 5). In addition to uploading the files, the extension updates CKAN through its API accordingly by registering a new package or updating an existing one. The data uploaded from Google Refine can be any combination of the CSV data, RDF data, provenance description and Google Refine JSON representation of operations history.

Having the data on CKAN means that it is available online for others to use, its description can be enhanced and it can be programmatically accessed using CKAN API. Multiple RDF representations of a specific dataset can co-exist and the community aspects of CKAN.net, such as rating and tagging, can be harnessed to promote the best and spread good practices in RDF conversion.

3 Case Study - Fingal County Catalogue

Fingal is an administrative county in the Republic of Ireland. Its population is 239,992 according to the 2006 census¹⁴. Fingal County Council, the local authority for Fingal, is one of the four councils in the Dublin Region. It is the first council to run an open data catalogue in the Republic of Ireland.

Fingal Open Data Catalogue, available at <http://data.fingal.ie/>, enables free access to structured data relating to Fingal County. It aims to foster participation, collaboration and transparency in the county. Catalogue's datasets cover various domains from demographics to education and citizen participation. Most datasets are published by Fingal County Council and the Central Statistics Office in Ireland. Datasets are available, under Ireland PSI general license¹⁵, in open formats such as CSV, XML and KML. In the light of Sir Tim Berners-Lee's star scheme¹⁶, Fingal Catalogue is a 3-star one.

¹⁴ <http://beyond2020.cso.ie/Census/TableViewer/tableView.aspx?ReportId=75467>

¹⁵ <http://data.fingal.ie/licence/licence.pdf>

¹⁶ <http://lab.linkeddata.deri.ie/2010/star-scheme-by-example/>

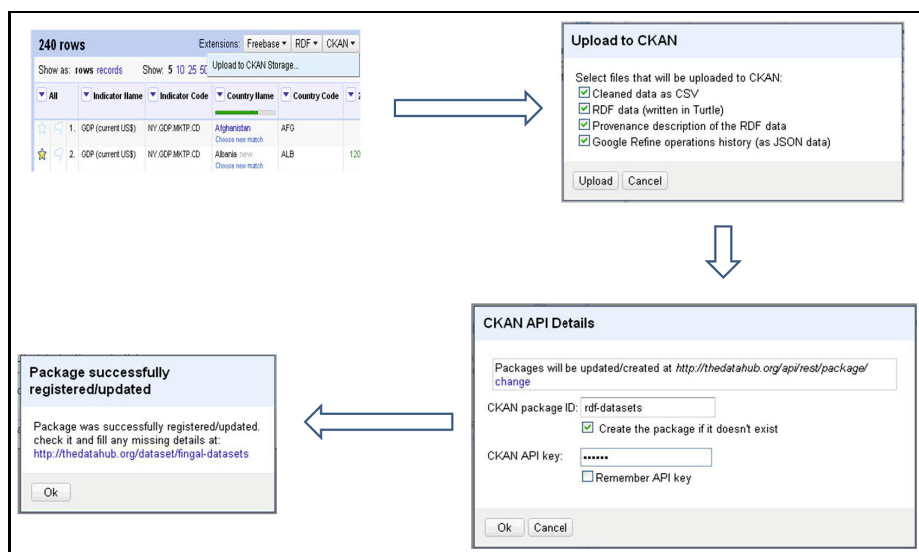


Fig. 5. User interaction flow with CKAN Extension

The catalogue provides fairly rich description of its datasets. Each dataset is categorized under one or more domain and described with a number of tags. Additionally, metadata describing spatial and temporal coverage, publisher and date of last update are also provided. Table 1 shows a quick summary of Fingal Catalogue at the time of writing.

Table 1. Fingal Catalogue summary

Number of datasets:	74 (68 available in CSV and 56 in XML)
Top publishers:	Fingal county Council (41), Central Statistics Office (17), Department of Education and Science (4)
Top domains:	Demographics(18), Citizen Participation(18), Education(9)

We applied the LGD Publishing Pipeline described in this paper to promote Fingal Catalogue to the five-star level i.e. to put the data in the interlinked RDF format. We briefly report on each of the involved steps in the following.

3.1 Machine-Readable Catalogue

Ideally, catalogues publishers make their catalogues available in some machine-readable format. Unfortunately, this is not the case with Fingal Catalogue. We had to write a scraper to get the catalogue in CSV format¹⁷. Then, using Google

¹⁷ The scraper is on ScraperWiki

http://scraperwiki.com/scrapers/fingaldata_catalogue/

Refine with RDF Extension we converted the CSV data into RDF data adhering to the Dcat model.

Most catalogues organize their datasets by classifying them under one or more domain [13]. Dcat recommends using some standardised scheme for classification so that datasets from multiple catalogues can be related together. We used the Integrated Public Sector Vocabulary (IPSV) available from the UK government. RDF representation of IPSV (which uses SKOS) is available by the esd-toolkit as a dump file¹⁸. We used this file to define a reconciliation service in Google Refine and reconcile Fingal Catalogue domains against it.

3.2 Data Cleaning-up

Google Refine capabilities were very helpful with data cleaning. For example, Google Refine Expression Language (GREL) was intensively used to properly format dates and numbers to adhere to XML Schema data types syntax.

3.3 Interlinking

Electoral divisions are prevalent in the catalogue datasets especially those containing statistical information. There are no URIs defined for these electoral divisions, so we had to define new ones under `data-gov.ie`. We converted an authoritative list of electoral divisions available from Fingal County Council into RDF. The result was used to define a reconciliation service using Google Refine RDF Extension. This means that in each dataset containing electoral divisions, moving from textual names of the divisions to the URIs crafted under `data-gov.ie` is only few clicks away. A similar reconciliation was applied for councillor names. It is worth mentioning that names were sometimes spelled in different ways across datasets. For instance, Matt vs. Mathew and Robbie vs. Robert. Reconciling to URIs eliminates such mismatches.

RDF Extension for Google Refine also enabled reconciling councillor names against DBpedia and electoral divisions against Geonames.

3.4 RDF-izing

Google Refine clustering and facets were effective in giving a general understanding about the data. This is essential to anticipate and decide on appropriate RDF models for the data. Most of the datasets in the catalogue contain statistical information, we decided to use the Data Cube Vocabulary for representing this data. Data Cube model is compatible with SDMX – an ISO standard for sharing and exchanging statistical data and metadata. It extends SCOVO [10] with the ability to explicitly describe the structure of the data and distinguishes between dimensions, attributes and measures. Whenever applicable, We also used terms

¹⁸ <http://doc.esd.org.uk/IPSV/2.00.html>

from SDMX extensions¹⁹ which augment the Data Cube Vocabulary by defining URIs for common dimensions, attributes and measures.

For other datasets, we reused existing vocabularies whenever possible and defined small domain ontologies otherwise. We deployed new custom terms online using *vocab.derri.ie* which is a web-based vocabulary management tool facilitating vocabularies creation and deployment. As a result, all new terms are documented and dereferenceable. Newly defined terms can be checked at <http://vocab.derri.ie/fingal#>.

3.5 Sharing

With the CKAN Extension, each RDF dataset published is linked to its source file and annotated with provenance information using the OPMV extension. By linking the RDF data to its source and to Google Refine operations history, a determined user is able to examine and (automatically) reproduce all these operations starting from the original data and ending with an exact copy of the published converted data.

In total, 60 datasets were published in RDF resulting in about 300K triples²⁰ (a list of all datasets that were converted and the vocabularies used is available in [12]). By utilising reconciliation, the published RDF data used the same URIs for common entities (i.e. no URI aliases) and were linked to DBpedia and Geonames. Based on our previous experience in converting legacy data into RDF, we found that the pipeline significantly lowers the required time and effort. It also helps reducing errors usually inadvertently introduced when using manual conversion or custom scripts. However, issues related to URI construction, RDF data modelling and vocabulary selection are not supported and need to be tackled based on previous experience or external services.

The RDF data were then loaded into a SPARQL endpoint. We used Fuseki to run the endpoint. We used the Linked Data Pages framework²¹ to make the data available in RDF and HTML based on content negotiation²². Resolving the URI of an electoral division, as the one for the city of Howth for example, gives all the facts about Howth which were previously scattered across multiple CSV files.

The combination of Dcat, VoiD and Data Cube vocabularies helped providing a fine-grained description of the datasets and each data item. Figure 6 shows how these vocabularies were used together. Listing 1.1 shows a SPARQL query that given a URI of a data item (a.k.a fact) locates the source CSV file from which the fact was extracted. This SPARQL query enables a user who finds a particular fact in the RDF data doubtful to download the original authoritative CSV file in which the fact was originally stated.

¹⁹ <http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/vocab/>

²⁰ The conversion required approximately two weeks effort of one of the authors.

²¹ <https://github.com/csarven/linked-data-pages>

²² The data is available online as part of <http://data.gov.ie>

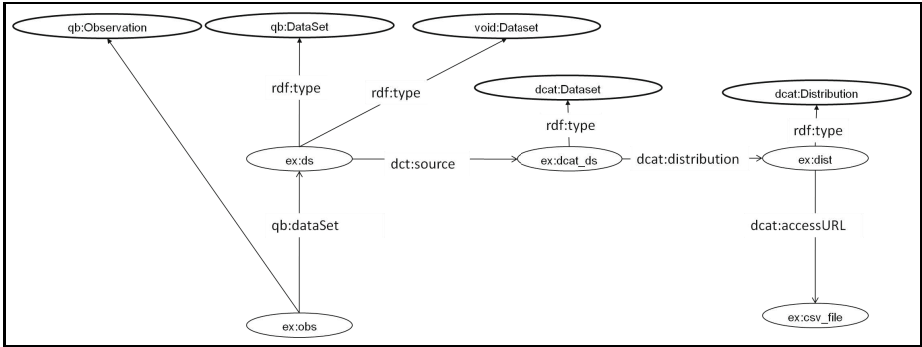


Fig. 6. The combination of Dcat, VoiD and Data Cube vocabularies to describe Fingal data

Listing 1.1. Getting the source CSV file for a particular fact (given as ex:obs)

```

1 SELECT ?dcat_ds ?csv_file
2 WHERE {
3   ex:obs qb:dataSet ?qb_ds .
4   ?qb_ds dct:source ?dcat_ds .
5   ?dcat_ds dcat:distribution ?dist .
6   ?dist dcat:accessURL ?csv_file ;
7         dct:format ?f .
8   ?f rdfs:label 'text/csv' .
9 }

```

Thanks to the RDF flexibility, the data now can also be organised and sliced in ways not possible with the previous rigid table formats.

4 Related Work

A number of tools for converting tabular data into RDF exist, most notably XLWrap [11] and RDF123 [9]. Both support rich conversion and full control over the shape of the produced RDF data. These tools focus only on the RDF conversion and do not support a full publishing process. Nevertheless, they can be integrated in a bigger publishing framework. Both RDF123 and XLWrap use RDF to describe the conversion process without providing a graphical user interface which makes them difficult for non-expert users.

Methodological guidelines for publishing Linked Government Data are presented in [17]. Similar to our work, a set of tools and guidelines were recommended. However, the tools described are not integrated into a single workbench and do not incorporate provenance description. The data-gov Wiki²³ adopts a wiki-based approach to enhance automatically-generated LGD. Their work and ours both tackle the LGD creation with a crowd-sourcing approach though in significantly different ways.

²³ <http://data-gov.tw.rpi.edu/wiki>

5 Future Work and Conclusion

In this paper, we presented a self-service approach to produce LGD. The approach enables data consumers to do the LGD conversion themselves without waiting for the government to do so. It can be seen as a civic-sourcing approach to LGD creation. To this end, we defined a publishing pipeline that supports an end-to-end conversion of raw government data into LGD. The pipeline was centred around Google Refine to employ its powerful capabilities.

We started by defining Dcat, an RDF vocabulary to describe government catalogues. Dcat was utilised to enable browsing catalogues from within Google Refine through a faceted interface. Google Refine was extended with RDF export and reconciliation functionality. Additionally, all the operations applied to the data are captured and formally represented without involving the user in the tedious and verbose provenance description. Finally, results can be shared on CKAN.net along with its provenance.

The publishing pipeline was applied to a local government catalogue in Ireland, Fingal County Catalogue. This results in a significant amount of Linked Data published. Data Cube vocabulary was used to model statistical data in the catalogue. Google Refine editing features and the added RDF capabilities were successfully applied to properly shape the data and interlink it.

Further work on the community and collaboration aspects of the publishing process would add a great value. Additionally, the problem of choosing a proper RDF model for the data is an important aspect that was not tackled in this work and cannot be considered solved in general.

Acknowledgments. The work presented in this paper has been funded in part by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2) and the European Union under Grant No. 238900 (Rural Inclusion).

References

1. 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.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 708–721. Springer, Heidelberg (2007)
2. Berners-Lee, T.: Putting Government Data Online. WWW Design Issues (2009)
3. Berrueta, D., Phipps, J.: Best Practice Recipes for Publishing RDF Vocabularies. World Wide Web Consortium, Note (August 2008)
4. Bizer, C., Cyganiak, R., Heath, T.: How to Publish Linked Data on the Web. Web page (2007) (revised 2008)
5. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems (IJSWIS) (2009)
6. Cyganiak, R., Maali, F., Peristeras, V.: Self-service Linked Government Data with dcat and Gridworks. In: Proceedings of the 6th International Conference on Semantic Systems, I-SEMANTICS 2010. ACM (2010)

7. de León, A., Saquicela, V., Vilches, L.M., Villazón-Terrazas, B., Priyatna, F., Corcho, O.: Geographical Linked Data: a Spanish Use Case. In: Proceedings of the 6th International Conference on Semantic Systems, I-SEMANTICS 2010. ACM (2010)
8. 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: Science, Services and Agents on the World Wide Web* (2011)
9. Han, L., Finin, T., Parr, C., Sachs, J., Joshi, A.: RDF123: From Spreadsheets to RDF. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 451–466. Springer, Heidelberg (2008)
10. Hausenblas, M., Halb, W., Raimond, Y., Feigenbaum, L., Ayers, D.: SCOVO: Using Statistics on the Web of Data. 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. 708–722. Springer, Heidelberg (2009)
11. 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)
12. Maali, F.: Getting to the Five-Star: From Raw Data to Linked Government Data. Master’s thesis, National University of Ireland, Galway, Ireland (2011)
13. Maali, F., Cyganiak, R., Peristeras, V.: Enabling Interoperability of Government Data Catalogues. In: Wimmer, M.A., Chappellet, J.-L., Janssen, M., Scholl, H.J. (eds.) EGOV 2010. LNCS, vol. 6228, pp. 339–350. Springer, Heidelberg (2010), http://dx.doi.org/10.1007/978-3-642-14799-9_29
14. Maali, F., Cyganiak, R., Peristeras, V.: Re-using Cool URIs: Entity Reconciliation Against LOD Hubs. In: Proceedings of the Linked Data on the Web Workshop 2011, LDOW 2011 (March 2011)
15. Nam, T.: The Wisdom of Crowds in Government 2.0: Information Paradigm Evolution toward Wiki-Government. In: AMCIS 2010 Proceedings (2010)
16. Sheridan, J., Tennison, J.: Linking UK Government Data. In: Proceedings of the WWW 2010 Workshop on Linked Data on the Web (LDOW 2010) (2010)
17. Villazón-Terrazas, B., Vilches-Blázquez, L.M., Corcho, O., Gómez-Pérez, A.: Methodological Guidelines for Publishing Government Linked Data. In: Wood, D. (ed.) *Linking Government Data*, ch. 2. Springer (2011)
18. Zhao, J.: The Open Provenance Model Vocabulary Specification. Technical Report, University of Oxford (2010)

Achieving Interoperability through Semantic Technologies in the Public Administration

Chiara Di Francescomarino, Mauro Dragoni, Matteo Gerosa, Chiara Ghidini,
Marco Rospoche, and Michele Trainotti

FBK-irst, Via Sommarive 18 Povo, I-38123, Trento, Italy
{dfmchiara, dragoni, gerosa, ghidini, rospoche, mtrainotti}@fbk.eu

Abstract. In this paper we report the experience of using semantic based tools and technologies for (collaboratively) modeling administrative procedures and their related documents, organizational roles, and services, in the Italian Public Administration (PA), focusing in particular on the interoperability aspects faced during the modelling process. This experience, the reported lessons learned and next steps identified, highlight the potential and criticality of using web 2.0 semantic technologies and tools to enhance participatory knowledge sharing, interoperability, and collaboration in the modeling of complex domains in the PA.

1 Introduction

In the last few years, the Public Administrations (PA) of several countries around the world have invested effort and resources into modernizing their services in order to improve labor productivity, as well as, PA efficiency and transparency. The recent contributions and developments in ICT (Information and Communication Technology) can boost this modernization process, as shown by the support the ICT can provide to the replacement of paper-based procedures with electronic-based ones (dematerialization of documents) within the PA. An important contribution of the ICT, in supporting the dematerialization of documents, is the production of proper and precise models of the administrative procedures of the PA and of the specific “entities” related to these procedures, such as the *documents* involved in the procedures, the *organizational roles* performing the activities, and the *services* needed to manage the electronic documents in an archival system. In fact, by following a model-driven approach [815], the availability of these models is a key factor towards both (1) the re-design and re-engineering of the administrative procedures, in order to replace paper-based documents with electronic-based ones, and (2) the definition of an appropriate archival system able to safely store, catalogue, manage, and retrieve the electronic documents produced within the PA. The definition of these models, which can act as “reference models” at the national level and enhance interoperability as described in [15], is often made complex, among the other problems, by the heterogeneity of procedures, document typologies, organizational structures, terminologies, and so on, present at regional or local level, due for instance to different regional laws or traditions.

In this paper, we report the experience of using semantic based technologies and a wiki-based modeling tool, MoKi [10], in the context of the ProDe Italian national project, in order to build national “reference models” for the management of electronic

documentation in the Public Administration (PA). These models aim at representing both the domain entities and the processes in several fields of the PA with the focus on document management. Due to internal reasons the project didn't adopt available standard conceptual schemata for the representation of data and processes, but the different regional actors were initially asked to model their administrative procedures in a bottom up manner. This fact originated highly heterogeneous representations, which needed to be shared, reconciled, and eventually re-defined in terms of a common conceptual schemata subsequently adopted within the project, and in terms of common (pre-existing and ad-hoc) terminologies and meta-data. Here we highlight how MoKi was used to support participatory knowledge sharing and collaboration within the modeling activities of the different regions involved in the project in the spirit of the Web 2.0, and we report some lessons learned and future steps in our work, especially emphasizing the aspects related to the collaboration of PA employees of different regions, the reconciliation of local PA procedures into high level interoperable ones, the confluence of terminologies into shared lexicons, as well as the mapping between organizational and technological layers.

The contribution of the paper is twofold: first, it identifies the different interoperability aspects that originated in the context of the creation of reference models in a national project, it classifies them in light of standard interoperability models such as the "European Interoperability Framework for European public services" (EIF) [4], and it provides an overview of how MoKi and semantic based technologies have been used to face these issues. Second, it provides an attempt to report lessons learned and future steps especially related to how semantic-wiki based systems can support distributed modeling, the confluence of terminologies into shared lexicons, the adoption of (standard) pre-existing terminologies and metadata when available, and the integration of different entities for the construction of complex models. This contribution extends the work presented in [3] where the experience of the ProDe project was analyzed by looking at the usefulness of MoKi to support the collaboration process between knowledge engineers and domain experts in their modeling activities, and where aspects related to interoperability and construction or usage of shared knowledge were not considered.

The paper is structured as follows. In Section 2 we report related works concerning: (i) the usage of Semantic Web technologies in the PA domain, (ii) reference interoperability frameworks for the PA and (iii) model driven approaches towards interoperability. Section 3 provides an overview of ProDe and of the interoperability aspects in it. Section 4 describes how a semantic-based platform, based on the MoKi tool, has been implemented to face interoperability issues, while Section 5 presents how each interoperability aspect has been addressed within the ProDe project, what we learned from the experience and what we plan to do next. We conclude in Section 6 with some final remarks.

2 Related Works

Several works have focused on the application of Semantic Web technologies in the PA domain. We recall a few of them which have some commonalities with the work presented in this paper. In [17], the authors present a web-based knowledge management system that, by providing an up-to-date and accurate legal framework, supports (i) civil

servants in the composition of administrative acts and (ii) civil servants, citizens and businesses in reasoning and substantiating administrative acts by means of precedents and opinions. In the context of the SAKE EU project, [19] proposes an ontology-based approach for the systematic management of changes in knowledge resources in public administrations. Successful applications of semantic wiki based technologies in the eGovernment domain have been reported in [11,20], to favour the management and sharing of information and knowledge.

Some interoperability frameworks have been defined to grant the interoperability between different systems in the context of complex infrastructures. The Levels of Information Systems Interoperability (LISI) [1] initiative of the US Department of Defense aims to identify the stages through which systems should logically progress, or “mature”, in order to improve their capabilities to interoperate. LISI considers five increasing levels of sophistication regarding system interaction and the ability of the system to exchange and share information and services. Each higher level represents a demonstrable increase in capabilities over the previous level of system-to-system interaction. A more recent framework, adopted by the European Commission, is the European Interoperability Framework for European public services (EIF) [4]. It defines a set of recommendations to support the delivery of European public services, by classifying the interoperability aspects to be addressed according to different interoperability levels (legal, organizational, semantic and technical).

Following the definition provided in [15], the approach taken in the ProDe project can be classified as a model-driven approach, where models have been systematically used as primary artifact for the definition of common procedures within different regions and for the engineering of the document management system. Of the three modeling sub-categories defined in [15], the models developed in ProDe cover the first and the second, that is, the specification of (domain) data - provided by means of OWL ontologies - and the specification of processes - provided by means of BPMN representations. The approach taken in ProDe, and the conceptual model developed to represent data, bring some relation with the effort carried put in the UK Government Common Information Model [13], where a reference model is defined to support the elicitation and setting out of the Requirements specifications for e-service development. Differently from [13], where the models are centered around the notion of e-service, the data models of ProDe are centered around the notion of document. Given the importance of documents within the project, data have been described in terms of the MoReq metadata standard [2], which in turn can be represented in terms of Dublin Core Metadata [5] as specified in [2]. Concerning the modeling of process knowledge, [15] classifies the efforts of the PA in two different families: (i) process modeling, and (ii) service modeling. In ProDe, the objective was to model general processes that are common to a large number of PA, and can therefore be classified as a process modeling effort. In that respect, the approach follows the one accomplished by SAP in the encoding of generic process models for different fields on its Solution Maps [16].

The ProDe project has been conceived keeping in mind the technological framework realized in ICAR¹, a national project addressing the establishment of the Italian Public Connectivity and Cooperation System (SPC).

¹ <http://www.progettoicar.it/>

3 Interoperability Aspects within ProDe

ProDe² is an Italian project with the aim of defining a national reference model for the management of electronic documentation (dematerialized document) in the Public Administration. This reference model follows an archival science perspective, and can be used for the identification of guidelines and functions needed to safely store, classify, manage, and retrieve, electronic documents produced within the PA in an archival system.

The project has a duration of 30 months (May 2010-October 2012) and the work-plan is composed of 11 tasks assigned to 11 teams (*task-teams*) each one coming from one among 10 Italian regions. The 11 tasks are divided in 4 central tasks and 7 peripheral tasks. The 4 central tasks are in charge of guiding the activities and developing a common framework in which all the regions could recognize themselves. Each of the 7 peripheral tasks provides instead a specific expertise on a different sector of the PA and is in charge of guiding the modeling of administrative procedures for the sector it has been assigned to. Thus, the central tasks provide the main expertise in archival science, while the peripheral tasks provide domain expertise in different fields of the PA.

The setting and objectives of the project shows that interoperability aspects, of various nature and involving diverse actors and entities, play an important role within ProDe. In the following, we first briefly summarize the interoperability aspects as they naturally arose within the project, and then show how they relate to a standard interoperability framework such as EIF.

Users Interoperability. The development of a reference model for managing the dematerialization of documents demands the involvement of actors with different backgrounds, modeling skills, and responsibilities, spanning from experts in archival science, experts in laws, business process analysts, and knowledge engineers. Supporting the collaboration and cooperation among these actors to achieve the development of a shared reference model is even more crucial in ProDe, as (i) users from different regions distributed over Italy are required to contribute to the definition of such model, and (ii) the domains of the documents considered for dematerialization are various (e.g., healthcare, human resources, material resources, and so on).

Procedures Interoperability. The Italian legislation provides regions with a high degree of independence/freedom in writing new laws and in organizing their own structure to answer the citizens' needs. This explains why the 10 regions participating in the ProDe project: (i) have different levels of dematerialization; (ii) refer to different laws and regulations; (iii) use different methods, structures and terms for representing their administrative procedures. Nevertheless, as one of the goals of the project is to develop an archival system based on a common reference model shared by all the regions, this heterogeneity has to be taken into account in that the administrative procedures in place in the various regions, being understood their specificity, have to be compatible/compliant with the common conceptualization adopted.

Lexicon Interoperability. The freedom of Italian regions in self-organizing their structure and regulations is also reflected in the heterogeneity of the lexicon adopted in

² <http://www.progettoprode.it/Home.aspx>

their administrative procedures. Indeed, it often happens that each region has its own name for designating documents reporting the same information, thus severely hindering comprehension of the process across regions.

Formal Language Interoperability. The management of document dematerialization requires to deal with different entities and artifacts: for instance, the (i) nature and properties of the documents to be dematerialized, (ii) the procedures and activities to store, catalogue, manage, and retrieve these document, and (iii) the actors involved in these activities. These entities have diverse intrinsic nature and are commonly formally represented with different modeling languages: for instance, documents and actors can be suitably modeled with declarative formalisms (e.g. ontologies), while business processes formalism are more appropriate to correctly represent procedures and activities.

Organizational and Technological Interoperability. In modeling the processes of a complex organization like a PA, it is common to identify at least two conceptual levels at which these processes take place: the *organizational layer*, comprising the activities, roles, processes, and organizational structure of the PA, and the *technological layer*, managing the set of information systems and software solutions that the PA uses to perform (part of) its activities. Although the conceptual connection between these two layers is rather evident, making it explicitly established and formalized in an integrated architecture enables to offer to complex organizations additional added-value services, like (i) verifying that the information systems supporting the organization are compliant with its processes, (ii) monitoring the execution of the organization processes, and (iii) checking (and possibly improve) the organization efficiency.

Placing ProDe Interoperabilities within the EIF. The interoperability issues encountered and identified in the ProDe project do not perfectly map to the four interoperability layers proposed by EIF [4]. This is mainly due to the different goals of the project and the framework: EIF is a set of recommendations that specify how European administrations should communicate with one another within the EU and across Member States borders in order to provide services; the ProDe project, instead, aims at defining national “reference models” of PAs’ procedures and domain entities, starting from the existing local ones. This means that, for example, procedures carried out locally, are aligned at an abstract level, leaving regions the freedom to detail them according to their needs, so that the abstract version of a process model developed by a region can be used as base for the specificities of other regions.

Nevertheless, the interoperability aspects that came out within ProDe, are explicitly or implicitly related to the EIF interoperability layers. In detail:

- *lexicon interoperability* and *formal language interoperability* are related to the EIF SEMANTIC INTEROPERABILITY level. Both the interoperability aspects, in fact, deal with language heterogeneity that, hampering a common understanding, requires the provision of a “precise meaning” associated either to a shared vocabulary or to the relationships existing among different formal languages.
- *procedures interoperability* lies in the middle between the EIF LEGAL and ORGANIZATIONAL INTEROPERABILITY layers. The aspect deals, in fact, with the

legislation and organization’s procedure heterogeneity and the consequent need of their alignment, though at an abstract level.

- *users interoperability* relates to both the SEMANTIC and, at a higher level, the ORGANIZATIONAL INTEROPERABILITY layer. The interoperability among users with different backgrounds, competencies and roles, in fact, demands, on one hand, the achievement of a common understanding of their different views and, on the other, their “collaboration for the achievement of their mutually agreed goals”.
- *organizational and technological interoperability* is orthogonal to the EIF layers: it in fact deals with the connection of the ORGANIZATIONAL and the TECHNICAL levels.

4 Toward Achieving Interoperability in ProDe

In order to face the interoperability issues described in the previous section, and to create a common reference model shared by all the regions belonging to the project, a common conceptual schema was proposed to the experts of the different task-teams to guide the modeling of their administrative procedures, the related documents, and the services to be provided by the document management system. This conceptual schema, whose simplified version is graphically depicted in Figure 1 using an Entity-Relationship notation, was developed by the experts in archival, computer, and organizational sciences working in the central tasks of the ProDe project, and it represents an extension of the one presented in [3]. In detail, the new entity *Service* is used to describe the functionalities required to the document management system by a given task in order to handle the documents managed within the task.

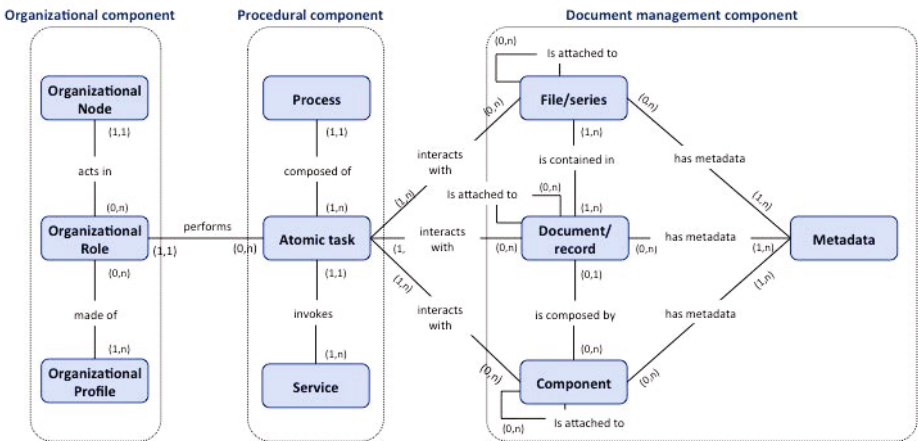


Fig. 1. The conceptual schema

³ We omit an in depth description of the ProDe conceptual schema. The interested reader can refer to [3] for a detailed description.

The second, and more important, contribution towards the achievement of interoperability was the customization and usage of a platform based on MoKi [9], a tool for collaborative modeling of integrated processes and ontologies, in order to obtain models following the conceptual schema presented in Figure 1. The platform developed for the ProDe project (hereafter referred to as the ProDeMoKi Platform) provides a set of MoKi installations: one installation for each of the peripheral tasks, hereafter named PT_1, \dots, PT_7 , and a single installation CT for all the central tasks, where each MoKi installation PT_1, \dots, PT_7 was connected with the one for the central task CT . The main idea of this platform is that, by using CT , the central tasks are able to create and manage entities (e.g., metadata for the description of documents) that are subsequently, and automatically, made available to PT_1, \dots, PT_7 (e.g., to describe their documents), thus favoring convergence and re-use.

Next we show in detail the general architecture of the MoKi tool.

4.1 The MoKi Architecture and Tool

MoKi [9] is a collaborative MediaWiki-based [12] tool for modeling ontological and procedural knowledge. The main idea behind MoKi is to associate a wiki page, containing both unstructured and structured information, to each entity of the ontology and process model. From a high level perspective, the main features of MoKi are:

- *the capability to model different types of conceptual models, described in different formal languages, in an integrated manner.* This feature is grounded on two different characteristics of MoKi. First of all, MoKi associates a wiki page to each *concept*, *property*, and *individual* in the ontology, and to each (complex or atomic) *process* in the process model. Special pages enable to visualize (edit) the ontology and process models organized according to the generalization and the aggregation/decomposition dimensions respectively. The ontological entities are described in Web Ontology Language (OWL [18]), while the process entities are described in Business Process Modeling Notation (BPMN [14]). Second, MoKi has extended the functionalities of the BPMN Oryx editor [6], to annotate process elements with concepts described in the ontology, or to incorporate data objects formalized in the ontology. The integrated procedural and ontological knowledge is then exported in a comprehensive OWL model following the approach described in [7].
- *the capability to support on-line collaboration between members of the modeling team, including collaboration between domain experts and knowledge engineers.* MoKi is an on-line tool based on MediaWiki, thus inheriting all the collaborative features provided by it. In addition MoKi facilitates the collaboration between domain experts and knowledge engineers by providing different access modes to the description (both structured and unstructured) of the elements contained in the model. In details, the current general version of MoKi is based on three different access modes⁵:

⁴ See also <http://moki.fbk.eu>

⁵ The reader is referred to [10], where the architecture of MoKi has been presented in more details, by describing also how domain experts and knowledge engineers are able to exploit these different access modes in order to work collaboratively for modeling ontologies.

Lightly-structured Access Mode: M0

The image shows a web-based form titled 'Properties' for defining metadata information. The form is organized into several sections:

- Category:** A dropdown menu with 'Metadato Descrittivi' selected.
- Description:** A large empty text area.
- Standard and Reference Model:** A large empty text area.
- Entity:** Four checkboxes: 'Documento/Record', 'Struttura Aggregativa', 'File', and 'Ruolo'.
- Requested:** A dropdown menu with 'Obbligatorio' selected, followed by a row of checkboxes labeled 'Obbligatorio per:' with options CE2, CE3, RA1, RA2, RA3, RA4, RA5, RA6, and RA7.
- Source:** A dropdown menu with 'RAS' selected.

Below the form is a 'Save' button.

Fig. 2. The template used to insert metadata information

- an *unstructured access mode* (for all users) to view/edit the unstructured content;
- a *fully-structured access mode* (for knowledge engineers) to view/edit the complete structured content; and
- a *lightly-structured access mode* (for domain experts) to view/edit (part of) the structured content in a simplified way, e.g. via light forms.

These features have been proved extremely important in the context of the ProDe project. In fact, the scenario addressed in the project required the modeling of administrative procedures, usually better described using a business process modeling notation, enriched with knowledge which typically resides in an ontology, such as the classification of document types, organizational roles, and so on. Moreover, the modeling team was composed by an heterogeneous group of domain experts and knowledge engineers situated in different Italian geographical regions.

Indeed, in the context of the ProDe project, many of the modeling actors involved in the ProDe project were not familiar with ontology modeling. Therefore, we facilitated the usage of MoKi by providing personalized lightly-structured access mode for each typology of entities that the users had to model (the ones in the Document management component, and Organizational structure component in Figure 1). An example of one of these personalized views is reported in Figure 2. The figure shows the template used for defining metadata entities.

Hereafter, we will refer to this version of MoKi providing personalized lightly-structured access mode as ProDeMoKi.

5 Interoperability in ProDe: What We Did, What We Learned and What We Will Do Next

The ProDeMoKi Platform has been extensively used by 2 central task-teams and 6 peripheral task-teams⁶ for the last 12 months. Overall, 2255 wiki pages have been created, 6809 revisions realized, 710 pages deleted and 71 pages renamed by both peripheral and central task users. Moreover, as comprehensively presented in [3], ProDeMoKi Platform users have been interviewed, by means of an on-line questionnaire, about the ease of use and the usefulness of the ProDeMoKi tool, in order to collect their subjective impressions.

The analysis of the huge amount of usage data (collected analyzing the MediaWiki database and the server log files), the users' subjective perception, and the concrete experience in the field gained during the project, allowed us to derive interesting observations about the support provided by semantic technologies to the different interoperability aspects demanded by ProDe (described in Section 3). In the following we report, for each of these interoperability aspects, the way in which it has been addressed by the ProDeMoKi Platform, the lessons we learned from the project experience and from the ProDeMoKi Platform usage, and some challenging ideas for future steps. Finally, we summarize some further related lessons learned.

5.1 Users Interoperability

The users involved into the ProDe project have different background based on their working area within the PA. Indeed, the domain experts, belonging to each of the peripheral task-teams, are specialized in specific topics, like healthcare, human resources, and financial resources. The MoKi platform provides a web-accessible knowledge sharing system that permits to all users - both within the same team and across different ones - to cooperate and to provide feedbacks about the modeled processes, and how documents are described in the platform.

Lessons Learned. In the evaluation of ProDeMoKi described in [3], we observed that about 45% of the users considered the collaboration support provided by ProDeMoKi one of the major strength of the tool and that users positively perceive its overall usefulness for the collaborative modeling of documents and processes. Such a usefulness is perceived more strongly by employees working in teams constituted by more than two persons (on average the usefulness of ProDeMoKi has been judged 3.8 out of a 5-point Likert scale for teams with more than two persons versus 2.8 for those with less than three). As presented in [3], there exists, in fact, a strong positive correlation between the size of the subject's team and his/her feedback about the ProDeMoKi usefulness for collaborative purposes. A similar relation (with the task-team's size) was also found for the perceived usefulness of the ProDeMoKi log history functionality. This functionality, although not frequently used by the peripheral task users (64 times in total), has been exercised 42% of times by the most productive (223 documents and 418 processes

⁶ Two central and one peripheral task-teams are not required to use the ProDeMoKi Platform in this phase of the project.

and tasks) and among the most numerous (4 modelers) task-teams, thus remarking its usefulness in case of large models and of collaborative work. These results show that the ProDeMoKi tool is particularly useful in situations in which users work in team on the same models.

Furthermore, the ProDeMoKi Platform is able to support the collaborative work of users with different backgrounds, by providing simplified views according to the roles and the specific competencies of the involved actors. Indeed, besides offering simplified views customized on the base of the specific domain (i.e., administrative procedures) to non-technical experts, the platform also tailors its interface to the specific actors' needs, e.g., offering different functionalities to the central and the peripheral task users. The extensive use of these views (the lightly-structured access mode of documents and the fully-structured access mode of processes have been accessed respectively 931 and 2533 times by the peripheral task users, while the central task-teams accessed the lightly-structured access mode of metadata and services 127 and 166 times, respectively) confirmed the usefulness of these simplified and customized views.

Next steps. We plan to better investigate with controlled experiments the collaboration mechanisms occurring among the different actors involved in the creation of interoperable models. The results and the feedback obtained will allow us to exploit the semantic web technologies to further support ProDeMoKi users in their modeling activities.

5.2 Procedures Interoperability

One of the aim of the ProDe project is to provide an archive of procedures representing the administrative processes of all the regions involved in the project. To this purpose, the ProDeMoKi Platform permits to all peripheral tasks to archive the process models they are in charge to deal with, and to make these models available to the users of the other peripheral and central tasks. This way, all users of the other tasks are able to verify the compliance between the processes stored in the archive and the ones actually used in their local government.

Lessons Learned. Differently from what happened in the modeling of document types and organizational aspects, where it was possible to identify relations and commonalities between the different models (taxonomies) produced by the different peripheral task-teams already at the early stage of modeling, the formal representation of PA procedures generated a number of extremely different and heterogeneous process models. Such a variety and heterogeneity, due to granularity issues, different modeling styles, lack of guidelines and reference standards, and to the difference among regional procedures, hampered the convergence to the ProDe archive of procedures commonly agreed by all the participant regions. In this scenario the ProDeMoKi Platform enabled the identification of these diverging modeling styles from the very early stages of the project by allowing participatory knowledge sharing and fostering the communication and the discussion among regions, and held an crucial role in supporting the process of converging towards a uniform common model. Currently, the model contains 109 processes modeled by 6 peripheral task-teams (with an average of 18 processes per task-team).

5.3 Lexicon Interoperability

The lexicon interoperability is one of the crucial issues of the ProDe project. As explained in Section 3, it is critical in order to avoid ambiguity problems that the domain experts are able to use a common lexicon for describing both the document properties and the atomic activities used in each process.

The effort spent during the project to this purpose was mainly devoted to: (i) the adoption of (standard) pre-existing terminologies and metadata (e.g. MoReq), when available; (ii) the creation of a shared vocabulary agreed among the different regions for the service definition.

The architecture of the ProDeMoKi Platform provides a mechanism to link the *CT* installation and the PT_1, \dots, PT_7 ones in order to grant the semantic interoperability of the used dictionaries, thus supporting regions in both these activities. Indeed, on one side, this linking functionality provides the tool with the capability to enable the definition of a common set of shared objects, that allowed the 4 central tasks to define a common set of metadata, services, functionalities and indicators to be used by all the peripheral tasks. On the other side, this functionality supports task-teams in reconciling synonyms and in mapping specific terms to more general and shared ones. This way, it allows them to come up with a common dictionary, based on MoReq, to be used by the peripheral tasks for describing both the document properties (metadata) and the services invoked by each atomic task.

Lessons Learned. The results of the effort spent on the convergence to a common dictionary, clearly appear in the reduction of the ambiguity of document and activity names in successive versions of the models created. For example, the number of different activities modeled dropped from more than 170, in the first version of the “Modello di riferimento”, to 22 in the last version, with a relative reduction of about 87%.

Next steps. The definition of high level models commonly agreed by all the participant regions, as well as the use of a shared set of metadata and of a common dictionary, represent the first step towards the possibility of (semi-)automatically verifying the compliance of the high level models to both national and regional laws, that is of primary importance for the PA. Moreover, the shared vocabulary of terms, fostering the definition of a mapping between the specific regional procedures and the commonly agreed models, could allow to verify the compliance of regional models to both national and local norms, and to support their adaptation to changes in the regulations.

5.4 Formal Language Interoperability

The complexity of PA procedures demands for the modeling and integration of different entities and artifacts. Each ProDeMoKi in the ProDeMoKi Platform permits to model the ontology of the documents, the processes in which they are used, and roles of the users involved in each process. To grant the interoperability of the formal languages used to describe these different conceptual models, the platform permits to build integrated models in which the entities defined in different formal languages can be semantically related, in order to better represent the PA procedures. An example is

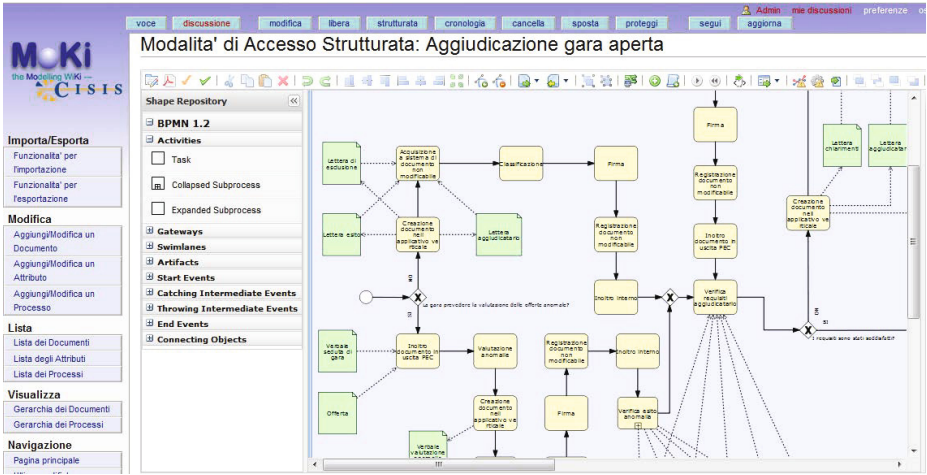


Fig. 3. The BPMN diagram enriched with documents

reported in Figure 3 in which the BPMN diagram shows how document entities are connected with processes.

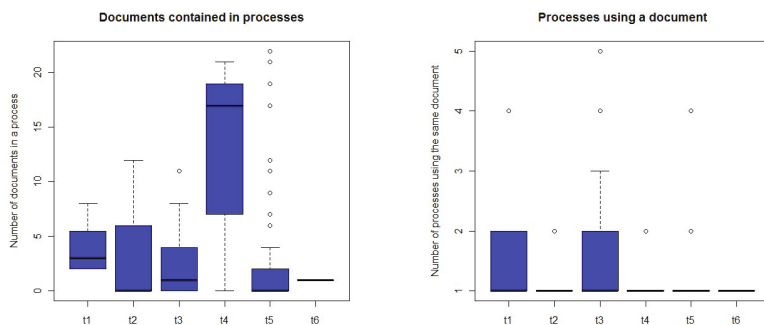
Lessons Learned. The importance of the interoperability among the formal languages used for describing the different conceptual models of the PA procedures can be grasped also from the data related to the ProDeMoKi Platform usage. For example, 3.32 documents have been used, on average, in each process diagram, with peaks of about 20 different documents in a process (as shown in the boxplot in Figure 4a). Moreover, the same document has been used on average by 0.82 processes, including cases in which the same document has been used by 4/5 different processes (Figure 4b). Users themselves are aware of the importance of such a facility: in fact, 45% of them judged such a capability of ProDeMoKi one of its major strengths.

5.5 Organizational and Technological Interoperability

As illustrated in Section 3, identifying and linking the organizational and the technological layers becomes a necessity when dealing with complex organizations like the PA, as occurred in the ProDe project.

Such an interoperability is achieved in the ProDeMoKi Platform thanks to the connection between each ProDeMoKi installation of the peripheral tasks with the one used in the central tasks. Indeed, as explained in Section 3, one of the goal of the central tasks is to define the services that can be invoked by each process modeled by the peripheral tasks. This way, the users of the peripheral tasks are able both to verify the completeness of the services provided, and to map these services with the organizational procedures and roles of their local government.

Lessons Learned. The conceptual difference between the organizational and technological layers is rather evident. However, we learned that the interoperability between



(a) Distribution of documents in processes per task-team

(b) Distribution of processes containing the same document per task-team

Fig. 4. Document and process integration

these two conceptual layers is hard to reach in the context of the ProDe project. This is mainly due to the several differences in the regions' organizations, that make the creation of a clear mapping between services, roles and procedures a challenging task.

Next steps. After having identified and modeled these two layers, the next step will be to draw formal relations between them. Being able to link these layers, for example determining when a certain technological component accomplishes a certain step in the business process, could allow us to monitor the PA organizational process by monitoring the progress of the corresponding process at the software layer. More importantly, when faced with a change in the organizational process we could automatically modify the technological process to reflect this change.

5.6 Additional General Lesson Learned

Among the peculiar findings of the project, we observed that PA employees are technological advanced users, even more than what we expected. Indeed, ProDeMoKi users not only work with a personal computer everyday for their job (mainly to write and read documents), but they are also people living in a world where the use of web technologies is constantly increasing. People navigate the Internet at home looking for news, events, restaurants; they use social networks to stay in touch with friends, online calendar to organize their lives and wikipedia when they want to research a specific topic.

Although PA employees don't commonly use Semantic Web technologies during their job, they use it everyday in their spare time. This is the reason why, for them, the use of a tool based on the same concept of the popular Wikipedia wasn't too challenging. We were surprised to discover that not only almost all the interviewed ProDeMoKi users frequently visit websites like wikipedia, but more than half of them edited at least one wiki page prior to use ProDeMoKi.

This familiarity with Semantic Web technologies allowed them to quickly learn how to use ProDeMoKi. Before the beginning of the modeling activities (February 2011),

the PA employees have been trained with a learning session of 1 day, in which all the features of ProDeMoKi have been illustrated, and hands-on exercises proposed. After this session, the time spent for learning was very limited (on average, 1-2 days) and the learning process did not require the involvement of ProDeMoKi developers (the preferred approach was the autonomous training).

The proliferation of Semantic Web technologies, that we can envisage for a next future, could allow the quick and easy adoption of platforms like the ProDeMoKi Platform, as well as the growth of communities around these technologies. A further lesson we learned from the ProDe project, indeed, is the importance to actively involve users in the development process of the project, making them collaborating as part of a community. In the context of ProDe, the active participation and collaboration of PA employees of different regions, allowed to develop a common lexicon (by sharing knowledge and discussing), to refine models and procedures (by confronting them with those of other regions), and could allow, in the future, to keep alive the attention in maintaining and evolving the models built together as the PA procedures change.

6 Conclusions

The paper reports our experience in the construction and usage of solutions based on Semantic Web technologies in the context of ProDe, a national project involving Italian Public Administrations. In particular, it presents how these technologies enabled the collaborative modeling of administrative procedures and their related documents, organizational roles, and services, and contributed to deal with the interoperability issues emerged in the context of project. More specifically, the features provided by the MoKi tool and its customizations to face the specific needs of the project allowed to promote interoperability among: users, PA procedures, terminologies, conceptual models, and the different conceptual layers required by the project.

Taking advantage of the experience and of the lessons learned during the project, we plan, for the future, to better investigate and support the collaboration and interoperability mechanisms among users with different competencies and roles, as well as to explore techniques and approaches for (i) enabling the compliant evolution of PA procedures and laws; and (ii) monitoring the execution of PA procedures to check their compliance to models.

References

1. Levels of information systems interoperability (lisi) (1998), <http://www.eng.auburn.edu/~hamilton/security/DODAF/LISI.pdf>
2. MoReq2 specification: Model requirements for the management of electronic records (2008), http://ec.europa.eu/transparency/archival_policy/moreq/doc/moreq2_spec.pdf
3. Casagni, C., Di Francescomarino, C., Dragoni, M., Fiorentini, L., Franci, L., Gerosa, M., Ghidini, C., Rizzoli, F., Rospocher, M., Rovella, A., Serafini, L., Sparaco, S., Tabarroni, A.: Wiki-Based Conceptual Modeling: An Experience with the Public Administration. 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. 17–32. Springer, Heidelberg (2011)

4. European Commission. European Interoperability Framework (EIF) for European public services (2010), http://ec.europa.eu/isa/documents/isa_annex_ii_eif_en.pdf
5. DCMI. Dublin core metadata initiative (2007), <http://dublincore.org/>
6. Decker, G., Overdick, H., Weske, M.: Oryx – An Open Modeling Platform for the BPM Community. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) BPM 2008. LNCS, vol. 5240, pp. 382–385. Springer, Heidelberg (2008)
7. Di Francescomarino, C., Ghidini, C., Rospocher, M., Serafini, L., Tonella, P.: Semantically-Aided Business Process Modeling. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 114–129. Springer, Heidelberg (2009)
8. France, R., Rumpé, B.: Model-driven development of complex software: A research roadmap. In: 2007 Future of Software Engineering, FOSE 2007, pp. 37–54. IEEE Computer Society, Washington, DC (2007)
9. Ghidini, C., Rospocher, M., Serafini, L.: Moki: a wiki-based conceptual modeling tool. In: ISWC 2010 Posters & Demonstrations Track: Collected Abstracts, Shanghai, China. CEUR Workshop Proceedings (CEUR-WS.org), vol. 658, pp. 77–80 (2010)
10. Ghidini, C., Rospocher, M., Serafini, L.: Conceptual modeling in wikis: a reference architecture and a tool. In: eKNOW 2012, The Fourth International Conference on Information, Process, and Knowledge Management, pp. 128–135 (2012)
11. Krabina, B.: A semantic wiki on cooperation in public administration in europe. Journal of Systemics, Cybernetics and Informatics 8, 42–45 (2010)
12. Wikimedia Foundation. Mediawiki, <http://www.mediawiki.org>
13. Cabinet Office Office of the e Envoy. e-services development framework primer v1.0b (2002), <http://www.dcc.uchile.cl/~cgutierr/e-gov/eSDFprimer.pdf>
14. OMG. BPMN, v1.1, www.omg.org/spec/BPMN/1.1/PDF
15. Peristeras, V., Tarabanis, K., Goudos, S.K.: Model-driven e-government interoperability: A review of the state of the art. Computer Standards & Interfaces 31(4), 613–628 (2009)
16. SAP. Solution maps, <http://www1.sap.com/solutions/businessmaps/solutionmaps/index.epx>
17. Savvas, I., Bassiliades, N.: A process-oriented ontology-based knowledge management system for facilitating operational procedures in public administration. Expert Systems with Applications 36(3, pt. 1), 4467–4478 (2009)
18. Smith, M.K., Welty, C., McGuinness, D.L.: Owl web ontology language guide. W3C Recommendation, February 10 (2004)
19. Stojanovic, N., Apostolou, D., Ntioudis, S., Mentzas, G.: A semantics-based software framework for ensuring consistent access to up-to-date knowledge resources in public administrations. In: Metadata and Semantics, pp. 319–328. Springer, US (2009)
20. Wagner, C., Cheung, K.S.K., Ip, R.K.F., Bottcher, S.: Building semantic webs for e-government with wiki technology. Electronic Government, an International Journal 3(1), 36–55 (2005)

Tackling Incompleteness in Information Extraction – A Complementarity Approach

Christina Feilmayr

Johannes Kepler University Linz, Altenberger Straße 69, 4040 Linz, Austria
cfeilmayr@faw.jku.at

Abstract. Incomplete templates (attribute-value-pairs) and loss of structural and/or semantic information in information extraction tasks lead to problems in downstream information processing steps. Methods such as emerging data mining techniques that help to overcome this incompleteness by obtaining new, additional information are consequently needed. This research work integrates data mining and information extraction methods into a single complementary approach in order to benefit from their respective advantages and reduce incompleteness in information extraction. In this context, complementarity is the combination of pieces of information from different sources, resulting in (i) reassessment of contextual information and suggestion generation and (ii) better assessment of plausibility to enable more precise value selection, class assignment, and matching. For these purposes, a recommendation model that determines which methods can attack a specific problem is proposed. In conclusion, the improvements in information extraction domain analysis will be evaluated.

Keywords: Information Extraction, Data Mining, Text Mining, Interestingness Measures, Information Integration.

1 Motivation

Imperfections in information extraction (IE) manifest as negative characteristics of the information retrieved, such as *unreliability*, *ambiguity*, *uncertainty*, *inconsistency*, *incompleteness* and *imprecision*. Fully correct, complete, and precise IE from unstructured text is not feasible with current IE methods. Inaccurate IE yields noisy and incomplete datasets with many missing values. In the case of the semantic web, this means inaccurate statements predominate, resulting primarily in erroneous annotations and ultimately in inaccurate reasoning on the web.

Incomplete data collection in IE tasks, especially in scenario template (ST) production, has serious consequences in subsequent processing. Information and model quality decreases proportionally with the number of values missing from template slots. Incompleteness generally results in indecisiveness, which means that on the one hand value selection, class assignment, and matching and mapping processes in IE domain analysis are profoundly affected by incomplete knowledge sources (ontologies, lexica), missing values in template attributes (originating from upstream IE tasks, e.g., natural language processing), missing descriptive context information, and missing or incomplete constraints (e.g., quality, background, syntactical, lexical or morphological constraints) and

conditions (in terms of recurrent occurring collocations/co-occurrences, concept dependencies). On the other hand, downstream processing is affected by erroneous and incomplete template slots. Incompleteness in IE causes inaccurate inference of semantic classes and inaccurate plausibility assessment.

In this context, this thesis focuses on the incompleteness problem in IE and analyzes the applicability of complementarity, and its impact on improving the IE process.

2 Proposed Approach

While tackling the above-mentioned problems primarily necessitates a reassessment of contextual information, it also requires strategies for predicting missing values and generating suggestions for missing slot values. Consequently, methods that obtain new, additional information –such as text and data mining– are needed. Further, a means of exploiting available context information to establish meaningful constraints, conditions, and thresholds for value selection, class assignment, and the matching and mapping procedures is required. Finally, a procedure must be devised to combine the information obtained, evaluate and estimate its reliability, and incorporate the available contextual information.

2.1 Contribution of the Research Work

This research work responds to these requirements by considering the application of the *principle of complementarity* –an approach known from the field of information integration– to IE, examining the impact of redundancy on the probability of correctness. Complementarity is defined as the combination of pieces of information from different sources, taking their respective levels of reliability into account [1]. These pieces of information are the outputs of several data mining methods and text mining tasks. Therefore it enables the alignment of (intermediate) information extraction results with results from data mining methods (or rather complete text mining tasks). Using complementarity allows several data mining and text mining tasks to be integrated. Hence, the main contribution is threefold: **(i) A recommendation model**, which supports the user in selecting appropriate text mining tasks and data mining methods (in accordance with the identified incompleteness types). Integrating several information sources (complementarity) supports **(ii) efficient reassessment of contextual information** and **suggestion generation** using prediction. Analysis of different sources according to confidence and interestingness yields **(iii) better identification of material with high information potential** and leads thereby to **better plausibility assessment**. In summary, these three elements contribute to better value selection, class assignment, matching and mapping, and consequently to more robust IE results. This makes IE results become more *precise* (in statistical terms), more *certain* (in terms of confidence values) and more reliable (confirmed by human) evaluation.

2.2 The Complementarity Approach

Different types of incompleteness require different approaches to attacking the problems involved. Problems of incompleteness are subdivided into *schema-level-problems* regarding template design and *instance-level-problems* that refer to

incompleteness in the attribute-value-pair level. Consequently, incompleteness in IE is classified into missing (i) attribute values, (ii) structural information (class labels, relationships between and within templates), and (iii) semantic information (conditions, constraints, and contextual information).

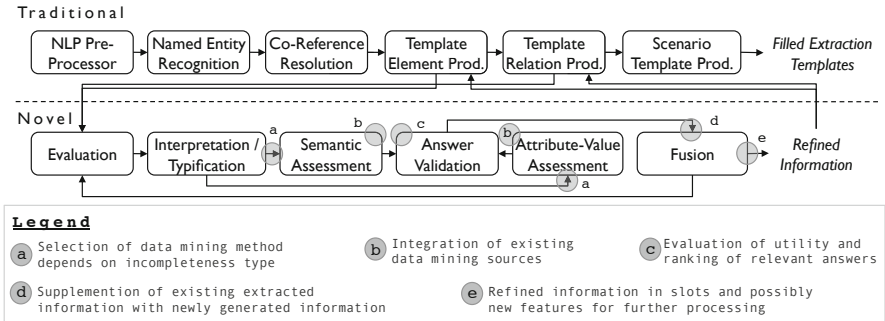


Fig. 1. General Approach of Complementarity

After the type of incompleteness has been identified, the appropriate data mining methods must be selected. As already mentioned, the thrust of the new IE approach is to integrate text and data mining into IE. Possible integration approaches are: (i) identification of collocations and co-occurrences, which can be used to resolve contradictions, perform word sense disambiguation, and validate semantic relations; (ii) constraint-based mining, which can, for example, learn different kinds of constraints (e.g., data type constraints); (iii) identification of frequent item sets (associations), which can also be used as constraints for ST and template merging; (iv) prediction provides additional evidence for missing slot values. (i)-(iii) are approaches, which are applied to assess semantics, and (iv) to assess attribute-values. New information in the form of constraints, conditions, or even suggestions for slots is generated and used to improve IE results. The results are evaluated for a selected set of features (using standard measures of the respective data mining method, e.g., accuracy, and/or selected interestingness measures [4]) that might impact the construction of answers in the presence of incompleteness. These features are combined in a flexible *utility function* (adapted from [7]) that expresses the overall value of information to a user. Determining utility means that first the utility is calculated for each answer, and second, if fusion is performed, the utility of the new value must be calculated in order to determine whether it is more appropriate than the available alternatives. Consequently, the utility value allows us to (i) define a meaningful ranking of candidates for filling incomplete templates and (ii) discover the best fusion.

A possible single point of failure is the automatic determination of the incompleteness type. Thus, the processable incompleteness types must be selected with care. Another possible limitation occurs if measures of interestingness are insufficiently meaningful: This renders the determined utility value useless and leads (depending

on the ranking and filtering methods used) to too much additional information being selected for fusion and even more contradictory, uncertain, and (semantically) imprecise information being produced.

3 Background

In the early PhD phase, the literature review focused mainly on the general idea of integrating data mining into IE. In [3], the author discussed the role of IE in text mining applications and summarized initial research work on the integration of data mining into IE. These first initiatives have been successful, but they discuss relatively simple problems. Most importantly, the projects [6], [8], and [10] demonstrate that the information extracted by such an integrated approach is of high quality (in terms of correctness, completeness, and level of interest).

To the best of the author's knowledge, there is no other research activity ongoing that deals with the integration of data mining into IE to overcome the specific problem of incompleteness. There are some well-established approaches based on complementarity in the knowledge fusion community. A general overview of knowledge fusion is given in [1]. Nikolov [9] outlined a knowledge fusion system that makes decisions depending on the type of problem and the amount of domain information available. Zeng et al. [11] implemented a classifier to acquire context knowledge about data sources and built an aggregation system capable of explaining incomplete data. Ciravegna et al. [2] proposed an approach based on a combination of information extraction, information integration, and machine learning techniques. There, methodologies of information integration are used to corroborate the newly acquired information, for instance, using evidence from multiple different sources. How to exploit redundancy in terms of IE and question answering/answer validation are described in [2] and [5], respectively.

4 Planned Research Work

The methodology comprises the following steps:

Requirements Analysis (completed). First, the requirements of IE domain analysis (regarding the incompleteness problem) had to be identified by means of an in-depth problem analysis that yielded a comprehensive summary of problematic IE issues, their intra- and interdependencies, and their impact on IE accuracy.

Classification of Data Mining Methods (in progress). Based on the requirements and problem analysis, several incompleteness types are identified and the available data mining methods classified accordingly.

Conceptual Design and Method Selection. For each incompleteness type, suitable IE and data mining methods and techniques must be selected. Based on the knowledge gained, a conceptual design for the interface between IE and data mining will be developed and a detailed conceptual design of complementarity created. Features that might impact the construction of answers in the presence of incompleteness must be identified.

Evaluation of Test Scenarios. The research work will conclude with a three-part analysis demonstrating the improvements in IE domain analysis: (i) the first part is a non-optimized information extraction process that provides the baseline; (ii) the second part integrates a gold standard for a specific problem in order to highlight the seriousness of the incompleteness problem; (iii) the third part is an information extraction process using complementarity in order to overcome the incompleteness problem. In comparison to (i) and (ii) the third part of evaluation should highlight the improvements in reducing incompleteness. Moreover, an expert evaluation is planned, which evaluate the several outcomes of complementarity modules.

Acknowledgement. This work is supported by an Austrian research grant (FIT-IT Semantic Systems Dissertation Fellowship Project) from BMVIT (project nr. 829601).

References

1. Bloch, I., Hunter, A., et al.: Fusion: General Concepts and Characteristics. *International Journal of Intelligent Systems, Special Issue: Data and Knowledge Fusion* 16(10), 1107–1134 (2001)
2. Ciravegna, F., Chapman, S., Dingli, A., Wilks, Y.: Learning to Harvest Information for the Semantic Web. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) *ESWS 2004*. LNCS, vol. 3053, pp. 312–326. Springer, Heidelberg (2004)
3. Feilmayr, C.: Text Mining Supported Information Extraction - An Extended Methodology for Developing Information Extraction Systems. In: *Proceedings of 22nd International Workshop on Database and Expert Systems Applications (DEXA 2011)*, pp. 217–221 (2011)
4. Geng, L., Hamilton, H.J.: Interestingness Measures for Data Mining: A Survey. *ACM Computing Surveys* 38(3), Article 9 (2006)
5. Magnini, B., Negri, M., Prevete, R., Tanev, H.: Is It the Right Answer? Exploiting Web Redundancy for Answer Validation. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pp. 425–432 (2002)
6. McCallum, A., Jensen, D.: A Note on the Unification of Information Extraction and Data Mining using Conditional-Probability, Relational Models. In: *Proceedings of the IJCAI Workshop on Learning Statistical Models from Relational Data* (2003)
7. Motro, A., Anokhin, P., Acar, A.C.: Utility-based Resolution of Data Inconsistencies. In: *Proceedings of the International Workshop on Information Quality in Information Systems*, pp. 35–43 (2004)
8. Nahm, U.Y., Mooney, R.J.: Using Soft-Matching Mined Rules to Improve Information Extraction. In: *Proceedings of the AAAI 2004 Workshop on Adaptive Text Extraction and Mining*, pp. 27–32 (2004)
9. Nikolov, A.: Fusing Automatically Extracted Annotations for the Semantic Web, PhD Thesis, Knowledge Media Institute, The Open University (2009)
10. Wong, T.-L., Lam, W.: An Unsupervised Method for Joint Information Extraction and Feature Mining Across Different Web Sites. *Data & Knowledge Engineering* 68(1), 107–125 (2009)
11. Zeng, H., Fikes, R.: Explaining Data Incompleteness in Knowledge Aggregation, Technical Report, Knowledge Systems, AI Laboratory, KSL-05-04 (2005)

A Framework for Ontology Usage Analysis

Jamshaid Ashraf

School of Information Systems, Curtin University, Perth, Western Australia
jamshaid.ashraf@gmail.com

1 Research Problem and Motivation

The Semantic Web (also known as the Web of Data) is growing rapidly and becoming a decentralized social and knowledge platform for publishing and sharing information. In the early days of the Semantic Web (1999-2006), research efforts of the community were centered around knowledge representation; thus, most of the research work was focused on building ontologies (ontology engineering), developing formal languages to represent them (ontology language), methodologies to evaluate and evolve ontologies (ontology evaluation and evolution (OE)), and logic for reasoning with them. As a result of this, even though ontologies were being developed but their instantiation was inadequate to provide the actual instance data needed for the evaluation and analysis of the developed ontologies. In order to overcome this issue, test data was often used to perform the above tasks [1]. However, in the recent past, the focus has shifted towards publishing data either with little or no use of ontologies [2]. This shift in focus is credited to the Linked Open Data (LOD) Project which has published billions of assertions on the Web using well known Linked Data principles. Because of this, the research focus has shifted from knowledge-centered to data-centered and is now settling down at the point where domain ontologies are being used to publish real-world data on the Web. This trend promotes consistent and coherent semantic interoperability between users, systems and applications. In this regard, several domain ontologies have been developed to describe the information pertaining to different domains such as Healthcare and Life Science (HCLS), governments, social spaces, libraries, entertainment, financial service and eCommerce.

According to the PingTheSemanticWeb.com which maintains a list of namespaces used in RDF documents, there are around 1810 namespaces (URIs) of ontologies/vocabularies currently being used on the Web (as of 12th Jan. 2012). However, while there are several ontologies being used, there is no formal approach to **evaluate, measure, and analyse the use of ontologies on the Web**. Such a study is very important to (a) *make effective and efficient use* of formalized knowledge (ontology) available on the web, (b) provide a *usage-based feedback loop* to the ontology maintenance process for a pragmatic conceptual model update, and (c) provide erudite *insight on the state of semantic structured data*, based on the prevalent knowledge patterns for the consuming applications. In the absence of such analysis, even though we may have some techniques to deal with information overload, but they may not provide us with efficient knowledge synthesizing techniques that are important to fully realize the potential of distributed knowledge space.

The presence of thousands of ontologies and billions of triples representing real-world data, now, provides the perfect foundation to carry out empirical studies to analyse the use of ontologies. Based on the aforementioned discussion, the aim of this PhD research is to develop a semantic framework for measuring and analyzing ontology usage known as an Ontology USage Analysis Framework (OUSAF). The proposed framework will be equipped with a set of metrics that measures qualitative and quantitative aspects of ontology usage, providing with important analysis on various detailed factors.

2 State of the Art

Related work in this area can be classified into two areas: first, work which focuses merely on knowledge (ontologies) and second, work which addresses RDF-data-related issues such as management, quality and usefulness. Pertinent to knowledge management and closely related to this research work is Ontology Evaluation (OE), which validates and verifies the developed ontology to measure the extent to which it serves and fits the intended purpose. Existing OE approaches focus mainly on evaluating the developed ontology and do not provide insight into how the ontology is utilised and adopted by its eventual users. As reported in [3], despite the fact that at present there are thousands of developed ontologies, very few of them are well populated and widely used in real-world applications. On the data-centered side, researchers have focused on evaluating the nature, quality and patterns of the RDF data published in response to the LOD project. For example in [4], while evaluating the quality, noise and inconsistency present in RDF data, the authors have provided guidelines for both publishers and consumers to assist in improving the quality and usefulness of data.

3 Methodology and Approach

The key step of the framework is to define the set of metrics to measure the usage of the ontologies and data and identify the emerging knowledge patterns. The metrics and measures that will be proposed in this research will evaluate the dataset from two aspects, the first of which is semanticity in which the presence and use of terminological knowledge in the dataset will be looked at by defining quantitative measures. The second aspect is the structurality whereby the knowledge base is conceived as an Ontology Usage Network, modeled using an affiliation network to understand and measure applicable social network properties such as centrality, degree and association (see Figure 1). The theoretical exploration and development of the framework is divided into the following sub-stages:

Construction of RDF dataset: In order to conduct an empirically grounded study, it is important to collect the real data that instantiate the ontologies on the Web. For increased effectiveness, the dataset is required to have two essential characteristics: 1) it should comprise real-world data collected from the Web, and 2) data should be collected during different time intervals to capture the changes and trends in knowledge patterns over time.

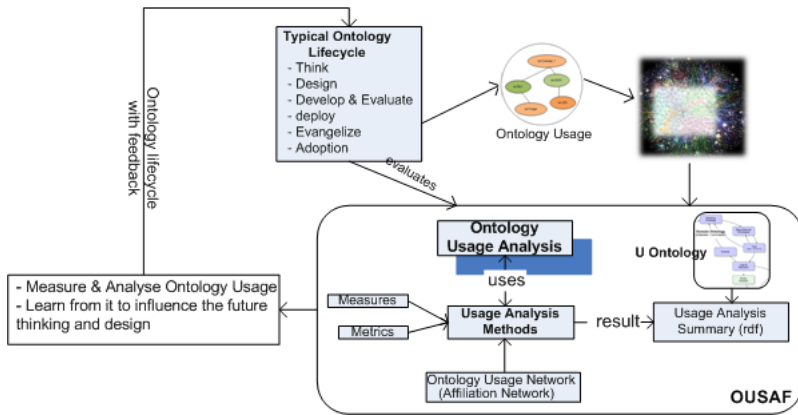


Fig. 1. Ontology Usage analysis Framework and Ontology Lifecycle with feedback loop

Define metrics and measurements: The framework supports empirical analysis from two perspectives: the ontology perspective and the RDF data perspective. From the ontology perspective, ontology is considered as an engineering artifact (ontology document) to characterize the components defined in a document such as vocabulary, hierarchical and non-hierarchical structure, axioms and attributes. From the RDF data perspective, we analyse the RDF triples to understand the patterns and the structure of the data available in the dataset. Metrics developed in OUSAF are grouped into three categories, namely, concept-centric metrics, relationship-(object property) metrics and attribute-(data property) metrics.

- *Concept-Centric Metrics:* In concept-centric metrics, the structure of each concept is considered in order to determine its importance within the ontology. Concept Richness, Concept Usage and Concept Population are potential metrics to measure the concept-centric metrics. These metrics help in measuring the concept instantiation and the information available with these instances.
- *Relationship and Attribute Centric Metrics:* Relationship-centric metrics aim to measure the relationship, attribute richness and usage of concepts in a knowledge base. To achieve this, metrics such as Relationship Value, Relationship Usage, Attribute Value and Attribute Usage are employed.

Furthermore, to increase the utilization of ontology usage, the analysis results will be conceptualized using Ontology Usage Ontology (U Ontology), as shown in Figure 1.

Structural Properties of the Ontology Usage Network: The aim of this step is to analyse the topological properties of the ontology usage network by extending the affiliation network model. We propose the construction of an on-

tology/vocabulary usage network as a bipartite network in which nodes are divided into two sets with links (considered as edges in our model) only between the nodes of different sets. A bipartite network provides the representation of our usage model and helps us to study general purpose network properties [5]. To topologically analyse the domain ontology usage in the web of data, several structural properties such as centrality (whether a network has a 'center' point or points), reciprocity (whether ties look identical from either end), density (how many potential ties in a network actually exist), and reachability (how many ties it takes to go from one 'end' of a network to the opposite 'end') will be studied to capture the topological aspect of ontology usage.

Evaluation: The proposed OUSAF framework will be evaluated by accessing Semantic Web data using the knowledge patterns generated by the usage analysis. Furthermore, the ontology usage summary will be used to auto-generate the prototypical queries to access the relevant information from the knowledge base and observe the precision and recall.

4 Initial Results and Proposed Benefits

To understand the nature of the structured (RDF) data published on the web, the domain ontologies used and their co-usability factor, the use of semantic web technologies and plausible reasoning (how much implicit knowledge is inferable), in [6] we considered the e-commerce dataset (called GRDS) by crawling the web. The latest version of the dataset comprises of 27 million triples collected from approximately 215 data sources. In [6], we performed empirical analysis on the dataset to analyse the data and knowledge patterns available and found that a small set of concepts (lite ontology) of the original model is, in fact, used by a large number of publishers. We also learnt that, with current instance data, there is not much for RDFS reasoner(s) to infer implicit knowledge due to the invariant data and knowledge patterns in the knowledge base. Based on the visibility obtained, in [7] we have proposed a framework and metrics to measure the concepts and property usage in the dataset by keeping in view their richness within the ontological model. In [8], the ontology usage framework is used to extract the web schema, based on the ontology instantiation and co-usability in the database. The web schema represents the prevailing schema, providing the structure of data useful for accessing information from the knowledge base and building data-driven applications. Based on the research done so far and the initial results obtained, we are confident of the benefits that can be realized with the implementation of OUSAF, which include: (a) assisting in building Semantic Web applications to offer rich data services by exploiting the available schema level information and assisting in providing an improved context driven user interface and exploratory search [9]. based on auto discovery of explicit and implicit knowledge; (b) enabling client applications to make expressive queries to the Web by exploiting the schema patterns evolving through the use of ontologies; and (c) empirical analysis of domain ontology usage, as shown in Figure 1.

that provides the feedback loop to the ontology development life cycle. Knowing the sub-model of the original ontology, which provides information about usage and adoption, will assist the ontology developer to pragmatically refine, update and evolve the conceptual model of ontology.

5 Conclusion and Future Work

The objective of this research is to design and implement a semantic framework to evaluate and analyse the usage of domain ontology. From a high level view, given a domain ontology and dataset, we would like to analyse the usage of ontology qualitatively and quantitatively, its co-usability factor with other ontologies and plausible reasoning. In future work, I will construct the dichotomized one mode ontology-by-ontology co-usability matrix to have the ontology linkage graph. Furthermore, the usage patterns represented through U Ontology will be further used to develop the access layer for the web-of-data (see Figure [11](#)).

References

1. Tao, J., Ding, L., McGuinness, D.L.: Instance data evaluation for semantic web-based knowledge management systems. In: HICSS, pp. 1–10. IEEE Computer Society (2009)
2. Jain, P., Hitzler, P., Yeh, P.Z., Verma, K., Sheth, A.P.: Linked data is merely more data. In: AAAI Spring Symposium: Linked Data Meets Artificial Intelligence. AAAI (2010)
3. Ding, L., Zhou, L., Finin, T., Joshi, A.: How the semantic web is being used: An analysis of foaf documents. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences - Track 4, vol. 4, pp. 113–120. IEEE Computer Society, Washington, DC (2005)
4. Hogan, A., Harth, A., Passant, A., Decker, S., Polleres, A.: Weaving the pedantic web. In: Linked Data on the Web Workshop (LDOW 2010) at WWW 2010 (2010)
5. Borgatti, S.P., Everett, M.G.: Network analysis of 2-mode data. *Social Networks* 19(3), 243–269 (1997)
6. Ashraf, J., Cyganiak, R., ORiain, S., Hadzic, M.: Open ebusiness ontology usage: Investigating community implementation of goodrelations. In: Linked Data on the Web Workshop (LDOW 2011) at WWW 2011, Hyderabad, India, March 29 (2011)
7. Ashraf, J., Hadzic, M.: Domain ontology usage analysis framework. In: SKG, pp. 75–82. IEEE (2011)
8. Ashraf, J., Hadzic, M.: Web schema construction based on web ontology usage analysis. In: JIST. Springer (2011)
9. Tvarožek, M.: Exploratory search in the adaptive social semantic web. *Information Sciences and Technologies Bulletin of the ACM Slovakia* 3(1), 42–51 (2011)

Formal Specification of Ontology Networks

Edelweis Rohrer

Instituto de Computación, Facultad de Ingeniería,
Universidad de la República, Uruguay
erohrer@fing.edu.uy

Abstract. Nowadays, it is very often to integrate existing ontologies, combining them in a ontology network to accomplish the requirements of more complex applications. This PhD research¹ aims to identify and formally define the relationships among the networked ontologies, addressing its use in real applications and taking care of their consistency.

Keywords: ontology network, ontology relationships formalization, logical consistency.

1 Motivation and Research Questions

Nowadays, autonomously developed ontologies in different domains (health, learning) are used together in complex applications. However, how they are combined is usually hidden in the application code. This situation leads to think on *ontology networks* as a new engineering concept, which explicitly expresses how ontologies are combined. Let suppose a scenario involving several domains, such as a web resource recommender system (Figure 1). The *Resources* domain describes web contents queried by users, the *Quality* domain, the quality assessment process of web resources, the *User Context* domain, the user profile and context and the *Criteria Selection* domain, the criteria used to recommend a given resource to a user. In this example, the relationship between *Quality* and

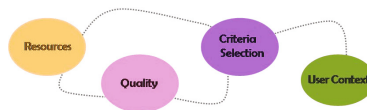


Fig. 1. A Recommendation Ontology Network

Resources domains appears since in this case study, web resources are assessed according to some quality criteria. Then, it is important to explicitly specify not only the semantics of each domain, but moreover adding knowledge about how

¹ Tutored by Regina Motz of Instituto de Computación, Facultad de Ingeniería, Universidad de la República, Uruguay and Alicia Díaz of LIFIA, Facultad de Informática, Universidad Nacional de La Plata, Argentina.

these domains are related. The main motivation of this thesis is the identification and formal definition of the different relationships among ontologies, to describe a particular application, keeping the logical consistency. That is, there should not be an axiom of an ontology that causes contradictory results over another ontology in an ontology network. In a real application, the ontology network consistency could be computationally hard to be checked, then, the trade-off between keeping the consistency and taking care of the computational properties, is one of the main issues of this work. The main contribution will be to facilitate developers in the design of ontology networks, expliciting how ontologies can be linked, keeping them as independent components. In the remainder of this paper: Section 2 gives a background overview, Section 3 explains the PhD approach, Section 4 introduces methodology issues and Section 5 presents the work already done.

2 State of the Art

According to the presented motivation, I take as starting point the work of Allocca et al. [1], who identify and define general relations between ontologies, such as *includedIn* and *equivalentTo*, describing them in the DOOR ontology.

Grau et al. [2] define an ε - *connection* as a “set of connected ontologies”, introducing *link properties*, which connect two ontologies. The semantic of these properties is like the *useSymbolsOf* relationship, defined in the PhD work. I also based my study in a more recent work of Grau et al. [3], which adapts the notions of *module* and *black-box behavior*, to the reuse of ontologies. Konev et al. [4] analyze the same concepts and others such as *robustness of a query language*, based on the concept of *inseparability* of ontologies. These two works [3,4] also analyse the computational complexity issue for Description Logics (DL) with different expressivity, so, I am taking advantage of their results.

The work of Borgida et. al [5] defines directional links between ontologies, called *bridge rules* and the concept of *distributed T-box*, DL T-boxes connected through bridge rules. The bridge rules capture the idea of linking ontologies through subsumption as well as more general relationships, while my work intends to clearly distinguish different ways of connecting ontologies, to make them explicit.

Giunchiglia et al. [6] define the concept of *abstraction* without relating it to ontologies. However, I take this idea to define the *isTheSchemaFor* relationship.

There also exist works that define the *ontology mapping* between concepts, roles and instances [7,8], taken to formalize the relationship *mapsSymbolsTo*.

3 The Proposed Approach

I have formalized a set of *ontology relationships*, which allowed me to design ontology networks for some case studies. For these case studies, this set of relationships was adequate to explicitly express the links among the different domain ontologies. Next, I give an intuitive description of each ontology relationship.

isAConsistentExtensionOf: describes an extension of an ontology by a number of additional axioms.

usesSymbolsOf: this relationship holds when an ontology O needs to be linked to individuals from another ontology O' , through a property which relates them.
mapsSymbolsTo: an ontology O *mapsSymbolsTo* an ontology O' if there exists an alignment from O to O' , covering part of the vocabulary of O .
isTheSchemaFor: keeps the link between a model and its meta-model.

In the web resource recommender system introduced in Section 11, the *Resources* domain is composed by three ontologies, illustrated in Figure 2 in a simplified version. The main concepts of the *WebSite Specification* ontology are

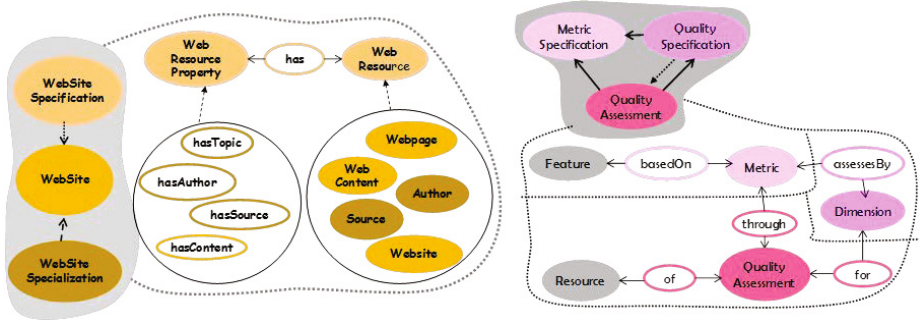


Fig. 2. Resources and Quality domains

WebResource and *WebResourceProperty*. A web resource is any resource identified by an URL, for instance a webpage. A web resource property models the properties of a web resource, for instance, *hasContent* and *hasAuthor*. The *WebSite* ontology has as main concepts: *WebContent*, *WebPage* and *WebSite*. The *WebSite Specialization* ontology adds properties to these concepts, such as *hasAuthor* and *hasSource*. In the Resources domain the *isTheSchemaFor* relationship links some ontologies. The *WebSite Specification* ontology is the meta-model for the *Website* and *WebSite Specialization* ontologies, since the concepts and relations of these ontologies are instances of *WebResource* and *WebResourceProperty* concepts.

The *Quality* domain is composed by three ontologies: *Metric Specification*, *Quality Specification* and *Quality Assessment*. They conceptualize metrics, quality assurance specifications and quality assessments. Figure 2 shows a simplified version. Some of these ontologies are related to ontologies of the *Resources* domain. The *mapSymbolsTo* relationship links the *Quality Assessment* and *WebSite* ontologies, through an alignment of the *Resource* and *WebContent* concepts. This relationship is also used between *Metric Specification* and *WebSite Specification* ontologies, mapping the *Feature* and *WebResourceProperty* concepts, to specify that a metric is based on some web resource property. Here, it is clear the convenience of having some ontologies that play the role of metamodel for others.

In the formalization of the ontology relationships I consider the language \mathcal{QL} , which is the selected DL with the adequate expressivity for the application

to be described. That is, besides the knowledge represented by each ontology, I consider the expressivity required by the application, for the knowledge inference of the ontology network. For a case study, maybe it is enough a DL, for example \mathcal{ALC} , and not a more expressive DL like \mathcal{ALCQ} , computationally more expensive [9,10]. I am addressing the study of the logical consistency of the *ontology network* based on the concept of *inseparability* introduced by Konev et.al [4], w.r.t. this language \mathcal{QL} and I am starting to study the computational complexity of the algorithms for checking the consistency. The results obtained will be analyzed varying the DL expressivity of the \mathcal{QL} , for the different relationships.

Although my work is inspired on [1], it is different since the main focus in [1] is the detection and definition of ontology relationships in a large ontology repository, while my focus is the identification and DL formalization of a set of ontology relationships, enough to design an ontology network for a particular application. This is done considering the logical consistency of the ontology network as well as computational complexity issues. A tool to design ontology networks allowing modelers to define different relationships, can benefit from the formalization.

4 Research Methodology

This work is being carried out following an iterative process. I started with the analysis of case studies to identify ontology relationships. This led to investigate the way other authors addressed this issue, reviewing theoretical foundations about DL and computational complexity when necessary. As a result, a set of relationship definitions is obtained, which is validated in a case study, and from the weaknesses found a new iteration starts, refining the previous definitions.

Regarding the evaluation of the approach, the implementation of an application to design ontology networks is being carried out. It will allow to validate a lot of important aspects: (i) its usability to define different relationships, reaching the adequate abstraction level (ii) the evaluation of the user satisfaction when the ontology network evolves. Here, it is important to know about the imposed restrictions for ensuring the ontology network consistency: if they help at the moment of introducing changes or they difficult the task in practice.

5 Results

I have formalized four ontology relationships, introduced in Section 3. A first formalization and its use to describe a web recommender system was presented in [11]. In the following, I present the *usesSymbolsOf* relationship.

First, I define a *relationship between two ontologies O and O' w.r.t. a query language \mathcal{QL}* as a set of axioms A_r , called *relationship axioms* such that:

$A_r \subseteq \{\alpha \in \mathcal{QL} \mid sig(\alpha) \subseteq sig(O) \cup sig(O') \cup S_r\}$ where

$S_r \subseteq \{X \mid X \in N_C \cup N_R \cup N_I\}$ is called the *relationship signature* with:

N_C the set of all the concept names, N_R the set of all the role names, N_I the set of all the individual names

$$S_r \cap sig(O \cup O') = \emptyset$$

usesSymbolsOf(O, O', \mathcal{QL}) is defined by a set of *relationship axioms* A_r such that:

$S_r \subseteq \{r \mid r \in N_R\}$ is the relationship signature, $sig(O) \cap sig(O') = \emptyset$

$A_r \subseteq \{r(i, j) \mid r \in S_r, i \in N_I \cap sig(O), j \in N_I \cap sig(O')\} \cup \{A \sqsubseteq C \mid A \in N_C \cap sig(O), C \text{ is a concept description of one of the forms: } \exists r.B, \forall r.B, \geq nr.B \text{ with } r \in S_r, B \in N_C \cap sig(O'), n \text{ a natural number}\}$, $A_r \neq \emptyset$

$O \cup A_r$ and O' are S-inseparable w.r.t. \mathcal{QL} for $S = sig(A_r) \cap sig(O')$

$O \cup A_r$ and O are S-inseparable w.r.t. \mathcal{QL} for $S = sig(O)$

The two last statements ensure the consistency, preventing contradictory results over the symbols of O' being used and over the ontology O , extended by the set of axioms A_r .

These relationships have been addressed by different authors separately, not always related to ontologies, some of them taking care of the logical consistency. This work intend to uniformly address the definition of a set of ontology relationships, enough to describe a real application, ensuring its consistency without neglecting complexity issues. I think this work will contribute to the definition of a methodology to design ontology networks.

References

1. Allocca, C., D'Aquin, M., Motta, E.: DOOR - Towards a Formalization of Ontology Relations. In: Dietz, J.L.G. (ed.) KEOD, pp. 13–20. INSTICC Press (2009)
2. Cuenca Grau, B., Parsia, B., Sirin, E.: Combining OWL Ontologies Using ε -Connections. Web Semantics: Science, Services and Agents on the World Wide Web 4, 40–59 (2006)
3. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Modular Reuse of Ontologies: Theory and Practice. J. Artif. Intell. Res (JAIR) 31, 273–318 (2008)
4. Konev, B., Lutz, C., Walther, D., Wolter, F.: Formal Properties of Modularisation. In: Stuckenschmidt, H., Parent, C., Spaccapietra, S. (eds.) Modular Ontologies. LNCS, vol. 5445, pp. 25–66. Springer, Heidelberg (2009)
5. Borgida, A., Serafini, L.: Distributed Description Logics: Assimilating Information from Peer Sources. Journal of Data Semantics 1, 153–184 (2003)
6. Giunchiglia, F., Walsh, T.: A Theory of Abstraction. Journal Artificial Intelligence 56, 323–390 (1992)
7. Ehrig, M.: Ontology Alignment. Bridging the Semantic Gap. Springer Science+Business Media, LLC (2007)
8. Suchanek, F.M., Abiteboul, S., Senellart, P.: Ontology Alignment at the Instance and Schema Level. Technical report, Institut National de Recherche en Informatique et en Automatique (2011)
9. Baader, F., Nutt, W.: Basic Description Logics. In: Baader, et al. [12], pp. 47–95
10. Donini, F.M.: Complexity of Reasoning. In: Baader, et al. [12], pp. 101–138
11. Díaz, A., Motz, R., Rohrer, E.: Making Ontology Relationships Explicit in a Ontology Network. In: The V Alberto Mendelzon International Workshop on Foundations of Data Management (May 2011)
12. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)

Leveraging Linked Data Analysis for Semantic Recommender Systems

Andreas Thalhammer

STI Innsbruck, University of Innsbruck, Austria
andreas.thalhammer@sti2.at

1 Motivation

Traditional (Web) link analysis focuses on statistical analysis of links in order to identify “influential” or “authoritative” Web pages like it is done in PageRank, HITS and their variants [10]. Although these techniques are still considered as the backbone of many search engines, the analysis of usage data has gained high importance during recent years [12]. With the arrival of linked data (LD), in particular Linked Open Data (LOD) [1] new information relating to *what* actually connects different vertices is available. This information can be leveraged in order to develop new techniques that efficiently combine linked data analysis with personalization for identifying not only relevant, but also diverse and even missing information. Accordingly, we can distinguish three problems that motivate the topic of this thesis:

Relevance. LOD is well known for providing a vast amount of detailed and structured information. We believe that the information richness of LOD combined with user preferences or usage data can help to understand items and users in a more detailed way. In particular, LOD data can be the basis for an accurate profile which can be useful for recommendation in various domains. As information about items in the user profile is often unstructured and contains only little background knowledge, this information needs to be linked to external sources for structured data such as DBpedia [2]. Also, product and service providers need to link their offers accordingly. Methods for this two-way alignment have to be specified and evaluated.

Diversity. According to [5], recent developments in semantic search focus on contextualization and personalization. However, approaches that semantically enable diverse recommendations for users, also in context to users’ profiles, remain barely explored. Of course, this states a complementary way of recommendation that is often only based on ranking by relevance. Consider the example of a news aggregation Web site which ranks articles by popularity. Popular articles are placed at the main page. On the same topic, there are hundreds of additional articles from other news sites and blogs indexed, but not visible on the main page. Of course, these articles get much

¹ Linking Open Data - <http://ow.ly/8mPMW>

² DBpedia - <http://dbpedia.org/>

less clicks than the ones from the main page. This results in the most popular sites gaining even more popularity. Our goal is to break up such feedback loops by introducing diverse recommendations.

Non-existence. During recent years the LOD cloud³ has been growing to a huge amount of interconnected triples. Approaches like Freebase⁴ and the Wikidata project⁵ focus on collecting information through direct input in order to establish an encyclopedic corpus. We believe that these systems are likely to face the same issue like Wikipedia:⁶ since 2006, its growth is decreasing [16]. This problem of Wikipedia gained focus of research and different article recommendation systems have been explored [4,8]. These systems point users to articles they might want to edit. This idea can be extended in order to work for Freebase or Wikidata: given the corresponding user profiles, it becomes feasible to point users to missing facts (e.g. the mayor of a city).

2 Related Work

During the last decade, several approaches that aim to link semantic technologies and recommender systems have been introduced. [15] introduces a framework that enables semantically-enhanced recommendations in the cultural heritage domain. Recommendation as well as personalization in this work rely on the CHIP ontology which is designed specifically for the cultural heritage domain. The core of the recommendation strategy bases on discovering domain-specific links between artworks and topics (e.g. the same creator, creation site, or material medium). In the outlook section of this work, the author emphasizes on LD as a core technology to enhance personalization and recommendation. The work presented in [7] states an approach to utilize LD in order to enhance recommender systems. Just like our focus on linking items in the user profile to LOD items, Heitmann and Hayes utilize LOD links in order to enhance background information for recommendation corpora. The recommendation approach is collaborative filtering (cf. [1]) as “the inconsistent use of these semantic features makes the cost of exploiting them high” [7]. In my thesis, I try to leverage exactly these semantic features for recommendation. [2] introduces a semantic news recommender system called “News@hand” that makes use of ontology-based knowledge representation in order to mitigate the problem of ambiguity and to leverage reasoning for mediating between fine and coarse-grained feature representations. The system supports content-based as well as collaborative recommendation models. Similar to our approach, the items and the user profiles are represented as a set of weighted features. The weights of the item features are computed with a TF-IDF technique which does not involve additional knowledge.

³ LOD cloud - <http://lod-cloud.net/>

⁴ Freebase - <http://www.freebase.com/>

⁵ Wikidata - <http://meta.wikimedia.org/wiki/Wikidata>

⁶ Wikipedia - <http://wikipedia.org/>

DBrec, a music-specific recommendation approach, is presented in [13]. In this paper, LD (i.e. DBpedia) is utilized for semantic recommendations. The approach is based on the similarity measure LDSO which aims at estimating the distance between two LD resources by considering entities in a two-hop radius. Just like in item-based collaborative filtering (cf. [1]), the similarity scores between entities can be computed offline. For storing the results, the author introduces the LDSO ontology. In comparison to the similarity measures we try to introduce, LDSO similarity computation is based on statistics about direct and indirect in and out links disregarding the importance of particular predicates.

Personalized property suggestions relate to two topics, i.e. personalized article recommender systems and property suggestion systems. Several recommender systems utilize user information to make recommendations like “You might want to edit this Wikipedia article” [48]. Others suggest facts that might be relevant for a certain Wikipedia entry [11][17]. However, the problem of proposing properties for Wikipedia, Freebase, or OntoWiki documents in a personalized way, like “Here are some missing facts about [some article] that you could know”, has not been addressed so far.

3 Proposed Approach

Our approach is based on identifying distinctive item features with the help of usage or rating data. As with all recommender systems, the main goal is to help users to find information that is important to them. On a different level, the macro goals are to identify and match information that is important about users with information that is important about items. Accordingly, the first part of the process can be broken down to the following steps:

1. Extract and create user profiles with items linked to LOD.
2. Find k-nearest neighbors for each item according to the user-item matrix. The entries of this matrix are ratings by users for items (cf. [14]).
3. Identify which features are important about specific items (which property-value pairs do they share with their neighbors).
4. Combining the results step 1 and 3, it is possible to find out what is specific about a user.

These processing steps can be performed offline. For the representation and storage of the results an appropriate vocabulary needs to be selected. After these steps, we have established a situation where we know what is important about specific items as well as users. Afterwards, in the second part, we investigate for different match-making techniques that help us to recommend items to the user that relate to relevant, diverse, or missing information. Accordingly, the following techniques may serve as starting points:

Relevance. Graph matching of the weighted user and item graphs.

Diversity. Clustering according to the most important properties or property-instance combinations in the user profile.

Non-existence. Recommend a missing fact which is very important for an article (e.g. population of a city) to the user who is frequently editing this property in related articles.

Some efforts towards semantically enabled cross-domain profiles and recommendations that refer to LOD have already started [6]. However, utilizing user profile or usage data in order to introduce feature weights for items has not been explored to the best of my knowledge. Also, using collaborative filtering background knowledge in order to discover important properties of LOD entities is a novel approach. There might also be a limitation of the approach: The user-item matrix that we make use of can be used directly for collaborative filtering. In previous experiments that were conducted by various users in connection with the Netflix Prize⁷ it turned out that sole collaborative filtering outperformed all approaches that tried to enrich the data set with background information. The pure collaborative filtering approach was much better in performance and also in terms of prediction quality. However, we target scenarios that are not constrained by a single fixed domain (such as movies). Moreover, measuring relevance only is not comparable to our scenarios as we also try to incorporate diversity. Given a specific domain, collaborative filtering can serve as a base line for our relevance-focused approach.

4 Methodology

For my thesis, I will conduct the following steps:

- identify related fields of research
- implement linking of items to LOD and the according weighted item and user representations
- identify existing match-making algorithms and evaluate their appropriateness
- conduct different algorithms for match-making
- evaluate the selected approaches according to relevance, diversity and estimated information gain

For first tests and results, we chose the HetRec2011 MovieLens2k dataset [3] that has been linked to Freebase data. The rating data stems from the MovieLens10M dataset⁸ that contains anonymous user profiles.

The evaluation of recommender systems is usually based on precision and recall which can also be applied in this case. In this field, a couple of approaches already exist that can serve as base lines. A measure which ranks diversity is introduced in [9]. The recommendation of missing content for Web 2.0 collections can be evaluated by comparing the number of edits with and without the recommendation approach.

⁷ Netflix Prize - <http://www.netflixprize.com/>

⁸ MovieLens10M - <http://www.grouplens.org>

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering* 17, 734–749 (2005)
2. Cantador, I., Bellogín, A., Castells, P.: Ontology-based personalised and context-aware recommendations of news items. In: *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, WI-IAT 2008*, vol. 1, pp. 562–565. IEEE Computer Society, Washington, DC (2008)
3. Cantador, I., Brusilovsky, P., Kuflik, T.: 2nd ws. on information heterogeneity and fusion in recommender systems (hetrec 2011). In: *Proc. of the 5th ACM Conf. on Recommender Systems, RecSys 2011*. ACM, New York (2011)
4. Cosley, D., Frankowski, D., Terveen, L., Riedl, J.: SuggestBot: Using Intelligent Task Routing to Help People Find Work in Wikipedia. In: *Human-Computer Interaction* (2007)
5. Dengel, A.: Semantische suche. In: Dengel, A. (ed.) *Semantische Technologien*, pp. 231–256. Spektrum Akademischer Verlag (2012)
6. Fernández-Tobías, I., Cantador, I., Kaminskas, M., Ricci, F.: A generic semantic-based framework for cross-domain recommendation. In: *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems, HetRec 2011* (2011)
7. Heitmann, B., Hayes, C.: Using Linked Data to Build Open, Collaborative Recommender Systems. *Artificial Intelligence* (2010)
8. Huang, E., Kim, H.J.: Task Recommendation on Wikipedia. *Data Processing* (2010)
9. Murakami, T., Mori, K., Orihara, R.: Metrics for Evaluating the Serendipity of Recommendation Lists. In: Satoh, K., Inokuchi, A., Nagao, K., Kawamura, T. (eds.) *JSAI 2007. LNCS (LNAI)*, vol. 4914, pp. 40–46. Springer, Heidelberg (2008)
10. Ng, A.Y., Zheng, A.X., Jordan, M.I.: Stable Algorithms for Link Analysis. *Machine Learning*, 267–275 (2001)
11. Oren, E., Gerke, S., Decker, S.: Simple Algorithms for Predicate Suggestions Using Similarity and Co-occurrence. In: Franconi, E., Kifer, M., May, W. (eds.) *ESWC 2007. LNCS*, vol. 4519, pp. 160–174. Springer, Heidelberg (2007)
12. Pariser, E.: *The filter bubble: what the Internet is hiding from you*. Viking, London (2011)
13. Passant, A.: dbrec — Music Recommendations Using DBpedia. 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. 209–224. Springer, Heidelberg (2010)
14. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: *Proceedings of the 10th International Conference on World Wide Web, WWW 2001*, pp. 285–295. ACM, New York (2001)
15. Wang, Y.: *Semantically-Enhanced Recommendations in Cultural Heritage*. PhD thesis, Technische Universiteit Eindhoven (2011)
16. Wikipedia. Modelling wikipedia’s growth, http://en.wikipedia.org/wiki/Wikipedia:Modelling_Wikipedia's_growth (online accessed March 12, 2012)
17. Zhang, H., Fu, L., Wang, H., Zhu, H., Wang, Y., Yu, Y.: Eachwiki: Suggest to be an easy-to-edit wiki interface for everyone. In: *Semantic Web Challenge* (2007)

Sharing Statistics for SPARQL Federation Optimization, with Emphasis on Benchmark Quality

Kjetil Kjernsmo

Department of Informatics, Postboks 1080 Blindern, 0316 Oslo, Norway
kjekje@ifi.uio.no

Abstract. Federation of semantic data on SPARQL endpoints will allow data to remain distributed so that it can be controlled by local curators and swiftly updated. There are considerable performance problems, which the present work proposes to address, mainly by computation and exposure of statistical digests to assist selectivity estimation.

For an objective evaluation as well as comparison of engines, benchmarks that exhaustively covers the parameter space is required. We propose an investigation into this problem using statistical experimental planning.

1 Motivation

Query federation with SPARQL, which is a standardized query language for the Semantic Web, has attracted much attention from industry and academia alike, and four implementations of basic query federation were submitted to the SPARQL 1.1 Working Group as input for the forthcoming work¹. This feature was supported by many group members, and the Last Call working draft of the proposed standard was published on 17 November 2011.

While the basic feature set of the proposed standard can enable users to create federated queries, it is not of great use as it requires extensive prior knowledge of both the data to be queried and performance characteristics of the involved query engines. Without this knowledge, the overall performance is insufficient for most practical applications.

To aid optimization, SPARQL endpoints should expose details about both data and performance characteristics of the engine itself. The proposed work has two focal points: *Statistical digests of data for optimizations* and *benchmarking SPARQL engines*.

The focus on SPARQL benchmarking is not only motivated from the perspective of optimization, as I have found the current state of the art in SPARQL benchmarking lacking in its use of statistics. The emphasis in the present paper is on statistics in benchmarking with the purpose of providing a firmer foundation on which assertions about engine performance can be backed with evidence.

¹ <http://www.w3.org/2009/sparql/wiki/Feature:BasicFederatedQuery>

The prior art in database theory is extensive, but I intend to focus on aspects that sets SPARQL apart from e.g. SQL, like the quad model, that it is a Web technology, or that data are commonly very heterogenous.

I have not yet started to explore the scientific literature around SPARQL Federation in any depth as I am still in an early phase of my work. I am currently focusing my efforts on benchmarking. The long-term goal of my work is SPARQL Federation, but that is a minor concern in this paper. Overall, the expected contributions are:

- Benchmarks that are able to cover all realistic performance-influencing parameters for SPARQL engines and make it possible to weigh different parameters, as well as quantify unexplained differences.
- To assist optimization, enable endpoint service descriptions to expose:
 - performance characteristics,
 - statistical digests of data that are optimized with respect to size and query performance.
- SPARQL engine developers can use the benchmark to reliably quantify unexpected adverse effects from a given modification.
- New SPARQL engine users can identify key differences between different engines, and therefore be able to more wisely choose engine to use.

2 State of the Art and Open Problems

2.1 In SPARQL Federation

I take the state of the art in technology to be represented by the current basic SPARQL 1.1 Federated Query Working Draft². In addition, many have implemented federation that doesn't require explicit references to service endpoints, e.g. [8]. A recent scientific treatment of the current specification is in [1]. In that paper, the authors also show an optimization strategy based on execution order of so-called well-designed patterns.

A recent review of the state of the art is in [4]. In addition, [8] proposes *bound joins* and proves they can dramatically reduce the number of requests to federation members that are needed, as well as the implementation of FedX.

It has been my intention to focus on the two problems listed in section 3.3.1 in [4], i.e. strike a balance between accuracy and index size, and updating statistics as data changes. Notably, histogram approaches generally suffer from the problem that they grow too large or become an insufficiently accurate digest, especially in the face of very heterogeneous data. [5] introduced QTrees, which may alleviate the problem of histogram size, but which may not solve it.

Therefore, the core problem is: How do we compute and expose a digest that is of optimal size for the query performance problem?

² <http://www.w3.org/TR/2011/WD-sparql11-federated-query-20111117/>

2.2 In Benchmarking

Numerous benchmarks have been developed for SPARQL, but [2] showed that currently most benchmarks poorly represent the typical data and queries that are used on the Semantic Web. Most recently, [6] addressed some of these problems by using real data and real queries from DBpedia. [7] has developed a benchmark for the federated case.

Current common practice in benchmarking SPARQL-enabled systems is to use or synthesize a certain dataset, then formulate a number of queries seen as representative of SPARQL use in some way. These queries are then executed, and some characteristic of performance is measured, for example the time it takes for the engine to return the full result. Since there is a certain randomness in query times, this process is repeated a number of times and an average response time is found. Different engines can be compared based on these averages.

In many cases, this is sufficient. Sometimes, one engine can execute a query in an order of magnitude faster than another. If this happens systematically for many different queries, there is hardly reasonable doubt as to which is faster. In most cases, the query response times differs little, however. Small differences may seem unimportant but may become important if they are systematic. Even if one engine is dramatically better than another in one case, small deficiencies may add up to make the other a better choice for most applications anyway.

In this case, we must consider the possibility that the random noise can influence the conclusions. Whatever metric is used, it should be treated as a *stochastic variable*. This opens new methodological possibilities, first and foremost using well-established statistical hypothesis testing or ranking rather than just comparing averages.

Furthermore, the current approach presents merely anecdotal evidence that one engine is better than another with respect to performance. It may be that while the benchmarking queries do seem to favor one engine, other queries that have not been tried may reveal that there are strong adverse effects that may not have been anticipated. A more systematic approach is needed to provide a comprehensive and objective basis for comparing SPARQL engines.

In physical science and engineering, conventional wisdom has been that you should only vary one variable at a time to study the effects of that one variable. In medical science, this has been abandoned several decades ago, thanks to advances in statistics. In for example a case where the researcher administrates different treatments to terminally ill patients, some of which may be painful or shorten their lives, experimental economy is extremely important.

Using techniques from statistical experimental design, I propose that it is possible to design an experiment (i.e. a benchmark) which makes it possible to cover all realistic cases and with which we can justify why the remaining corner cases are unlikely to influence the result. For further elaboration, see Section 3.2.

So far, the benchmarking problem has been seen as a software testing problem, but as stated in the introduction this is not the only objective, we may now see if benchmark data can be exposed to help federation query optimizers along with a statistical digest.

The problems addressed by existing benchmarks such as the ones cited above are almost orthogonal to the problems considered by my proposed project. While I have seen some cases that compare performance based on box plots³, it seems not to be common practice. Furthermore, I have not to date seen any work towards using methods like factorial designs to evaluate the performance of software, but there may be a limit in terms of complexity for where it is feasible, and I will restrict myself to SPARQL for this thesis.

3 Proposed Approach and Methodology

3.1 In SPARQL Federation

There are many possible approaches for this part of the thesis. As I expect substantial advances to be made before I start tackling this problem, I have not chosen any methodology, but an interesting direction for work seems to be to find more space-efficient ways to expose statistics in the service description and standardize them.

To this end, I have briefly looked into two approaches: [3] used Bayesian Networks and Probabilistic Relational Models to efficiently represent the joint distribution of database tables, a formalism that could be extended to RDF databases.

Another approach that I have not seen used in the literature is to use parametrized statistics. This would amount to an attempt to fit data to a known distribution function and expose which distribution and its parameters in the service description.

Finally, I have seen little work on the problem of rapidly changing data, so the adaption of existing techniques to such situations may also be interesting.

The evaluation methodology for the SPARQL federation work of the thesis will largely be covered by running the elaborate benchmark designs of the thesis.

3.2 In Benchmarking

Already in 1926, Ronald Fischer noted that complex experiments can be much more efficient than simple ones⁴, starting the experimental design field. One of the simpler designs is “fractional factorial design”, in which several “factors” are studied. In terms of SPARQL execution, the SPARQL engine is clearly a factor, but also, for example, the nestedness of `OPTIONALS` can be a factor, or the number of triples in a basic graph pattern, etc. These numbers are varied to different “levels”. The key to understanding why this can be efficient is that these variations need not occur in the same experiment. Therefore, for the SPARQL language, many combinations of factors can be studied by carefully designing queries to cover different factors, and a formalism called “resolution” has been developed to classify how well this has been achieved, partly answering the question of evaluation methodology for this part of the thesis. We should also validate by comparing this benchmark with conclusions from existing benchmarks.

³ See <http://shootout.alioth.debian.org/> for example

⁴ Cited in http://en.wikipedia.org/wiki/Factorial_experiment

A systematic approach based on statistical experimental design means developing methods to add factors (not only language features will be factors), and then use this methodology to add and then vary all the factors in an optimal fashion. Developing a software suite to perform most of the experiments is a key requirement as the number of experiments that results will be very large.

Factorial design inspires the present work and is covered in elementary textbooks in statistics but is inadequate for this purpose, so I intend to go further into the statistical literature to see if there is a methodology that is better suited to the problem. I also intend to use existing results on complexity bounds for SPARQL query answering to find suitable factors to see if my admittedly bold proposition that it is possible to design a benchmark to cover all realistic cases can be shown to hold.

With this analysis, I speculate based on superficial experience with factorial designs and analysis of variance that certain estimated coefficients can be exposed in the service description to give federated query engines assistance in optimizing for performance characteristics of certain SPARQL implementations.

This approach will yield the contributions listed above if completely successful, but will also advance the state of the art if only partially successful.

References

1. 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)
2. Duan, S., Kementsietsidis, A., Srinivas, K., Udrea, O.: Apples and oranges: a comparison of RDF benchmarks and real RDF datasets. In: *Proc. of the 2011 Int. Conf. on Management of Data, SIGMOD 2011*, pp. 145–156. ACM (2011)
3. Getoor, L., Taskar, B., Koller, D.: Selectivity estimation using probabilistic models. In: *Proc. of the 2001 ACM SIGMOD Int. Conf. on Management of Data, SIGMOD 2001*, pp. 461–472. ACM (2001)
4. Görlitz, O., Staab, S.: Federated Data Management and Query Optimization for Linked Open Data. In: Vakali, A., Jain, L.C. (eds.) *New Directions in Web Data Management 1*. SCI, vol. 331, pp. 109–137. Springer, Heidelberg (2011)
5. Harth, A., Hose, K., Karnstedt, M., Polleres, A., Sattler, K.-U., Umbrich, J.: Data summaries for on-demand queries over linked data. In: *Proc. of the 19th Int. Conf. on World Wide Web, WWW 2010*, pp. 411–420. ACM (2010)
6. Morsey, M., Lehmann, J., Auer, S., Ngomo, A.-C.N.: DBpedia SPARQL Benchmark – Performance Assessment with Real Queries on Real 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. 454–469. Springer, Heidelberg (2011)
7. 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)
8. 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)

A Reuse-Based Lightweight Method for Developing Linked Data Ontologies and Vocabularies

María Poveda-Villalón

Ontology Engineering Group, Departamento de Inteligencia Artificial
Facultad de Informática, Universidad Politécnica de Madrid
Campus de Montegancedo s/n
28660 Boadilla del Monte, Madrid, Spain
mpoveda@fi.upm.es

Abstract. The uptake of Linked Data (LD) has promoted the proliferation of datasets and their associated ontologies for describing different domains. Particular LD development characteristics such as agility and web-based architecture necessitate the revision, adaption, and lightening of existing methodologies for ontology development. This thesis proposes a lightweight method for ontology development in an LD context which will be based in data-driven agile developments, existing resources to be reused, and the evaluation of the obtained products considering both classical ontological engineering principles and LD characteristics.

Keywords: ontology, vocabulary, methodology, linked data.

1 Motivation and Research Questions

The Linked Data (LD) initiative enables the easy exposure, sharing, and connecting of data on the Web. Datasets in different domains are being increasingly published according to LD principles¹. In order to realize the notion of LD, not only must the data be available in a standard format, but concepts and relationships among datasets must be defined by means of ontologies or vocabularies².

New vocabularies to model data to be exposed as Linked Data should be created and published when the existing and broadly used ontologies do not cover all the data intended for publication. Based on the guidelines for developing and publishing LD [5], LD practitioners should describe their data (a) reusing as many terms as possible from those existing in the vocabularies already published and (b) creating new terms, when available vocabularies do not model all the data that must be represented. During this apparently simple process several questions may arise for a data publisher. This PhD thesis proposal aims to develop a lightweight method to guide LD

¹ <http://www.w3.org/DesignIssues/LinkedData.html>

² At this moment, *there is no clear division between what is referred to as “vocabularies” and “ontologies”* (<http://www.w3.org/standards/semanticweb/ontology>). For this reason, we will use both terms indistinctly in this paper.

practitioners through the process of creating a vocabulary to represent their data. The ambition is to maintain the advantages, whilst offering solutions to cover the insufficiencies. The proposed method will be mainly based in reusing widely deployed vocabularies, describing data by means of answering the following questions:

- Where and how can vocabularies be found?
- Which vocabularies or elements should be reused?
- How much information should be reused?
- How to reuse elements or vocabularies?
- How to link elements or vocabularies?
- How should terms be created according to LD and ontological principles?

2 State of the Art

The 1990s and early years of the new millennium have witnessed a growing interest in methodologies that support the creation of ontologies. All these approaches have facilitated major progress, transforming the art of building ontologies into an engineering activity. However, the existing methodologies should be reviewed and adapted to support ontology development and evolution in the Linked Data context. For example, the well-known traditional methodologies as METHONTOLOGY [3, 2], On-To-Knowledge [9] and DILIGENT [7], as well as the NeOn Methodology [10], propose time and resource consuming activities instead of simple and (semi)automatic processes as is desirable in an LD application development. Other approaches propose agile methodologies for ontology development, however, these are often unsuitable when working with Linked Data. The eXtreme Method [6] assumes that the requirements are set at the beginning and are unchanging, which is unrealistic while working with ontology evolution using LD. Other works as the XD Methodology [8] consider Ontology Design Pattern as the only kind of resource to be reused during the development, or do not consider ontological resource reuse as is the case of RapidOWL [1] instead of basing the development in reusing the terms already available in the LD cloud. In addition, none of the above mentioned methodologies consider particular requirements of ontologies and vocabularies that are going to be used in a LD environment. Within the literature on Linked Data [5], some high-level guidelines have been outlined to create vocabularies; however no concrete processes and detailed guidelines have been proposed to carry out such a development. Therefore, to the best of the author's knowledge there are no methodological approaches that help ontology developers to build ontologies or vocabularies to be used in an LD context taking into account its particular characteristics, following a lightweight approach and providing detailed methodological guidelines for the proposed activities.

3 Proposed Approach

This PhD thesis proposal investigates how traditional and heavy methodologies for the development of ontologies and ontology networks could be lightened and adapted to an LD context by considering its particular requirements. A lightweight method for

ontology development with a data-driven approach will be created including techniques and tools to carry out each of the proposed activities. Ontology evaluation techniques according to LD principles and architecture will also be developed in a pattern-based way in order to make their application highly automated and reusable.

As Fig. 1 illustrates, the proposed method consists of a workflow of activities based on the data intended for publication. With the aim of following a data-driven ontology/vocabulary development rather than the competency question development [4], used in the majority of existing methodologies, this workflow starts with a term extraction activity. The next step is to search for available vocabularies and ontologies already used in LD following the approach proposed in [5]. Subsequently, through the next steps, the available resources will be selected and integrated in order to produce a first model describing the data, for which, methodological guidelines will be provided. Finally, this model will be completed, in case it does not cover all the data to be represented, and evaluated before publishing and making the data available.

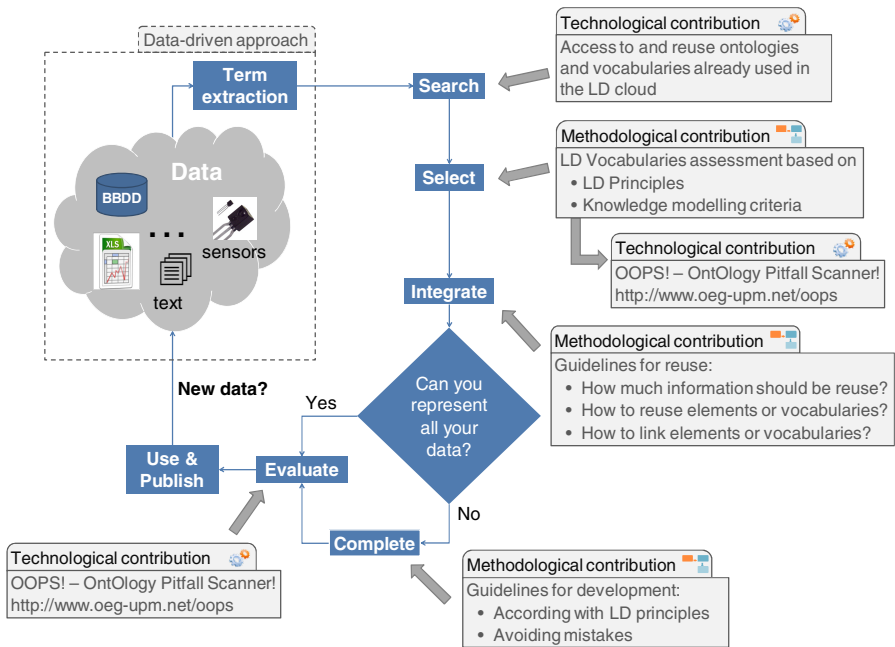


Fig. 1. Lightweight method for building Linked Data ontologies and vocabularies

As well as the methodological contributions, Fig. 1 also shows the technological products that will be provided in this PhD thesis including technological support to access ontologies and vocabularies already used in LD cloud and OOPS!, an ontology validation tool. Table 1 presents the contributions that will be provided by this thesis in order to throw light upon the open questions presented in Section 1.

4 Planned Research Methodology

The research methodology to be followed will consist of prototype development from which a high level abstraction will be extracted. Results are evaluated at each iteration and used to inform the approach, which is fine-tuned until the results are satisfactory.

Initially, the state of the art in ontology development will be analyzed with a particular emphasis on problems involved with working in an LD environment.

Once the problem is defined, a first prototype of the agile method for ontology development presented in Section 3 will be proposed together with a first version of the technological application supporting it. This method will include prescriptive methodological guidelines for the proposed steps, namely, Search, Select, Integrate, Complete and Evaluate. These guidelines will be provided in a pattern-based manner whenever possible with the aim of enhancing their applicability and reusability.

Following this, an experimentation phase based on controlled experiments will be carried out over the obtained results taking as use cases the Ontology Engineering Group³ LD developments. The experimental results will be used to improve the method and associated technological support. At least another iteration to evaluate the results and improvements will be carried out before proposing the final solution.

To conclude, the method will be analyzed a) from a user point of view by questionnaires to measure applicability; b) by controlled experiments involving the evaluation of ontologies developed for Linked Data applications carried out within the author's group (e.g: GeoLinkedData, UPMLinkedData, EcoGeoLinkedData, etc.) as initial evaluation environment and developments carried out by external organizations during a later evaluation step; and c) by comparison with other methods. The technological support will be compared with other tools with a similar purpose, gathering measures of: time spent by user in evaluating an ontology, usability tests, and user satisfaction after using the tool.

Table 1. Proposed steps, addressed open questions and PhD contributions

Step(s)	Addressed Open Question(s)	PhD Contributions
Search	1. Where and how can vocabularies be found?	<ul style="list-style-type: none"> • Comparative study of the indexes or registries for ontologies used in LD cloud. • Techniques to access vocabularies.
Select	2. Which vocabularies or elements should be reused?	<ul style="list-style-type: none"> • Guidelines for assess and select vocabularies or elements to be reused. • Tool for evaluating vocabularies with respect to a set of modeling criteria
Integrate	3. How much information should be reused? 4. How to reuse elements or vocabularies? 5. How to link elements or vocabularies?	<ul style="list-style-type: none"> • Guidelines for ontology pruning and merging. • Guidelines for providing links between vocabularies and elements.
Complete Evaluate	6. How should terms be created according to LD and ontological principles?	<ul style="list-style-type: none"> • Guidelines for developing and enriching ontological terms according to LD criteria and ontological foundations. • Guidelines and technological support for ontology evaluation according to modeling criteria.

³ <http://www.oeg-upm.net/>

5 Conclusion

Describing data by means of vocabularies or ontologies is crucial for the Semantic Web and LD realization. LD development characteristics such as agility and web-based architecture force the revision and lightening of existing methodologies for ontology development. This paper briefly presents the motivation and the proposed approach of the thesis, the main goal of which is to propose a lightweight method for ontology development in an LD context following a data-driven approach. Such a method will be developed together with technological support to ease its application and will be based in agile developments and the evaluation of the obtained products considering both classical ontological engineering principles and LD characteristics.

The next steps consist of analyzing particular characteristics of LD developments and proposing a first prototype both for the method and its technological support. Following this, the obtained results will be evaluated in order to improve them in an iterative way.

Acknowledgments. This work has been supported by the Spanish project BabelData (TIN2010-17550). I would also like to thank Asunción Gómez-Pérez and Mari Carmen Suárez-Figueroa for their advice and dedication and Edward Beamer for his comments and revisions.

References

1. Auer, S.: RapidOWL - an Agile Knowledge Engineering Methodology. In: STICA 2006, Manchester, UK (2006)
2. Fernández-López, M., Gómez-Pérez, A., Juristo, N.: METHONTOLOGY: From Ontological Art Towards Ontological Engineering. In: Spring Symposium on Ontological Engineering of AAAI, pp. 33–40. Stanford University, California (1997)
3. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering. Advanced Information and Knowledge Processing. Springer (November 2003) ISBN 1-85233-551-3
4. Gruninger, M., Fox, M.S.: The role of competency questions in enterprise engineering. In: Proceedings of the IFIP WG5.7 Workshop on Benchmarking - Theory and Practice, Trondheim, Norway (1994)
5. Heath, T., Bizer, C.: Linked data: Evolving the Web into a global data space, 1st edn. Morgan & Claypool (2011)
6. Hristozova, M., Sterling, L.: An eXtreme Method for Developing Lightweight Ontologies. CEUR Workshop Series (2002)
7. Pinto, H.S., Tempich, C., Staab, S.: DILIGENT: Towards a fine-grained methodology for DIStributed, Loosely-controlled and evolvInG Engineering of oNTologies. In: de Mantaras, R.L., Saitta, L. (eds.) Proceedings of the ECAI 2004, August 22-27, pp. 393–397. IOS Press, Valencia (2004) ISBN: 1-58603-452-9, ISSN: 0922-6389
8. Presutti, V., Daga, E., Gangemi, A., Blomqvist, E.: eXtreme Design with Content Ontology Design Patterns. In: WOP 2009 (2009)
9. Staab, S., Schnurr, H.P., Studer, R., Sure, Y.: Knowledge Processes and Ontologies. IEEE Intelligent Systems 16(1), 26–34 (2001)
10. Suárez-Figueroa, M.C.: Doctoral Thesis: NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse. Universidad Politécnica de Madrid, Spain (2010)

Optimising XML–RDF Data Integration

A Formal Approach to Improve XSPARQL Efficiency

Stefan Bischof

Siemens AG Österreich, Siemensstraße 90, 1210 Vienna, Austria
bischof.stefan@siemens.com

Abstract. The Semantic Web provides a wealth of open data in RDF format. XML remains a widespread format for data exchange. When combining data of these two formats several problems arise due to representational incompatibilities. The query language XSPARQL, which is built by combining XQuery and SPARQL, addresses some of these problems. However the evaluation of complex XSPARQL queries by a naive implementation shows slow response times. Establishing an integrated formal model for a core fragment of XSPARQL will allow us to improve performance of query answering by defining query equivalences.

Keywords: Data Integration, Query Optimisation, XQuery, SPARQL.

1 Integrating XML and RDF Data Lacks Efficiency

Data exchange became faster and more convenient by using the Internet. Two of the predominant formats to exchange data over the Internet but also inside organisations are *XML*, used for business data, financial data, etc., and *RDF*, the data format of the Semantic Web. While XML is tree based with relevant child order, RDF data is unordered due to its graph structure. The W3C recommended query languages for XML and RDF are XQuery and SPARQL, respectively. A brief example scenario: Fred has to send a report containing the number of bought items for each customer to Julie. Julie expects a XML document while Fred stores his data in a RDF triple store. Fred also has to integrate XML documents from Julie into his triple store. When using standard query languages Fred ends up with several queries in different languages glued together by scripts.

XSPARQL [3] is a combined query language making transformation and integration of XML and RDF data easier. XSPARQL extends XQuery by two additional grammar expressions: the *SparqlForClause* to use SPARQL's graph pattern matching facility including operators, and the *ConstructClause* to allow straightforward creation of RDF graphs. Listing 1 shows an example query. Fred produces the XML document for Julie by taking the RDF graph as input. Graph patterns are used to query the RDF graph and the variables of the graph pattern are then used by any XQuery expression. Since XSPARQL is based on XQuery, a Turing-complete language, it supports all kinds of more sophisticated data manipulations. Fred can also use XSPARQL to convert Julie's XML documents to custom RDF by using a single query containing a `construct` template.

Listing 1. XSPARQL query: List customers with number of bought items

```

for * where { [] foaf:name $name ; :id $id . }
return <customer name="{ $name }"> { count(
  for * where { [] :buyer [:id $id]; :itemRef []. }
  return <item/> ) } </customer >

```

Efficiency is important when transforming data between XML and RDF. Evaluating complex XSPARQL queries, i.e., queries containing nested graph patterns, shows slow query response times with the current prototype.¹ The performance impact can be explained by the architecture which rewrites XSPARQL queries to XQuery queries containing external calls to a SPARQL engine. The main advantages of such an implementation are reuse of state of the art query optimisation as well as access to standard XML databases and triple stores. But for complex XSPARQL queries a high number of SPARQL queries (similar to each other) are yielded, resulting in a major performance impact [3, 4, 5]. Listing 1 contains a nested *SparqlForClause* in line 3, which depends on the specific customer ID ($\$id$). The implementation issues a separate SPARQL query for each customer. Simple join reordering [3, 4] improves query answering performance. But there is still a performance gap between simple flat queries and complex nested queries, thus optimisations on a more fundamental level are needed.

The XQuery semantics specification [8] formalism, i.e., natural semantics, is not well suited for concisely expressing query equivalences. As opposed to Relational Algebra, which serves as the basis for query languages like SPARQL, natural semantics uses calculus-like rules to specify type inference and evaluation semantics. Since the XSPARQL semantics reuses the formalism of XQuery, a concise description of possible optimisations is inhibited by the formalism.

To find and express new optimisations and prove their correctness, we need a more suitable formalism. Finding such a formalism is the first goal of the presented PhD topic. Like other formalisms used for query optimisation we also use only a core fragment of the query language to optimise. We thus propose an integrated formal model of an XSPARQL core fragment called XS.

Section 2 gives an overview of related work and describes open problems. Section 3 explains the approach we propose which involves creating a core formalism to express different kinds of optimisations by query equivalence and rules for cost-based optimisations. Lastly Sect. 4 outlines the research methodology for addressing the efficiency problem by a formal approach.

2 Related Work

Some published approaches to combine XML and RDF data use either XML or RDF query languages or a combination. But none of these approaches is advanced enough to address “cross-format” optimisation of such transformations. In general such optimisations could be implemented by translating queries completely to a

¹ An online demo and the source code are available at <http://xsparql.deri.org/>

single existing query language, such as XQuery, and shift optimisation and query evaluation to an XQuery engine. Another approach to implement a combined query language is to build an integrated evaluation engine from scratch.

Translate SPARQL to XQuery. Groppe et al. [12] present a language extension syntactically similar to XSPARQL, which extends XQuery/XSLT to allow SPARQL queries as nested expressions. After an initial RDF to XML data translation the query translation uses an intermediary algebra allowing SPARQL optimisation, resulting in a pure XQuery/XSLT query. Fischer et al. [9] implement a similar approach of rewriting SPARQL queries to XQuery. By using a more sophisticated initial XML data transformation and heavier triple pattern join reordering, they achieve better performance for queries with simple joins and filters. Other approaches relying on ontology information are not relevant for the current work, as a query language like XSPARQL gives the user fine grained control over the output and intermediary transformations, but does not work automatically. None of the above approaches address “cross-format” optimisation. As we have found in our initial practical evaluation, query engines not catering for the combination of both languages will suffer a severe performance impact when evaluating complex queries. Complex queries containing nested graph patterns will occur frequently in practice when transforming data from RDF to XML because of the inherent tree shape of the target XML documents.

Optimisation Formalisation. The current formal specification of XSPARQL, based on XQuery, is too verbose to allow comprehensible specification of optimisations. Therefore we are looking for a formalisation supporting our specific requirements: concise semantics and optimisation specification of a language fragment.

XQuery optimisation is an obvious starting point when investigating optimisation of XSPARQL. Some of the recent works on XQuery optimisation use custom algebras [1, 13, 14]. Other approaches [2, 6, 7, 10, 11] map XQuery to standard logics (Datalog/logic programming, monadic second order logic) in order to profit from well studied properties. A native algebra easily transfers to an implementation prototype, however it makes comparisons to other formalisms hardly feasible. Using a well known formalisation like Datalog would make custom XSPARQL optimisations easier since existing optimisation approaches could be reused.

3 A Formal Model for Cross-Format Optimisation

The first step in addressing the XSPARQL efficiency problem, is to express the query language semantics in a formal way. Since the XQuery semantics formalism is operational and rather verbose, we look into alternative formalisms capable of expressing the language semantics, at least for a core fragment, as well as expression equivalences and corresponding proofs.

We aim to formalise an XSPARQL core fragment called *XS*, which is expressive enough to cover relevant queries or query parts, and simple enough to be able to use query equivalences for optimisation heuristics. *XS* will therefore be a well behaving formal model providing a unified representation and manipulation framework for (unordered) graph data and (ordered) tree data at the same time. The optimisation heuristics can also be used for related approaches since the model works on the data models of XML and RDF. A cost model will be created, enabling cost based optimisations as well. We will gather data about the underlying data distribution for the cost model. We will build an *XS* prototype to allow comparing the query answering performance to the naive implementation.

Optimisation of query languages operating on the data models of XML and RDF at the same time are not well studied so far. Since SPARQL and XQuery are based on different paradigms—SPARQL is a declarative language comparable to SQL, while XQuery is a functional query language—formalisations differ greatly, even when considering only fragments, as usually studied for optimisation.

One Limitation of the approach is expressivity: although XSPARQL is very expressive, we aim at a concise formalisation. Thus *XS* will disallow many queries. Even though we are aiming at finding a good compromise between optimisation and expressivity, increasing expressivity is out of scope of this work.

4 Research Methodology

Finding novel optimisations for querying across formats requires several tasks: literature search, formalisation, theoretical verification, prototyping, practical evaluation and implementation of a relevant use case. All of these tasks are not executed strictly in sequence, but rather in several iterative refinement steps.

Literature Search. For clearly defining the research problem and ensuring novelty of the approach, we are currently performing an extensive literature search. Data integration and data exchange are recent topics attracting researchers from different domains. Integrating data from different representations however, has not seen broad attention. A second topic currently under investigation is finding an appropriate formalisation for *XS*.

Core Fragment Isolation and Optimisation. We will isolate an XSPARQL core fragment which allows expressing practically relevant queries and holds optimisation potential. Optimisations are defined by query equivalences.

Theoretical Evaluation. The core fragment allows theoretical correctness verification. Using *XS* we will prove theoretical properties such as complexity bounds and query/expression equivalences.

Prototype. A prototype implementation is needed to devise practical evaluations and to implement a demonstration use case. The prototype will be built using state of the art technology but should still be kept simple enough to allow quick adaption. We also plan to publish concrete use case solutions to show feasibility and relevance of our approach in practise.

Practical Evaluation. In previous work we proposed the benchmark suite XMarkRDF [4] to measure the performance of RDF to XML data transformation. XMarkRDF is derived from the XQuery benchmark suite XMark.

We will extend XMarkRDF by queries covering specific types of joins as addressed by the XS optimisations. With this benchmark suite we plan to compare our optimisations to the current implementation and to comparable implementations such as the translator presented by Groppe et al. [12].

In summary the research topic includes isolating an XSPARQL core fragment, finding or creating a suitable formalisation (a unified graph-tree data processing framework), describing optimisations by query equivalences, proving soundness, evaluating performance practically and showing applicability in a prototype. With the unified formal model for XML–RDF querying, we aim to provide a tool to formally study heterogeneous data integration and improve query performance.

Acknowledgements. The work of Stefan Bischof will be partly funded by a PhD thesis grant from Siemens AG Österreich. The goal of this grant is that in the course of his thesis, he will investigate the potential to deploy the developed technologies within real-life data integration use cases within Siemens. Preliminary results have been partly funded by Science Foundation Ireland grant no. SFI/08/CE/I1380 (Lion-2) and an IRCSET grant.

References

1. Beeri, C., Tzaban, Y.: SAL: An Algebra for Semistructured Data and XML. In: WebDB (Informal Proceedings) 1999, pp. 37–42 (June 1999)
2. Benedikt, M., Koch, C.: From XQuery to Relational Logics. *ACM Trans. Database Syst.* 34(4), 25:1–25:48 (2009)
3. Bischof, S., Decker, S., Krennwallner, T., Lopes, N., Polleres, A.: Mapping between RDF and XML with XSPARQL. Tech. rep., DERI (March 2011), <http://www.deri.ie/fileadmin/documents/DERI-TR-2011-04-04.pdf>
4. Bischof, S., Decker, S., Krennwallner, T., Lopes, N., Polleres, A.: Mapping between RDF and XML with XSPARQL (2011) (under submission)
5. Bischof, S., Lopes, N., Polleres, A.: Improve Efficiency of Mapping Data between XML and RDF with XSPARQL. In: Rudolph, S., Gutierrez, C. (eds.) RR 2011. LNCS, vol. 6902, pp. 232–237. Springer, Heidelberg (2011)
6. ten Cate, B., Lutz, C.: The Complexity of Query Containment in Expressive Fragments of XPath 2.0. *J. ACM* 56(6), 31:1–31:48 (2009)
7. ten Cate, B., Marx, M.: Axiomatizing the Logical Core of XPath 2.0. *Theor. Comp. Sys.* 44(4), 561–589 (2009)
8. Draper, D., Fankhauser, P., Fernández, M., Malhotra, A., Rose, K., Rys, M., Siméon, J., Wadler, P.: XQuery 1.0 and XPath 2.0 Formal Semantics, 2nd edn. W3C Recommendation, <http://www.w3.org/TR/2010/REC-xquery-20101214/>
9. Fischer, P.M., Florescu, D., Kaufmann, M., Kossmann, D.: Translating SPARQL and SQL to XQuery. In: XML Prague 2011, pp. 81–98 (March 2011)
10. Gottlob, G., Koch, C., Pichler, R.: Efficient Algorithms for Processing XPath Queries. *ACM Trans. Database Syst.* 30, 444–491 (2005)

11. Gottlob, G., Koch, C., Pichler, R., Segoufin, L.: The Complexity of XPath Query Evaluation and XML Typing. *J. ACM* 52(2), 284–335 (2005)
12. Groppe, S., Groppe, J., Linnemann, V., Kukulenz, D., Hoeller, N., Reinke, C.: Embedding SPARQL into XQuery/XSLT. In: SAC 2008, pp. 2271–2278. ACM (2008)
13. Jagadish, H.V., Lakshmanan, L.V.S., Srivastava, D., Thompson, K.: TAX: A Tree Algebra for XML. In: Ghelli, G., Grahne, G. (eds.) DBPL 2001. LNCS, vol. 2397, pp. 149–164. Springer, Heidelberg (2002)
14. Zhang, X., Pielech, B., Rundesnteiner, E.A.: Honey, I Shrunk the XQuery!: an XML Algebra Optimization Approach. In: WIDM 2002, pp. 15–22. ACM (2002)

Software Architectures for Scalable Ontology Networks

Alessandro Adamou^{1,2}

¹ Alma Mater Studiorum Università di Bologna, Italy

² ISTC, National Research Council, Italy

Abstract. Theory and practice in ontology management are drifting away from monolithic ontologies towards ontology networks. Processing interconnected knowledge models, e.g. through reasoning, can provide greater amounts of explicit knowledge, but at a high computational cost if the whole knowledge base has to be handled by concurrent processes. Our research tackles this problem via a software engineering approach. We devised a framework that combines privileged containers for ontology models with volatile containers for instance data that vary frequently. A reference implementation was developed in an Apache project for content management, and its adoption is providing data and use cases for validating it with objects from Fedora Commons repositories.

1 Introduction

The discipline and methodologies of ontology engineering are gradually shifting away from the monolithic notion of ontologies. Current practices and empirical sciences see reuse as a key criterion for constructing knowledge models today, so that networking-related concepts are being applied to interconnectivity between ontologies. As a consequence, the notion of *ontology network* has begun to surface. An ontology network is created either at design time (e.g. the engineer adds OWL import statements and reuses imported entities) or at a later stage (e.g. someone discovers alignments between multiple ontologies agnostic to each other, creates an ontology with alignment statements and sets up a top ontology that imports all of them). Our research concentrates on exploiting the latter scenario while still accommodating the former.

Establishing ontology networks can have a number of advantages, the most apparent ones being related to redundancy minimisation and reasoning. Interconnecting ontology modules keeps from re-defining basic or shared entities, can augment knowledge on the given entities and produce even more inferred knowledge when reasoners are run on the network. This is, however, when reasoning performance issues kick in. Description Logic (DL) classifies a whole knowledge structure non-selectively, and whether the process is incremental depends on the reasoner implementation. However, if a reasoner is being run for a specific task, portions of this knowledge structure could be unnecessary and inflate the computation to produce results that are no use to the task at hand. Let us consider the social network domain for an example. If the task is to infer a *trust*

network of users based on their activity, there is hardly any point in including the GeoSpecies ontology that classifies life form species¹, which could however be part of the knowledge base. If, however, another task is to infer an *affinity network*, GeoSpecies could be considered for assessing affinity between zoologists.

There are also considerations to be made on the network substructure. The OWL language relies upon import statements to enforce dependency resolution to a certain extent, and as a de facto standard mechanism it has to be respected. However, import statements are static artifacts that imply strict dependencies identified at design time, therefore the dynamic usage of such OWL constructs has to be delegated to a software platform that serves or aggregates ontologies. We argue it should be combined with an OWL2-compliant versioning scheme.

2 Problem Statement and Research Questions

Given the above challenges, we have formulated these **research questions**:

RQ1. How can a single software framework create ontology networks on top of a common knowledge base, in order to serve them for different processing tasks?

We call these ontology networks *virtual*, in that it is the framework that can create, rewrite and break ontology linkage at runtime, as required by a specific instance-reasoning task, without affecting the logical axioms in the ontologies.

RQ2. Is it possible for such a framework to safely accommodate concurrency in multi-user contexts, and with minimum resource overhead?

Instance data may vary across simultaneous users of a system, but also with time (e.g. only a daily snapshot of data feeds, or the whole history of an ABox for a domain). This variability implies multiple simultaneous virtual networks. A TBox, on the other hand, can be comprised of consolidated schemas and vocabularies, which would be preposterous to copy across ontology networks. However, they are generally the ones with a greater impact on reasoner performance, and users should be able to exclude unneeded parts of them from their networks. We must also make sure that (i) changes in a user-restricted ABox do not affect another user's ABox, and (ii) the memory footprint of ontology networks in the framework is negligible compared to the combined size of the knowledge base.

RQ3. Can the constructs and limits of standard ontology languages be followed and exploited to this end, without altering the logical structure of an ontology?

Our goal with RQ3 is to make sure that, whatever objects the framework introduces, they can always be exported to legal OWL 2 artifacts; that they do not require yet another annotation schema; that they do not force the addition of logical axioms or the interpretation of existing ones in the original ontologies.

¹ GeoSpecies knowledge base, <http://lod.geospecies.org>

3 Related Work

Networked architectures. Ontology architecture focuses on applying development and deployment schemes *within* the ontology lifecycle [6]. In this respect, some later efforts in *ontology repository* design such as *Cupboard* [3] are showing some concern over their potential use to implement linked ontology networks, as in the networked model of [5]. The study on ontology architectures has led to the problem of determining the modular decomposition of monolithic ontologies, whose theoretical foundations are extensively described in [7].

Methodologies with reuse support. We are concerned with the importance of reuse in ontology engineering, since reused concepts can be potential interconnection nodes, with little to no alignment effort. The *NeOn Methodology* [8] and *eXtreme Design* [2] are examples of such reuse-oriented design methodologies.

Scalable reasoning. Scalability is addressed with regard to reasoning on modularised ontologies [1], but also to adaptively reacting to changes in an ontology. Forward-chaining rule-based reasoners such as LMF² have been proposed as a trade-off between expressiveness and computational complexity in large datasets. As for DL reasoning, the approach of Fokoue et al. [4] processes the ontologies at reasoning time through pruning, summarizing and in-memory imaging.

4 Approach and Implementation

The requirement analysis, design, development and evaluation of our proposed solution all occur in the field of content management, with the aim to deliver semantically enhanced capabilities to Content Management Systems (CMS). This setting has allowed us to expand our investigation vertically across knowledge, persistence, interaction and user management, all the while maintaining an angle on industrial adoption trends and software engineering.

Our research and design work opted for a *separation of concerns* between the structures holding class and instance data, following a finer granularity than the classic TBox/ABox pair. We devised an architecture where variable instance data (across time or users) can be orthogonal to vocabularies, meta-models or consolidated instance data. We then distinguished scenarios where some architectural layers should be considered ‘privileged’ for update by applications and CMS administrators, while others (mainly ABox-related) are devoted to volatile instance data supplied by users or external services for single or batch operations. Our framework assembles ontology networks using the following constructs:

- **Session.** A container for instance data loaded at runtime. A user or client application can open a session, load the ontologies containing instance data and attach other parts of the network (see below). For instance, to classify the knowledge extracted from a daily blog post feed, a client can push

² Linked Media Framework reasoner,
<http://code.google.com/p/kiwi/wiki/Reasoning>

the corresponding RDF graph into the same session day-by-day. A session takes ownership of any non-versioned ontologies loaded into it. Although not intended for persistence, it can last across more HTTP sessions over time.

- **Scope.** A “realm” for all the ontologies that model a given domain or context. For instance a “social networks” scope can reference FOAF, SIOC, alignments between these and other related custom models. One or more scopes can be attached to one or more sessions at once, thereby realizing the virtual network model. Each scope is divided in two spaces. The *core* space contains the ontologies that provide immutable knowledge, such as foundational ontologies. The *custom* space extends the core space with additional knowledge such as alignments with controlled vocabularies. Note that one occurrence of the same ontology can be shared across multiple spaces.

When these artifacts are exported as OWL ontologies, linkage relations are materialised as `owl:imports` statements forged for each ontology network, while `owl:versionIRIs` are used by these artifacts to either “claim ownership” of the ontologies they manage or share them (e.g. TBoxes) across networks.

This ontology network management architecture has been implemented as part of the **Apache Stanbol** service platform for semantic content management³. Since Stanbol is an extensible framework, any plugin can setup and manage its own ontology scopes and sessions. We contributed the following components to the Stanbol ontology manager package:

- **OntoNet** implements an object model of sessions, scopes and spaces, and comes with Java and RESTful APIs for configuring ontology networks. OntoNet implements policies for determining which ontologies to store persistently and which should be either shared or replicated across networks. The OntoNet constructs are supported by the reasoner features of Stanbol, which implement background jobs and concurrent reasoning services with configurable expressivity.
- **Ontology registry manager** is the facility for referencing ontologies external to Stanbol. By setting up a registry, which is itself an RDF graph, Stanbol can aggregate its ontologies from all over the Web and make them available on-demand or OntoNet to assemble them into ontology networks.

At the time of writing, Stanbol is undergoing its release candidate phase. Our contribution also comes as part of the Interactive Knowledge Stack project⁴.

5 Validation

With the reference implementation of our framework in place, we are moving on to the phase of validating it on use cases from the content management domain. Small and medium enterprises have committed to adopt Apache Stanbol. The main use case is provided by a company working in content curation for

³ Apache Stanbol, <http://incubator.apache.org/stanbol>

⁴ Interactive Knowledge Stack (IKS), <http://code.google.com/p/iks-project/>

the visual arts domain. There, a Fedora Commons digital object repository⁵ is used to maintain image metadata, which are interconnected with a SKOS-based representation of the Getty ULAN repository⁶. OntoNet and reasoners will be employed to produce knowledge enrichments over the standard Fedora metadata vocabulary and present different user interface views on them based on different ontology network configurations. In another use case, OntoNet will be used to manage simultaneous content hierarchies from a shared repository. Scopes are selected depending on the knowledge domains that each user decides to activate, while sessions contain the metadata of user-owned and shared content items.

At the same time, we are developing benchmarking tools to evaluate the computational efficiency of the system *tout-court*. Benchmarking will consider (i) the amount and complexity of different networks that can be setup at the same time on the same knowledge base, their memory usage and the minimum Java VM size required; (ii) the overhead given by loading the same ontology into multiple scopes or sessions; (iii) the duration of DL classification runs on an ontology network large enough to deliver the expected inferences, compared against standard DL reasoners called over the non-pruned knowledge base. The possible size of the knowledge base per se will not be measured as it is strictly bound to the storage mechanism employed. To date, the system effectiveness has been measured through unit-testing and stress tests are guaranteeing that we can set up a scope on a ~ 200 MiB ontology using a VM ~ 1.2 times as large.

References

1. Bao, J.: Representing and reasoning with modular ontologies. Ph.D. thesis, Iowa State University (2007)
2. Blomqvist, E., Presutti, V., Daga, E., Gangemi, A.: Experimenting with eXtreme Design. In: Cimiano, P., Pinto, H.S. (eds.) EKAW 2010. LNCS, vol. 6317, pp. 120–134. Springer, Heidelberg (2010)
3. d’Aquin, M., Euzenat, J., Le Duc, C., Lewen, H.: Sharing and reusing aligned ontologies with Cupboard. In: Gil, Y., Noy, N.F. (eds.) K-CAP. ACM (2009)
4. Fokoue, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K.: The Summary Abox: Cutting Ontologies Down to Size. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 343–356. Springer, Heidelberg (2006)
5. Haase, P., Rudolph, S., Wang, Y., Brockmans, S., Palma, R., Euzenat, J., d’Aquin, M.: D1.1.1 networked ontology model. NeOn Deliverable 1(D1.1.1), 1–60 (2006)
6. Obrst, L.: Ontological Architectures, pp. 27–66. Springer, Netherlands (2010)
7. Stuckenschmidt, H., Parent, C., Spaccapietra, S. (eds.): Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization. LNCS, vol. 5445. Springer, Heidelberg (2009)
8. Suárez de Figueroa Baonza, M.C.: NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse. Ph.D. thesis, Universidad Politécnica de Madrid, Madrid, Spain (2010)

⁵ Fedora Commons, <http://fedora-commons.org/>

⁶ Union List of Artist Names, <http://www.getty.edu/research/tools/vocabularies/ulan/>

Identifying Complex Semantic Matches

Brian Walshe

KDEG & FAME, Department of Computer Science and Statistics,
Trinity College Dublin
walshebr@scss.tcd.ie

Abstract. One-to-one correspondences are not always sufficient to accurately align ontologies, and instead complex correspondences with conditions and transformations may be required. Correspondence patterns provide models which can be used to guide the process of developing complex correspondences. However, it is necessary to first identify which pattern to apply to a given alignment problem. This PhD proposes the development of algorithms, methods and processes for refining elementary correspondences between concepts or relations into complex ones by identifying which correspondence pattern best represents a given correspondence. To date an evaluation of a system to refine correspondences between classes in the YAGO and DBpedia ontologies has been completed. This evaluation showed that for a subsumption correspondence, a training set of 30 instances of the class being mapped was sufficient to refine the match to a conditional one 89% of the time. Hence we have shown that this is a promising approach for correspondences with a conditional element, and correspondences with a translation element will be examined next.

1 Motivation

When organizations wish to work together, integrating their data is usually a significant challenge. As each organization's data resources have often been developed independently, they are subject to heterogeneity – primarily semantic, syntactic and structural. The use of ontologies has long been identified as an effective way of facilitating the integration process, seeing use in diverse fields, including for example, bio-medicine and high-energy physics [1].

Suitable techniques for discovering and describing matches between ontologies are still very much an open problem [2]. Matching ontologies is a demanding and error prone task. Not only does it require expert knowledge of the matching process, but also an in-depth understanding of the subject the ontologies describe, and typically knowledge of the principals used to construct the ontologies. The ontologies can differ in scope, granularity and coverage, they can use different paradigms to represent similar concepts, or use different modeling conventions [3]. Therefore producing high quality correspondences is typically semi-automated – an expert user approves and refines match candidates produced by an automatic semantic matcher tool [4].

Semantic matcher tools typically focus on detecting schema-level equivalence relations. However elementary correspondences between named ontology elements are often not sufficient for tasks such as query rewriting, instance translation, or instance

mediation. Complex correspondences which contain conditions and transformations are necessary to satisfy these real-world use cases. Recent semantic matching research has produced a taxonomy of complex correspondences, called Correspondence Patterns by Scharffe [5]. As of version 4.0, the INRIA Alignment API [6] has introduced the Expressive Declarative Ontology Alignment Language (EDOAL) for expressing complex relationships of this nature, but support among semantic matchers for discovering matches that make full use of EDOAL is still limited.

While progress has been made in matching research, the potential use cases for semantic matching have also expanded with the realization of a web of data as Linked Data. Linked Open Data (LOD) represents a large amount of the content contained in the current Semantic Web, with DBpedia [7] currently describing 3.64 million things, 1.83 million of which are classified in a consistent ontology¹. As these sources overlap and complement each other, mapping between them is important. However the shallow taxonomies typically used in LOD sources means that complex alignments are required if we wish to use this instance data with the richer structure found in a richer ontology. For example an instance of type *AmericanFilmDirector* in YAGO must become an instance of type *Person*, with the *occupation* attribute set to *Director* and *nationality* set to *American* if it is to be understood in terms of the DBpedia schema.

The research question for this PhD is as follows: *To what extent can elementary semantic matches be refined to complex matches – containing constraints and transformations – using a semi-automatic process based on classifying the matches against the Correspondence Patterns scheme.* In this context *elementary* matches are ones which match named ontology elements using relations such as subsumption. Existing semantic matching tools rely primarily on analyzing the structural information in ontologies, however as many data sources on the Semantic Web contain large amounts of instance data, this work will focus on techniques which use instance data to discover relationships in the structural data.

The following section of this paper outlines the state of the art in semantic matching and related techniques, as well as some of the shortcomings of current solutions. Section 3 discusses the proposed research approach, and the research methodology that will be employed. Section 4 outlines some preliminary results from an initial experiment.

2 State of the Art

This section outlines a survey of the state of the art in the semantic matching process, including tools, processes and formats for describing matches. It also includes a summary of schemes for classifying forms of heterogeneity, as well as machine learning tools which will be of benefit in identifying complex matches.

While it is held as impossible to completely automate the alignment process [4], several tools have been developed to assist a human operator with the endeavor. These include both automated candidate match generation [8], and graphical interfaces to assist with discovering and confirming correspondences between ontologies [9].

¹ <http://blog.dbpedia.org/2011/09/11/dbpedia-37-released-including-15-localized-editions/>

The INRIA Ontology Alignment API [6] provides a comprehensive set of automated match generators, drawn from the methods described by Shvaiko and Euzenat [8]. It also provides a format for describing matches, so that they may be exchanged.

Current semantic matching tools focus on discovering *equivalence* (\equiv) relationships, and less commonly *less general* (\sqsubseteq), *more general* (\sqsupseteq) and *disjointness* ($\not\equiv$) relationships. Schemes exist for classifying more detailed forms of relationships, these include Correspondence Patterns [5], and the THALIA framework [10]. The EDOAL [6] language provides a method for describing more complicated correspondences, using an OWL-like syntax, and was developed in conjunction with correspondence patterns. As yet, there are few matchers capable of generating the complex matches that EDOAL allows [6]. Ritze et. al. [11] describe a first attempt process for detecting complex matches, and Sváb-Zamazal et. al. [12] provide a set of pattern based tools for describing and managing these matches by relating them to ontology design principals.

The field of Machine Learning provides many tools which could be of use in analyzing matches between the less structured elements found in Linked Data. The Weka suite [13] provides a range of these tools as both an API and as part of a GUI. These include attribute selection tools such as Information Gain, and regression analysis tools.

3 Research Approach and Methodology

The focus of this research will be on the development of algorithms, methods and processes for refining elementary semantic relationships such as equivalence into more complex correspondences with conditions and transformations. The novelty of this approach lies in the use of instance element information to identify a correspondence pattern that best applies to a given ABox relationship and then using the pattern to guide the production of a complex correspondence.

For example, the *Class by Attribute Value* correspondence pattern occurs when a class in one ontology is equivalent to a class in a second ontology defined by instances with a specific property value. This pattern can be detected by the use of attribute selection methods, and once detected a declarative alignment may be created specifying the classes that match and the attribute and value condition necessary for the match to hold true. Section 4 outlines an experiment where an Information Gain based attribute selection method was used to detect correspondences of type *Class by Attribute Value* and *Class by Attribute Existence*.

A further pattern is that of the *Property – Relation Correspondence*, where a property in one ontology corresponds with a relation in another ontology. Here clustering could be used to group data values in the range of the property so they may be matched to the individuals in the range of the relationship. This would allow the creation of a declarative match consisting of a metric and cutoff point matching to a named individual. *Property Value Transformations* are another form of correspondence that can occur. This pattern would be detected when the values of matched properties for matched instances differ in some consistent way. For numerical properties, linear regression could be used to investigate what transformation is occurring, for example unit conversions or combinations such as $o_1\#price + o_1\#tax \rightarrow o_2\#price$.

Many of the techniques that will be investigated are reliant on analyzing instance data contained in the ontologies, and one possible limitation is that if the ontologies being mapped do not have suitable instance data these techniques will not be applicable. Because of this there will be a focus on Linked Data, as Linked Data sources contain large amounts of instance data, and often overlap to some extent. A further obstacle to this research will be the ability to measure the effectiveness of the matching techniques against a standard test set such as those used by the OAEI [14].

The methodology of this PhD consists of a literature review to establish the current state of the art in semantic matching tools, methods for their evaluation, and formats available for describing alignments. Following from this, techniques for detecting restriction type Correspondence Patterns such as Class by Attribute Value and Class by Attribute Existence and converting them to complex alignments will be developed. Techniques for analyzing Property Value Transformation correspondence patterns by regression or other means shall also be developed. A suitable test suite to evaluate the these techniques will be required, which should be made public to allow comparison with other matchers capable of producing complex matches

To date a tool has been developed which is capable of refining class matches to include declarative restrictions. An experiment was carried out to evaluate the ability of this system, and the results are described in the following section.

4 Current Work

An evaluation was carried out of the ability of a system to refine elementary correspondences between the class *Person* in DBpedia [7] and several classes of occupation in YAGO [15] to produce complex correspondences. Using matched instances of these classes that had been identified in both sources, attribute selection was used to test if restriction type matches were occurring between the classes, and determine which attribute these restrictions were dependent on. This evaluation demonstrated that the Information Gain measure is a suitable scoring function for finding the attribute and attribute value to condition matches of type *Class by Attribute Value* and *Class by Attribute Existence*. This Information Gain function allowed us to reliably select a gold standard correspondence pattern – consisting of an attribute and value to condition – as our top result in two of the four mappings tested, and reliably returns the best correspondence pattern in the top five results for all four test cases. In the cases where the search algorithm did not return the best correspondence pattern as the top result, this was because there were several patterns that could be considered valid and selecting the “best” among these was difficult. The IG score requires that a suitable training set of instances be used, and the evaluation demonstrated that a random sample of 30 instances was sufficient to rank the gold standard attribute and attribute-value pair in the top 5 results at least 89% of the time. The results of this experiment have been accepted for publication at the DANMS 2012 workshop.

Acknowledgement. This research is supported by the Science Foundation Ireland (Grant 08/SRC/I1403) as part of the FAME Strategic Research Cluster (www.fame.ie).

References

1. Weidman, S., Arrison, T.: Steps toward large-scale data integration in the sciences: summary of a workshop. National Academy of Sciences, 13 (2010)
2. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. *IEEE Transactions on Knowledge and Data Engineering* (preprint, 2012)
3. Klein, M.: Combining and relating ontologies: an analysis of problems and solutions. In: *Proc. of Workshop on Ontologies and Information Sharing, IJCAI*, pp. 53–62 (2001)
4. O’Sullivan, D.: The OISIN framework: ontology interoperability in support of semantic interoperability. PhD thesis, Trinity College Dublin (2006)
5. Scharffe, F.: Correspondence patterns representation. PhD thesis, University of Innsbruck (2009)
6. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The alignment API 4.0. *Semantic Web* 2(1), 3–10 (2011)
7. Bizer, C., et al.: DBpedia - A crystallization point for the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web* (7), 154–165 (2009)
8. Shvaiko, P., Euzenat, J.: A Survey of Schema-Based Matching Approaches. In: Spaccapetra, S. (ed.) *Journal on Data Semantics IV*. LNCS, vol. 3730, pp. 146–171. Springer, Heidelberg (2005)
9. Falconer, S.M., Storey, M.-A.: A Cognitive Support Framework for Ontology Mapping. 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. 114–127. Springer, Heidelberg (2007)
10. Hammer, J., Stonebraker, M., Topsakal, O.: THALIA: Test Harness for the Assessment of Legacy Information Integration Approaches. In: *Proc. of the Int. Conference on Data Engineering (ICDE)*, pp. 485–486 (2005)
11. Ritze, D., Meilicke, C., Sváb-Zamazal, O., Stuckenschmidt, H.: A pattern-based ontology matching approach for detecting complex correspondences. In: *Proc. of Int. Workshop on Ontology Matching, OM* (2009)
12. Sváb-Zamazal, O., Daga, E., Dudás, M.: Tools for Pattern-Based Transformation of OWL Ontologies. In: *Proc. of Int. Semantic Web Conference* (2011)
13. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter* 11(1), 10–18 (2009)
14. Euzenat, J., et al.: First results of the Ontology Alignment Evaluation Initiative 2011. In: *Proc. of 6th Int. Workshop on Ontology Matching, OM 2011* (2011)
15. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A large ontology from wikipedia and wordnet. *Web Semantics: Science, Services and Agents on the World Wide Web* 6(3), 203–217 (2008)

Data Linking with Ontology Alignment

Zhengjie Fan

INRIA & LIG

655, avenue de l'Europe, Montbonnot Saint Martin, 38334 Saint-Ismier, France
zhengjie.fan@inria.fr

Abstract. It is a trend to publish RDF data on the web, so that users can share information semantically. Then, linking isolated data sets together is highly needed. I would like to reduce the comparison scale by isolating the types of resources to be compared, so that it enhances the accuracy of the linking process. I propose a data linking method for linked data on the web. Such a method can interlink linked data automatically by referring to an ontology alignment between linked data sets. Alignments can provide them entities to compare.

Keywords: Data Linking, Ontology Alignment, Linked Data.

1 Motivation and Research Questions

Nowadays, countless linked data sets are published on the web. They are written with respect to different ontologies. Linking resources in these various data sets is the key for achieving a web of data. However, it is impossible for people to interlink them manually. Thus, many methods are proposed to link these data sets together. Here, I propose a data linking method based on ontology matching, which can automatically link data sets from different domains. Formally, data linking is an operation whose input are two collections of data. Its output is a collection of links between entities from both collections, in which there are binary relations on entities corresponding semantically to each other [4]. So, the research problem which will be tackled here is: given two RDF data sets, try to find out all possible “owl:sameAs” links between them automatically and correctly. My work is part of the Datalift¹ project, which aims to build a platform for data publishing. It is made of several modules, such as vocabulary selection, format conversion, interconnection and the infrastructure to host linked data sets. My work is to build the interconnection module, the last step of Datalift, that is, linking RDF data sets.

The paper is organized as follows. First, the state of the art on data linking is briefly analyzed in Section 2. Then in Section 3, a data linking method is introduced. Finally, Section 4 outlines the planned research methodology.

¹ <http://datalift.org/>

2 State of the Art

Data linking is the process of linking data, it can be done according to the results from comparing property values of instances in source classes with the ones of instances in target classes. Suppose there are m instances in the source class. There are n instances in the target class. So there should be $m * n$ comparison pairs. Thus, several techniques are proposed to reduce such comparison scale, while enhancing the precision and recall of the linking process.

One fundamental strategy is finding the keys of data sets. For key is used to identify and distinguish instances within the data set. The linking method only need to compare the property values within keys, then it can decide whether two instances are similar or not. So ideally, it can reduce the comparison scale of linking process. However, there are limitations for such strategy. On one hand, some key property cannot be used to compare, because they are meaningless out of the data source, such as the code or id. Usually different data sets have different coding formats. For these codes are not meaningful for human being, it is hard to find a transformation function. On the other hand, if the properties within a key do not have corresponding properties within one key of another data set. It is impossible to compare those keys for judging whether two instances are “owl:sameAs” or not. Thus, it is common to combine the key with other techniques such as machine learning [5,11].

Machine learning is a widely used strategy for data linking. It is used to find out the potential comparison pairs, as shown in [5,6,7]. That is, from which classes and properties values should be compared. For example, Hu et al. (2011) uses machine learning to enlarge the key property set for matching instances. The linking method in [5] considers not only key properties, but also uses machine learning to search frequently linked properties to find out similar instances using machine learning. There are two kinds of machine learning methods: supervised methods and unsupervised methods. Supervised methods use a training data set to find out the most suitable comparison pairs [5,8]. While more work focus on unsupervised methods for interlinking instances, for it saves time on collecting training sets [1]. Besides machine learning, graph structure and statistical technique are also used for finding out comparison pairs and reducing comparison scale [8]. Usually, these strategies are combined to fulfill the data linking process.

Above all, these linking strategies need to find out the linking pattern between two data sets. That is to say, which comparison pairs are more likely to contain similar instances, which pairs are not. Such linking pattern can be found out more easily with certain heuristics, such as ontology alignment. Ontology Alignment contains correspondences which may be used as pairs of entities from which to find instance to compare. It could be the corresponding classes, or corresponding properties, or corresponding class and property with restrictions. Thus, it can be used to automate the data linking process.

There are several research problems on using ontology matching for data linking. First, what kind of correspondence can be used for data linking? what cannot? Second, which correspondence or group of correspondences can

efficiently help comparing data? How far ontology matching can enhance the linking speed and accuracy? What its main advantage over other techniques? What are its limitations? In which case, it cannot efficiently help linking RDF data sets?

3 Proposed Approach

As a well-known data linking tool, SILK [2] is designed to execute the data linking process, following manually written scripts specifying from which class and property the value should be compared, as well as which comparison method is used. So, I plan to realize the data linking process by transforming alignments into SILK scripts. Then the data linking process can be triggered on SILK afterwards.

The data linking method proposed here is illustrated in Fig. 1. Suppose there are two data sets to be linked. Firstly, their vocabularies, namespaces or ontology URIs are sent to an Alignment Server, which is an alignment storage [3], so as to check whether there is an ontology alignment available. If there is an alignment and it is written in EDOAL² [10], which is an expressive language for expressing correspondences between entities from different ontologies, or not. It cannot only express correspondences between classes and attributes, but also express complex correspondences with restrictions. Then I directly produce a SILK script. If it is not written in EDOAL, then each concept's keys are computed according to the TANE algorithm [12], which is for finding out functional and approximate dependencies between properties of data sets, or coverage and discriminating rate of the properties. If the Alignment Server does not contain any alignment, then the vocabularies are searched. And an ontology matcher is used to generate an alignment between the data sets' ontologies. So, the linking method introduced here has limitations when there is no correspondences or vocabularies available. It will take extra time to compute the data set's ontology and ontology alignment.

At the early stage of my PHD research, I simplify my data linking method as "extracting correspondences information from alignment written in EDOAL to generate SILK script". After it is successfully done, the key will be taken into consideration to complete the picture.

4 Planned Research Methodology

i Methods for data collection and presentation

For the work is part of the Datalift project, geographical RDF data sets provided by Datalift will be tested with my data linking method. Furthermore, data from the Linked Data Cloud will also be tested. An interface for the interconnection module will be built, which not only shows the owl:sameAs links, but also shows the details of instances.

² <http://alignapi.gforge.inria.fr/edoal.html>

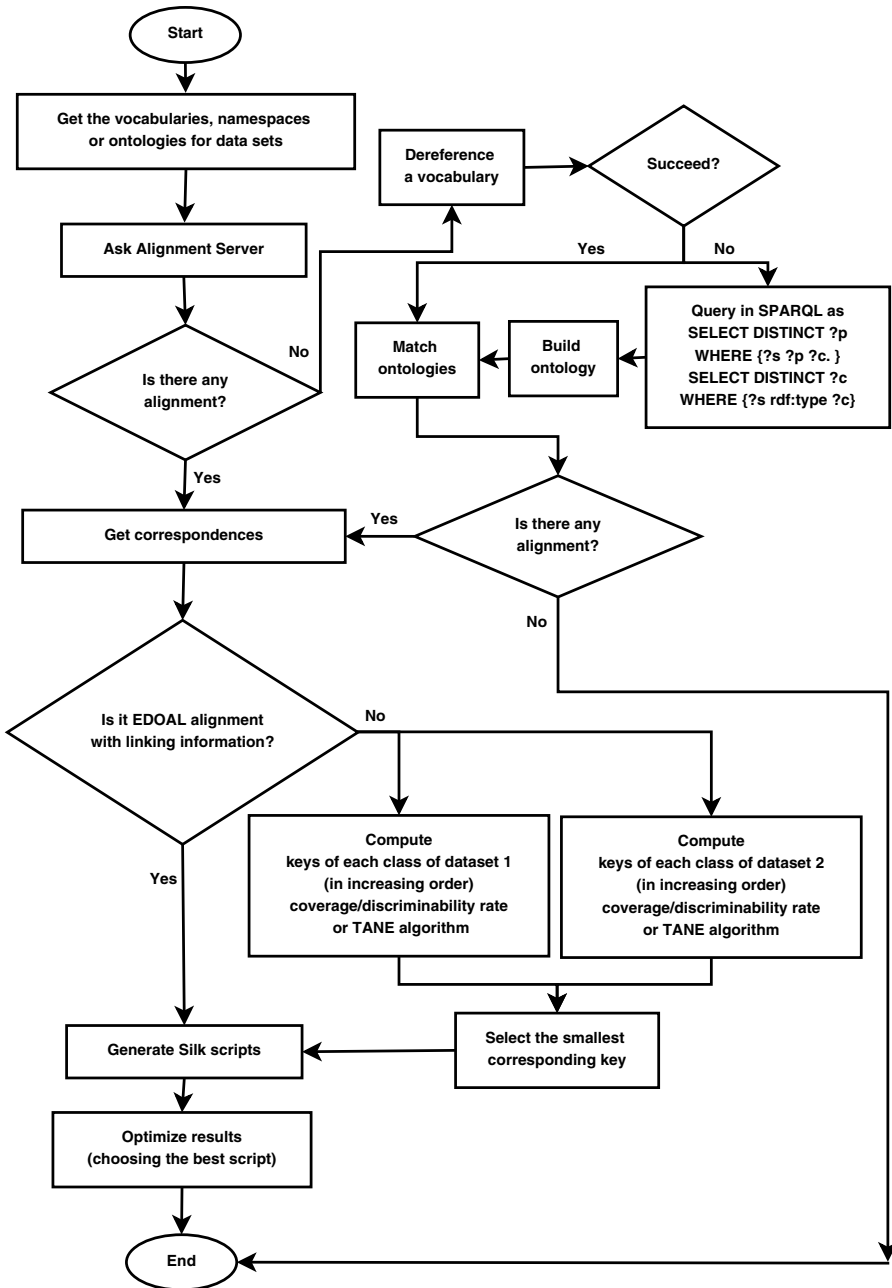


Fig. 1. Workflow of Data Linking Method Based on Ontology Alignment

ii Methods for data analysis and evaluation

The linking method will take part in IM@OAEI. I will test my linking method on OAEI data sets. The precision, recall and duration of the method will be computed. If there are several SILK script produced, the result will be analyzed and improved as below:

- i Compute the duration, precision and recall of producing and running each script. Find out the wrong linkings and the missing linkings of each script.
- ii Compute the average duration, precision and recall. Pick out the script S_a whose duration, precision and recall near the average values. Pick out the scripts S_d , whose duration is the smallest, S_p , whose precision is the highest, S_r , whose recall is the highest.
- iii Find out which linkings cost more time to be found by comparing the linking set of S_d with S_p , S_d with S_r . What correspondences they belong to. What kind of wrong links tend to be produced by comparing the linking set of S_a with the linking set of S_p . How to adjust the script to increase the precision and recall for less wrong linkings and missing linkings.

References

1. Araújo, S., Hidders, J., Schwabe, D., de Vries, A.P.: SERIMI - Resource Description Similarity, RDF Instance Matching and Interlinking. CoRR. abs/1107.1104 (2011)
2. Bizer, C., Volz, J., Kobilarov, G., Gaedke, M.: Silk - A Link Discovery Framework for the Web of Data. CEUR Workshop Proceedings, vol. 538, pp. 1–6 (2009)
3. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The Alignment API 4.0. Semantic Web Journal 2(1), 3–10 (2011)
4. Ferrara, A., Nikolov, A., Scharffe, F.: Data linking for the Semantic Web. International Journal of Semantic Web in Information Systems 7(3), 46–76 (2011)
5. Hu, W., Chen, J., Qu, Y.: A Self-Training Approach for Resolving Object Coreference on the Semantic Web. In: Proceedings of WWW 2011, pp. 87–96. ACM (2011)
6. Isele, R., Bizer, C.: Learning Linkage Rules using Genetic Programming. In: OM 2011. CEUR Workshop Proceedings, vol. 814 (2011)
7. Ngonga Ngomo, A.-C., Lehmann, J., Auer, S., Höffner, K.: RAVEN - Active Learning of Link Specifications. In: OM 2011. CEUR Workshop Proceedings, vol. 814 (2011)
8. Nikolov, A., Uren, V.S., Motta, E., Roeck, A.N.D.: Handling Instance Coreferencing in the KnoFuss Architecture. In: IRSW 2008. CEUR Workshop Proceedings, vol. 422, pp. 265–274 (2008)
9. Raimond, Y., Sutton, C., Sandler, M.: Automatic Interlinking of Music Datasets on the Semantic Web. In: LDOW 2008. CEUR Workshop Proceedings, vol. 369 (2008)
10. Scharffe, F.: Correspondence Patterns Representation. PhD thesis, University of Innsbruck (2009)
11. 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)
12. Huhtala, Y., Kärkkäinen, J., Porkka, P., Toivonen, H.: TANE: An Efficient Algorithm for Discovering Functional and Approximate Dependencies. The Computer Journal 42(2), 100–111 (1999)

A Semantic Policy Sharing Infrastructure for Pervasive Communities

Vikash Kumar

The Telecommunications Research Center Vienna (FTW), Austria
kumar@ftw.at

Abstract. We propose a policy sharing infrastructure that enables a semantic rule created in one set of environmental, physical and contextual settings to be translated, while maintaining the underlying semantics, for use in a situation when those settings/parameters change. With an aim to enable sharing of useful intelligence described in the form of policies, we base our discussions on smart home and mobile commerce use cases.

Keywords: Policy sharing, Semantic Rules, Smart homes, mCommerce.

1 Introduction and Motivation

Rules are increasingly being used in semantic applications as well as in traditional IT systems to provide a formal and powerful way of representing information like individual preferences, privacy constraints etc [1,2]. To make such systems more versatile, there have been many initiatives that aim to transform a rule created in one semantic standard to one created in another [3,5,10]. However, in all these cases there is no provision for a rule or a policy¹ to be shared or reused among applications where its semantic meaning is preserved even while the context and other parameters of respective application environments change. This thesis will elaborate on a solution for an intelligent semantic policy sharing infrastructure that enables the agents of an application across different settings, context, environments, etc. to share and reuse semantic policies amongst themselves. For this we identify variables that affect the definition of a policy at different usage sites of the application and provide them as input to an underlying translation engine. The investigations are carried out in the use case of a smart home community as well as in an mobile commerce (m-commerce) setting.

Smart meters are increasingly being installed in common households in most countries and at the same time there is an increasing tendency of people residing in living communities like in apartments of a common large building in the current urban landscape. A natural next step in such a scenario would be to further utilize the advantages of Information Communication Technologies (ICT) in creating connected urban environments which promote increased partnerships among residents of a living community through better information sharing and transparency [11,12]. A part of such information sharing and exchange of ideas

¹ A policy consists of one or more rules.

can be in the form of policies that the resident of an apartment creates for energy saving and/or better user experience in his/her respective home.

In the m-commerce use case, we aim to use telecommunications (telco) specific information like identity, location, etc. in providing personalized advertisements (ads) to users based on their own preferences set in the form of policies. The proposed infrastructure would enable them in getting better recommendations by sharing efficient policies among acquaintances in a Web 3.0 environment.

The main research question to be investigated in this thesis is: *For the same application, can a semantic policy created for one set of physical, environmental and contextual conditions and settings be effectively translated and applied into another different set of settings while preserving its core idea?* Along the way, this work will also show how to translate the benefits achieved (in terms of cost/energy savings, etc.) by application of a policy from one environment to another.

2 Related Work

There has been considerable interest in the areas of rule interchange and profile matching in the semantic web community. RuleML [6] (Rule Markup Language) was the first initiative aimed at creating a unifying family of XML-serialized rule languages that includes all the web rules. The REVERSE II Rule Markup Language (R2ML) furthered this cause by proposing a comprehensive XML rule format by integrating languages like OCL, SWRL and RuleML [10].

W3C launched the Rule Interchange Format (RIF) [5] working group in 2005 tasked with producing a core rule language using which rules can be represented across all systems. The RIF framework for rule-based languages consists of a set of *dialects* which formally describes information about the syntax, semantics and XML serialization of a language. A semantics-enabled layered policy architecture has been proposed in [3] as an extension of W3C's Semantic Web architecture aimed at facilitating the exchange and management of policies created in multiple languages across the web.

Several projects have tried to utilize the benefits of smart meters and build applications and services around data collected by them [9,11,12]. Several others have suggested the use of mobile specific enablers for providing privacy aware services based on semantic rules for mobile users [13,14].

While this thesis takes inspirations from the existing works in the direction of rule interchange and sharing, the unique feature of the proposed infrastructure will be the *translation* of a policy created in one set of conditions into that in another set of conditions for the "same" application in a privacy aware manner. Therefore the focus won't be on *application or language independence* as proposed in other approaches.

3 Approach and Methodology

In order to re-use the ideas and best practices of different users, the first requirement would be to serialize their preferences and policies in a common format

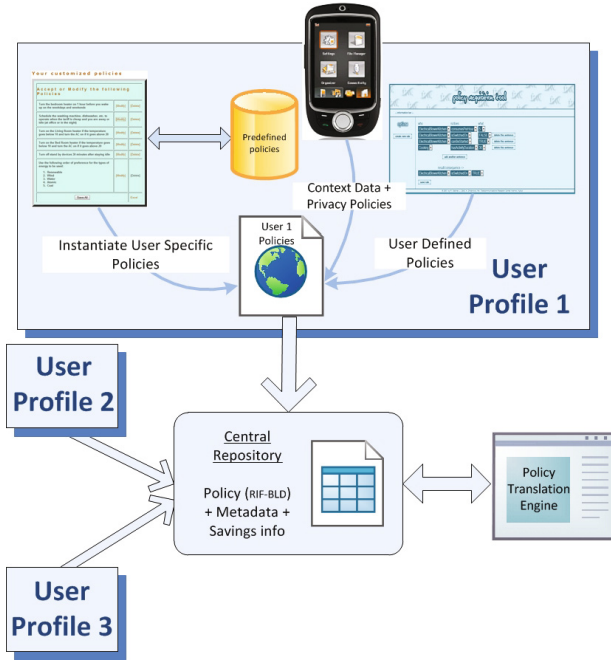


Fig. 1. The policy sharing infrastructure

(possibly *RIF-BLD* [5]) which could easily be adapted from one array of settings to another. Thereafter, for the translation, we need to first identify the static and dynamic parameters of a rule and then replace the dynamic ones with those matching the new set of conditions. For example, user *X*'s policy: “Send me all offers for *iPad3* from *Saturn electronic store*” will be translated for user *Y* as “Send me all offers for *Samsung Galaxy tab* from *Amazon*” where *Y* is interested in electronic ads but his/her profile shows an inclination for Samsung products (rather than Apple) and the shopping behavior shows transactions mostly from Amazon (rather than Saturn). The italicized and underlined parts represent the static and dynamic contents of the rule respectively. Other inferences affecting this translation obtained by reasoning over user profile and domain ontology, their online behavior and system policies may be facts like *iPad3* is a device in *Tablet* category while *Saturn* is a shop in the category *Electronic Store*. A suitable technique for such a matching of ontology concepts needs to be developed. Another ontology containing the meta-policy information and other rules governing policy translation would be a part of this infrastructure. A central repository (Figure 1) will collect all the user policies annotated with their respective metadata containing information about their static and dynamic parameters, the perceived quantifiable advantage(s) achieved by their application, etc. It will also contain user and environment data like profile, preferences, temperature, etc. that will substitute the dynamic parameters of a rule.

A *Policy Translation Engine* performs the actual translation of individual rules in a policy from one setting to another using data available in the repository. It also computes the perceived savings attained by the application of the policy in both the settings and recommends the translated policy to interested users. The overall research is planned in various phases described below.

Preliminary Study & Analysis. A study of state of the art in various aspects of this project will be taken up. This would include a survey of existing rule based service enablers, semantic rule interchange standards, utilizing data from smart homes and personalized recommendation based m-commerce initiatives.

Specification of the Infrastructure. Based on the above inputs, a specification of the proposed semantic policy sharing infrastructure will be laid out including formal definitions of individual components.

Prototype Implementation. The prototype implementation would be done in several stages comprising of development of each module of the system followed by an integration phase. The modules envisaged as of now are (i) policy creation & modification tool, (ii) ontology development, (iii) the underlying policy translation module, (iv) intelligent context manager and recommendation engine for adaptation in telco scenario and (v) integration.

Prototype Evaluation. A separate user evaluation will be conducted for each module individually and for the prototype as a whole in context of both the use cases based on various parameters like accuracy, user acceptance, etc.

4 Preliminary Results

At the end of this thesis, we expect to have a working prototype of the policy translation infrastructure in the mentioned use cases. The focus of smart home use case would be to allow sharing of effective energy saving policies in a resident community and the evaluations would investigate the semantic similarity of the translated policy along with preciseness of calculated savings data. The m-commerce use case will primarily aim at using sharable policies as a tool for preserving user privacy while still being able to infer useful information from their publicly shared data for ad recommendations. Evaluations will be based on relevance of ads served by effect of original and translated policies.

5 Conclusions

In this paper, we introduced the idea of a semantic policy translation infrastructure and described some related work which would form a starting point of the research carried out in this thesis. Thereafter, we also mentioned some preliminary results of the work so far. According to the methodology shown in Section 3, the next major steps are to complete the policy translation engine, context management system as well as the policy recommendation tool. Finally, we intend to test our hypothesis through extensive user tests of the infrastructure proposed in this paper.

Acknowledgments. This work has been partially supported by FFG funded SESAME-S project [4] and partly by APSINT [8]. The Telecommunications Research Center Vienna (FTW) is supported by the Austrian government and the City of Vienna within the competence center program COMET. Thanks to Dr. Anna Fensel and Prof. Gabriele Kotsis for supervising this work.

References

1. Toninelli, A., Jeffrey, B.M., Kagal, L., Montanari, R.: Rule-based and Ontology-based Policies: Toward a hybrid approach to control agents in pervasive environments. In: *Semantic Web and Policy Workshop (2005)*
2. Grosz, B.N.: Representing E-Business Rules for the Semantic Web: Situated Courteous Logic Programs in RuleML. In: *Workshop on Information Technologies and Systems, WITS 2001 (2001)*
3. Hu, Y.J., Boley, H.: SemPIF: A Semantic meta-Policy Interchange Format for Multiple Web Policies. In: *Proc. of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2010)*, vol. 1, pp. 302–307. IEEE Computer Society, Washington, DC (2010)
4. The SESAME-S Project, <http://sesame-s.ftw.at/>
5. Rule Interchange Form, <http://www.w3.org/2005/rules/>
6. RuleML, <http://www.ruleml.org/>
7. Zhdanova, A.V., Zeiß, J., Dantcheva, A., Gabner, R., Bessler, S.: A Semantic Policy Management Environment for End-Users and Its Empirical Study. In: Schaffert, S., Tochtermann, K., Auer, S., Pellegrini, T. (eds.) *Networked Knowledge - Networked Media. SCI*, vol. 221, pp. 249–267. Springer, Heidelberg (2009)
8. The APSINT Project, <http://www.apsint.ftw.at/>
9. Kumar, V., Tomic, S., Pellegrini, T., Fensel, A., Mayrhofer, R.: User Created Machine-Readable Policies for Energy Efficiency in Smart Homes. In: *Proc. of the Ubiquitous Computing for Sustainable Energy (UCSE 2010) Workshop at the 12th ACM International Conference on Ubiquitous Computing, UbiComp 2010 (2010)*
10. Bădică, C., Giurca, A., Wagner, G.: Using Rules and R2ML for Modeling Negotiation Mechanisms in E-Commerce Agent Systems. In: Draheim, D., Weber, G. (eds.) *TEAA 2006. LNCS*, vol. 4473, pp. 84–99. Springer, Heidelberg (2007)
11. Möller, S., Krebber, J., Raake, E., Smeele, P., Rajman, M., Melichar, M., Pallotta, V., Tsakou, G., Kladis, B., Vovos, A., Hoonhout, J., Schuchardt, D., Fakotakis, N., Ganchev, T., Potamitis, I.: INSPIRE: Evaluation of a Smart-Home System for Infotainment Management and Device Control. In: *Proc. of the LREC 2004 International Conference, Lisbon, Portugal*, pp. 1603–1606 (2004)
12. Kamilaris, A., Trifa, V., Pitsillides, A.: HomeWeb: An application framework for Web-based smart homes. In: *18th International Conference on Telecommunications (ICT)*, pp. 134–139 (2011)
13. Gandon, F.L., Sadeh, N.M.: Semantic web technologies to reconcile privacy and context awareness. *Journal of Web Semantics* 1, 241–260 (2004)
14. Toninelli, A., Montanari, R., Kagal, L., Lassila, O.: A Semantic Context-Aware Access Control Framework for Secure Collaborations in Pervasive Computing Environments. 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. 473–486. Springer, Heidelberg (2006)

Involving Domain Experts in Ontology Construction: A Template Based Approach

Muhammad Tahir Khan*

FBK-IRST, Via Sommarive 18, 38050 Trento, Italy
tahir khan@fbk.eu

1 Introduction

The availability of knowledge encoded in computer-readable forms, and expressed according to precise conceptual models and formal languages such as the ones provided by ontology languages, is an important pillar for the provision of flexible and better integrated ways of handling content, production processes, and knowledge capital of organisations and enterprises.

In spite of the efforts and progresses made in the area of knowledge elicitation and modelling, methodologies and tools available nowadays are mainly tailored to Knowledge Engineers [1], that is, people who know how to create formal conceptualisations of a domain, but do not know the domain to be modelled. These tools are instead scarcely usable by Domain Experts, that is, the people who know the organisation's capital, but often do not have any skills in formal model creation. As a result the interaction between Knowledge Engineers and Domain Experts is regulated by rigid iterative waterfall paradigms which make the process of producing and revising good quality ontologies too complex and expensive for the needs of business enterprises.

The work of the MoKi [2] project aims at producing a Web2.0 tool able to support an active and agile collaboration between Knowledge Engineers and Domain Experts, as well as the mining of the organisation's content, to facilitate the production of good quality formal models. The work of this thesis aims at improving MoKi in supporting a more agile construction of good quality ontologies by encouraging Domain Experts to actively participate in the construction of the formal part of the model. More in detail it aims at exploring: (i) how templates, based on precise characterisations provided by Top Level or specific Domain ontologies, can be used to describe knowledge at a (semi-)formal level, and (ii) how Ontology Design Patterns (ODPs) can be used to reuse existing knowledge without having to know the details of the underlying languages nor the ODPs in all their detail. We explore and apply our ideas to the construction of Enterprise Models [3], which provide the use case for this thesis.

2 Previous and Related Work

Our work touches upon different areas of research. We focus here on Enterprise modelling and ontology engineering.

* Thesis Supervisor: Dr. Chiara Ghidini.

¹ moki.fbk.eu

Enterprise Modelling Approaches: The work in [4] presents a methodology called Enterprise Knowledge Development Method (EKD); a participative approach to enterprise modelling. In this approach stakeholders meet in modelling sessions to create models collaboratively and document them on large plastic sheets using paper cards. It requires an extra effort of transferring the models from plastic sheets to computer based models by modelling technicians. The work in [2] proposed a collaborative framework and a tool called MoKi, that supports the the creation of articulated enterprise models through structured wiki pages. Moki enables heterogeneous teams of experts, with different knowledge engineering skills, to actively collaborate in a modelling process. A comparison of MoKi and some other wiki-based tools is contained in [2] and due to lack of space we do not elaborate further on the comparison.

Ontology Engineering Approaches: There has been some work done on capturing organisational aspects through the use of an enterprise ontology. In [5], a preliminary proposal of an ontology of organisations based on DOLCE ontology [6] was presented. They propose that the ontological analysis of an organisation is the first fundamental step to build a precise and rigorous enterprise model. DOLCE is a top level ontology describing very general concepts which are independent of a particular problem or domain. So it is required to provide some support for Domain Experts to work with them, which is missing in [5]. A tutoring methodology called TMEO, based on ontological distinction embedded in owl-lite version of DOLCE was proposed in [7] to guide humans in elicitation of ontological knowledge. In [8], a controlled natural language based approach was introduced to involve Domain Experts in the ontology construction process. This approach requires that the Domain Experts learn the controlled language before starting to model and it works with certain languages (e.g., English). An approach that attempts to enrich ontologies by finding partial instantiations of Content ODPs and refining ontology using axiomatised knowledge contained in ODPs, is found in [9]. This approach assumes the availability extended kind of ODPs which contain additional lexical information, and it is more focused toward Ontology Engineers.

3 Supporting Domain Experts in Conceptual Modelling with Moki

State of the art methodologies and tools for ontology construction usually ask Domain Experts to provide / revise knowledge (sources), and Knowledge Engineers to formally encode that knowledge, but lack suitable guidance and support to ensure their active involvement in the construction of the formal part of the model (as they often do not have required skilled in conceptual modelling methodologies and tools). Nevertheless, as argued in [8] involving the Domain Experts in authoring (rather than only providing knowledge for) ontologies is an important task for improving the process of ontology construction and making it more agile. To facilitate the involvement of Domain Experts, we have implemented a collaborative (enterprise) modelling tool called MoKi, that supports access to the enterprise model at different levels of formality (informal, semi-formal, and formal) and encourages Domain Experts to actively participate in the construction of the formal part of the model by providing not only access to the knowledge inserted by Knowledge Engineers, but also comment or directly modify part of it. In this thesis we aim at involving Domain Experts in the construction of ontologies by

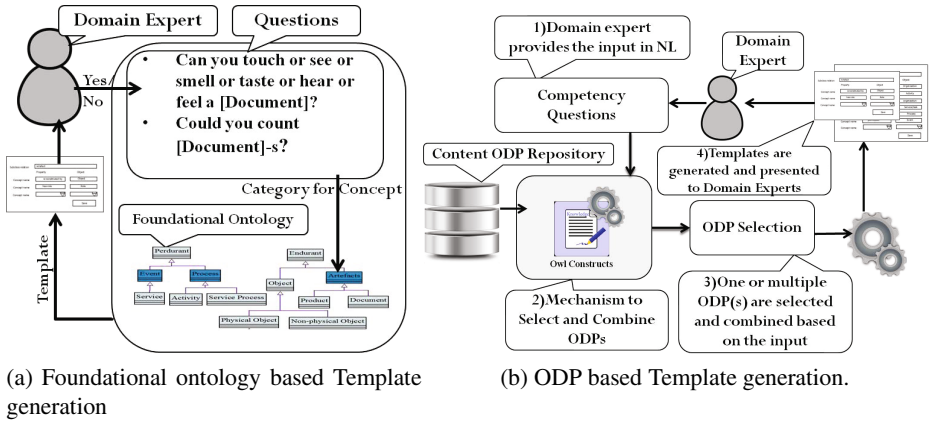


Fig. 1. An overview of our approach

using “templates” for helping them in describing knowledge at a (semi-)formal level, and Ontology Design Patterns, for helping them in reusing existing knowledge. So the idea is to start from the initial version of Moki and further develop it in order to guide and facilitate Domain Experts in entering *rich* and *good quality* knowledge, even if they do not have a deep understanding of knowledge engineering techniques and concepts. We illustrate these ideas in the following.

Helping Domain Experts via Templates: Top level ontologies can be considered as a set of formal guidelines for domain modelling. They can serve as a starting point for building new ontologies as they answer the question what things are there in the domain to be modelled. We are starting with the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [6] and analyse which of its ontological categories can be used in Moki to facilitate the modelling activities of Domain Experts by providing a precise characterisation of the main “types” of entities in their domain. The intention is to select the DOLCE categories which are easy to grasp and to work with for Domain Experts. Through this approach, we intend to answer the following questions:

1. How can a top level ontology be used to guide the overall process of model construction by Domain Experts, in particular through generating templates from it?
2. How can users be guided to select the appropriate template?

Concerning the first question, template generation has been done to some extent for ODPs but it is more focused toward knowledge engineers. Moreover, it needs to be extended, as usability aspects have not been studied and it has not been used for top level ontologies. A template has a predefined information with placeholders and parameters for capturing required information. This means, it will help Domain Experts to know what information to fill in according to its position within the template, which is missing in the controlled natural language based methods. As shown in Figure 1a, the main idea is to provide “templates”, based on foundational ontology, for the Domain Experts in order to perform the characterisation of entities in a “guided” manner. These categories are selected after reviewing the material available in DOLCE and different extensions of DOLCE, which will act as a starting point of the modelling activity. The idea is to

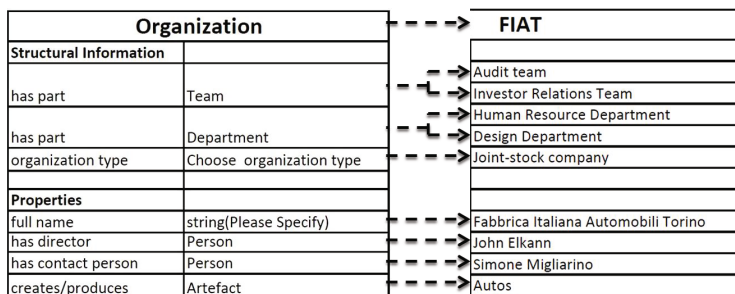


Fig. 2. Organization template and its Instantiation

characterise the main entities with DL axioms, provide a mechanism that transform the logical characterisation in templates (for the specification of sub-concepts or individuals) and guiding user through the different fields of the templates by providing list of possible entities suitable to fill in the different fields of the templates. Here we present a portion of organization template and its instantiation in Figure 2 to show that how these templates can be used to model real world entities. As shown in Figure 2, the arrows represent the instantiation of Organization template to a particular organization “FIAT” along with all the structural information and properties, such as “Audit Team” is one of a Team in “FIAT” and the director of the organization is “ John Elkann”.

Concerning the second question, we plan to use a Natural Language Question Answering (QA) system like TMEO [7] to help the user in selecting the correct category (and therefore template) to which the entity belongs. As shown in Figure 1a, the Domain Experts ask for assistance from the system to select the right template for the concept. The QA system will ask the Domain Expert a list of question and depending on the answers (Yes/No) given by the Domain Expert, the system will purpose the template.

Helping Domain Experts via Ontology Design Patterns: Our goal in this part is to discover whether some piece of knowledge that the Domain Expert needs to model can be efficiently modelled through the use of ontology design patterns (ODPs). ODPs provide repositories of already formalised knowledge and can be used as building blocks in ontology design, as shown in [10]. The initial line of research will be to explore if ODPs can be detected directly from competency questions. Through this approach, we intend to answer the following questions:

1. How can Content ODPs help Domain Experts to model complex distinctions in an ontology?

We aim at answering this question by providing a mechanism for Domain Experts to specify the requirements as competency questions and select the content ontology design patterns covering competency questions. In this direction, we have conducted a small experiment in [11] about detection of patterns in ontologies. However, this approach was fairly naive considering only concepts in the matching process. The concepts in patterns are abstract enough to match multiple competency questions, so the matching process will require more than matching concepts. It will also match relations

because they are the major indicators of pattern instantiation. As shown in Figure 1b, the process of identifying ODPs starts by taking input from Domain experts in the form of competency questions and then matching it against the Content Ontology Design patterns repository to see if there exist a complete match against this input or if it partially instantiates some ODP. If it partially instantiates some ODP then a recommendation will be given to the user suggesting to add the missing concepts and properties.

4 Results and Future Work

The result we have so far is based on a deep analysis of material related to the DOLCE ontology and its extensions, to extract a model to facilitate Domain Experts in their modelling task. We analysed all these extensions of the DOLCE ontology and taken out most common ontological categories that are easy to grasp for Domain Experts along with the relation existing between them and built an initial draft of the templates.

This work is done in the context of moki but the approach is general enough to be easily tailored and implemented for other tools. Currently we are working on the pre-evaluation of the templates with the Domain Experts, to verify the acceptability and usability of these templates before proceeding to implement them in MoKi. In future, we plan to provide an implementation of these templates in MoKi and incorporate a QA support system in MoKi to help the Domain Experts in selecting correct category. The next step will be to provide a mechanism for pattern selection on the basis of competency questions and generating templates from them. Once these changes are implemented we will perform a post-evaluation of updated moki in some use cases and also to conduct experiments with different types of users, e.g., ontology engineers as well as Domain Experts to verify the usability and effectiveness of the tool.

References

1. Noy, N.F., McGuinness, D.L.: *Ontology development 101: A guide to creating your first ontology* (2001)
2. Ghidini, C., Rospocher, M., Serafini, L.: *Conceptual modeling in wikis: a reference architecture and a tool*. In: *eKNOW*, pp. 128–135 (2012)
3. Fox, M.S., Grüninger, M.: *Enterprise modeling*. *AI Magazine* 19(3), 109–121 (1998)
4. Janis Bubenko Jr., J.S., Persson, A.: *User guide of the knowledge management approach using enterprise knowledge patterns* (2001), http://www.dsv.su.se/~js/ekd_user_guide.html/
5. Bottazzi, E., Ferrario, R.: *Preliminaries to a dolce ontology of organisations*. *Int. J. Business Process Integration and Management* 4(4), 225–238 (2009)
6. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A.: *WonderWeb deliverable D18 ontology library (final)*. Tech. Rep. (2003)
7. Oltramari, A., Vetere, G., Lenzerini, M., Gangemi, A., Guarino, N.: *Senso comune*. In: *Proceedings of the LREC 2010, Valletta, Malta* (May 2010)
8. Dimitrova, V., Denaux, R., Hart, G., Dolbear, C., Holt, I., Cohn, A.G.: *Involving Domain Experts in Authoring OWL Ontologies*. In: Sheth, A.P., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) *ISWC 2008*. LNCS, vol. 5318, pp. 1–16. Springer, Heidelberg (2008)

9. Nikitina, N., Rudolph, S., Blohm, S.: Refining ontologies by pattern-based completion. In: Proceedings of the Workshop on Ontology Patterns (WOP 2009), vol. 516 (2009)
10. Gangemi, A.: Ontology Design Patterns for Semantic Web Content. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 262–276. Springer, Heidelberg (2005)
11. Khan, M.T., Blomqvist, E.: Ontology design pattern detection - initial method and usage scenarios. In: SEMAPRO, Florence, Italy (2010)

Quality Assurance in Collaboratively Created Web Vocabularies

Christian Mader

University of Vienna, Faculty of Computer Science
`christian.mader@univie.ac.at`

Abstract. In recent years, controlled vocabularies have become available on the Web using SKOS¹, i.e. they are linked to each other in order to be used in an interoperable way. Well-crafted controlled vocabularies are beneficial for, e.g., search and retrieval systems that provide functionalities like search term completion, query expansion or the ability for inter-domain queries. Some of these vocabularies are created collaboratively by experts, holding expertise in different domains. In order to support vocabulary contributors to create high quality vocabularies, we propose a method that semi-automatically ensures vocabulary quality in collaborative authoring processes. The proposed approach tackles this issue by (i) defining a set of criteria that serve as metrics to measure vocabulary quality and (ii) introducing a method to continually assess and improve this quality. As a result of our approach, the developed vocabularies are expected to better fit the intentions of the contributors and are more useful for reuse and adoption on the Web of Data.

1 Motivation

Most institutions that build and publish controlled vocabularies create them for search and retrieval purposes, with specific functionalities in mind like, e.g., query expansion or faceted search [9]. However, shortcomings during the vocabulary creation process impinge upon these functionalities, affecting the effectiveness of the systems backed by these vocabularies, e.g., in terms of recall and precision. Among the problems arising in that context are missing relations between concepts, ambiguous labeling or lack of documentation. Furthermore, duplicate or abandoned entries, or logical contradictions might also be introduced in the vocabulary creation process.

However, when publishing a vocabulary on the Web, additional requirements have to be taken into consideration. Contributing to the Linked Data cloud involves providing references to other data sources in order “to connect disparate data into a single global data space” [6]. With the increasing availability of vocabularies expressed in SKOS, finding and utilizing a well-accepted and well-maintained vocabulary becomes even more challenging. Furthermore, as a consequence of the ever-changing nature of the Web, resources might also become unavailable, introducing the problem of “broken links”.

¹ Simple Knowledge Organization System, <http://www.w3.org/2004/02/skos/>

All of these issues could be subsumed under the term “controlled vocabulary quality”. It is important for various reasons: as mentioned above, quality assurance measures primarily aim to improve search and retrieval use-cases, since this traditionally has been a very common motivation for creation of controlled vocabularies. However, they can also serve to enhance the usage experience for human users who directly interact with the vocabulary itself, e.g., for getting an overview about the covered domain or incorporating changes.

Especially in open linked environments, vocabulary quality is crucial for acceptance of a vocabulary by others, which in turn is a key concept of the Linked Data principles. Once published as Linked Data, controlled vocabularies can and should be referenced, enhanced, and reused, and with the “building blocks” being of high quality, this is expected to happen to a much greater extent.

Research questions addressed in the proposed approach encompass: (i) what does “vocabulary quality” mean in open, collaboratively maintained environments and how can it be measured? (ii) how can quality assessment be integrated with collaborative vocabulary development environments? (iii) how does vocabulary quality assessment affect the quality of collaboratively created vocabularies?

2 Related Work

Existing standards for thesaurus construction [2,10] and manuals [3,7] propose guidelines and best practices for testing and evaluating controlled vocabularies. Many of them are hardly suitable for automatic assessment because additional knowledge about the creation process, target user group or intended usage would be required. [1] mentions vaguely formulated guidelines like, e.g., inclusion of “all needed facets” or adherence of the term form to “common usage”, whereas others, like “both BT and RT relationships occur between the same pair of terms” [3] are more precise and better suited for algorithmic evaluation. However, these guidelines are not specific for a concrete representation (e.g., SKOS) or form of publication (e.g., Linked Data, relational database, hardcopy).

In [8], Kless & Milton provide a list of measurements constructs for the intrinsic quality of thesauri, examining a thesaurus as an artifact itself, i.e. isolated from an application scenario. As stated by the authors themselves, the constructs (e.g., “Conceptual clarity” or “Syntactical correctness”) are “solely based on theoretical analysis” and application to existing thesauri is subject to their future work. Although undeniably useful for assessment by humans, algorithmic methods to measure the defined constructs are not covered. Furthermore, multilinguality and, since the paper focuses on intrinsic quality metrics, collaborative aspects of the creation process were not taken into consideration.

In the field of ontology engineering, metrics have been developed to evaluate and validate ontologies [5,11]. Common to these metrics is the fact that they are designed to be applied to general ontologies and instance data. As a consequence, they either do not deal with specific requirements in development of controlled vocabularies and applicability of the metrics for measuring vocabulary quality is still unclear.

Various initiatives that create controlled vocabularies publish details of their construction and validation process on the Web, such as the National Cancer Institute thesaurus (NCIt) or Food and Agriculture Organization (FAO). Despite employing different guidelines and (proprietary) tools [4], certain methods (e.g., duplicate checks) can be abstracted that prove useful in other domains.

3 Proposed Approach

In recent years, SKOS has been adopted by many organizations² as a technology for expressing vocabularies on the Web in a machine-readable format. As a consequence, our approach focuses on processing vocabularies represented in the SKOS language.

The overall goal of the approach is to ensure iterative improvement of a controlled vocabulary’s quality in a collaborative development process. The “View” in Figure 1 constitutes contributors taking part in the collaborative process. At the core of the work is the “Quality Controller” component, which is based on a catalog of quality criteria (cf. Table 1) and acts as a proxy between view and model, managing quality assessment, user notification, and concurrency issues.

Upon instantiation the quality controller is parameterized with a vocabulary (the model). Every contributor has to register at the quality controller by providing contact information and gets her own “working copy” of the vocabulary. The quality of this working copy is analyzed on every relevant modification. Based on this analysis, notification messages are created, containing information about quality issues. These messages are then disseminated to the contributor who can now decide to fix the issues or keep the current state of the vocabulary. Eventually the changes of the contributor are synchronized with the model.



Fig. 1. Conceptual overview of the proposed approach

Quality assessment of the vocabulary is not only triggered on user interaction, but also after a certain period of time. This is due to the fact that (i) the model might also be changed by contributors bypassing the quality controller and (ii) because the quality of the vocabulary may be affected by changes and independent evolution of other vocabularies on the Web.

Based on existing research and evaluation of SKOS vocabularies available on the Web, a (preliminary) catalog of quality criteria (cf. Table 1) has been identified that can be automatically evaluated. The fact that at least one violating

² e.g., AGROVOC, GEMET, Standard Thesaurus Wirtschaft.

Table 1. Identified quality criteria

Qual. Criterion	Description	Example of Impact
Loose Concepts	Concept with no hierarchical or associative relations to other resources.	No hierarchical query expansion
Weakly Conn. Components	Subgraphs within the vocabulary that are not connected to each other.	Obstructive for understanding and querying
Cyclic Relations	Cycles break the hierarchy, might reveal logical problems.	No drill-down search possible
Lack of External Links	No linkage to resources in foreign namespaces (<i>external</i> resources).	Gathering knowledge from other domains
Unavailable Resources	Resources must be dereferencable via their HTTP URIs (no broken links).	Information content
Low Concept In-degree	Concepts that serve as link targets in other vocabularies.	Impression on the vocabulary acceptance
SKOS Inconsistency	Conflicts with consistency criteria of SKOS reference or invention of new terms within SKOS namespace.	Standard conformance
Deprecated Property Usage	Some properties have been removed from the current SKOS version.	Interoperability
Poor internationalization	Inclusion of language tags and concepts labeled with same set of languages.	Translation use-cases
Ambiguous Labeling	SKOS labels are pairwise disjoint; avoid identical labels for different concepts.	Retrieval precision
Unconnected Related Concepts	Concepts labeled (slightly) different but mean the same and are not hierarchically or associatively connected.	Expose structural misconceptions
Lack of Documentation	Usage of properties documenting vocabulary concepts in human-readable form.	Disambiguation

vocabulary for each criterion could be found on the Web indicates practical relevance of this catalog.

4 Research Methodology

In a first step, as a **problem definition and state-of-the-art survey**, existing publications targeting data quality, vocabulary and thesaurus development as well as ontology building principles will be reviewed. It is important to find out to what extent quality criteria in these areas exist and to elaborate on their importance to controlled vocabularies. Based on the findings in the first step, we **propose a set of general quality criteria** for controlled vocabularies. The result of this step is a list of criteria together with algorithms that allow for programmatic evaluation. After that, **implementation of the tools**, i.e. a library (API) that creates a metrics based on the quality criteria, will be started. In the course of an **analysis** step, existing vocabularies available on the Web will be evaluated against the quality criteria. Community feedback collected in this step might lead to adjusting and reformulating the quality criteria which target research question (i). To

evaluate the approach, instantiating (or integrating into) a collaborative vocabulary development process is essential, addressing research question (ii). A valid setting would be to assign two comparably skilled groups of users with the task of concurrently creating a vocabulary. The continuous quality assessment of the approach will be activated for both groups, with only one group getting support by the automatic feedback mechanism. That way it is possible to track the evolution of vocabulary quality in both groups, obtaining information how the quality assessment influences development and contribute to research question (iii). If those groups that are supported by automatic quality feedback, develop higher-quality vocabularies, the proposed approach is said to be successful.

5 Preliminary Results and Conclusion

Based on the identified criteria, qSKOS³, an open source library for vocabulary quality assessment, has been created. First results⁴ in utilizing the library on various vocabularies were promising and it will be continually updated based on the community's feedback and as research progresses. qSKOS also provides the basis for further research regarding quality implications in collaborative settings.

References

1. Proceedings of ACM/IEEE 2003 Joint Conf. on Digital Libraries (JCDL 2003), Houston, Texas, USA, May 27-31. IEEE Computer Society (2003)
2. Information and documentation – thesauri and interoperability with other vocabularies – part 1: Thesauri for information retrieval. Norm (Draft) ISO 25964-1, Int. Org. for Standardization, Geneva, Switzerland (2011)
3. Aitchison, J., Gilchrist, A., Bawden, D.: Thesaurus construction and use: a practical manual. Aslib IMI (2000)
4. de Coronado, S., et al.: The nci thesaurus quality assurance life cycle. *Jour. of Bio. Inf.* 42(3), 530–539 (2009)
5. Gangemi, A., Catenacci, C., Ciaramita, M., Lehmann, J.: Ontology evaluation and validation: an integrated formal model for the quality diagnostic task. Tech. rep., Lab. of Applied Ontologies – CNR, Rome, Italy (2005)
6. Heath, T., Bizer, C.: *Linked Data: Evolving the Web into a Global Data Space*, 1st edn. Morgan & Claypool (2011), <http://linkeddatabook.com/>
7. Hedden, H.: *The accidental taxonomist*. Information Today (2010)
8. Kless, D., Milton, S.: Towards Quality Measures for Evaluating Thesauri. In: Sánchez-Alonso, S., Athanasiadis, I.N. (eds.) MTSR 2010. CCIS, vol. 108, pp. 312–319. Springer, Heidelberg (2010)
9. Nagy, H., Pellegrini, T., Mader, C.: Exploring structural differences in thesauri for skos-based applications. In: *I-Semantics 2011*, pp. 187–190. ACM (2011)
10. NISO: ANSI/NISO Z39.19 - Guidelines for the Construction, Format, and Management of Monolingual Controlled Vocabularies (2005)
11. Tartir, S., Arpinar, I.B.: Ontology evaluation and ranking using ontoqa. In: *ICSC 2007*, pp. 185–192 (2007)

³ The source code can be downloaded from <https://github.com/cmader/qSKOS/>

⁴ https://github.com/cmader/qSKOS4rb/raw/master/results/qskos_results.ods

Author Index

- Aanen, Steven S. 300
Abdul Manaf, Nor Azlinayati 270
Adamou, Alessandro 844
Ahmed-Baker, Maha 674
Alani, Harith 514
Allocca, Carlo 453
Ambite, José Luis 375
Antoniu, Ioannis 763
Arias Gallego, Mario 437
Ashraf, Jamshaid 813
Augenstein, Isabelle 210
- Bechhofer, Sean 270
Bernstein, Abraham 1
Berrueta, Diego 195, 590
Bischof, Stefan 838
Bloehdorn, Stephan 134
Boer, Victor de 733
Borbinha, José 718
Bratsas, Charalampos 763
Breyer, Matthias 633
Burel, Grégoire 514
Burkhardt, Dirk 633
- Caimi, Federico 179
Calado, Pável 718
Callahan, Alison 647
Carral Martínez, David 345
Champin, Pierre-Antoine 391
Chevalier, Edouard 603
Chiarcos, Christian 225
Clément de Saint-Marcq, Vianney le 391
Collins, Trevor 748
Corcho, Oscar 530
Cruz, Isabel F. 179
Cyganiak, Richard 778
- Dali, Lorand 484
Damljanovic, Danica 24
d'Aquin, Mathieu 119, 453
Deville, Yves 391
Di Francescomarino, Chiara 793
Dimou, Anastasia 763
Dragoni, Mauro 793
- Duc, Thanh Tran 56, 484
Dumontier, Michel 647
- El Ghali, Adil 195
Etcheverry, Lorena 469
- Fabiani, Alessio 179
Fan, Zhengjie 854
Feilmayr, Christina 808
Fernández, Javier D. 437
Fernández-Reyes, Francis C. 240
Fletcher, George H.L. 406
Fortuna, Blaž 484
Fräsincar, Flavius 300
Freire, Nuno 718
- García-Silva, Andrés 530
Gasevic, Dragan 315
Gerosa, Matteo 793
Ghidini, Chiara 793
Goel, Aman 375
González-Moriyón, Guillermo 590
Grondelle, Jeroen van 2
Groth, Paul 87
Guéret, Christophe 87
Gupta, Shubham 375
- Halevy, Alon 3
Harth, Andreas 56
Hartig, Olaf 8
Hatala, Marek 315
He, Yulan 514
Hidders, Jan 406
Hildebrand, Michiel 733
Hitzler, Pascal 345
Hogan, Aidan 687
Hoppermann, Christina 285
Horrocks, Ian 330
Huang, Yi 164
- Iglesias, Miguel 590
Ion, Patrick 763
Isaac, Antoine 733
- Jay, Nicolas 618
Jiang, Xueyan 164

- Kang, Jeon-Hyung 530
 Karampinas, Dimitris 545
 Kawamura, Takahiro 575
 Keet, C. Maria 240
 Khan, Muhammad Tahir 864
 Kjærnsmo, Kjetil 828
 Klusch, Matthias 499
 Knoblock, Craig A. 375
 Kohlhammer, Jörn 633
 Kohlhase, Michael 763
 Kołcz, Aleksander 4
 Kumar, Vikash 859
- Ladwig, Günter 56
 Lam, Monica S. 5
 Lampe, Ulrich 499
 Lange, Christoph 763
 Laublet, Philippe 24, 39
 Lehmann, Jens 87
 Lerman, Kristina 375, 530
 Lieber, Jean 618
 Lösch, Uta 134
 Luo, Yongming 406
 Lyko, Klaus 149
- Maali, Fadi 778
 Mader, Christian 870
 Magka, Despoina 330
 Mallick, Parag 375
 Martínez-Prieto, Miguel A. 437
 Meilender, Thomas 618
 Mellotte, Marc 687
 Mladenčić, Dunja 484
 Morales-González, Annette 240
 Motik, Boris 330
 Motta, Enrico 119, 453
 Mulholland, Paul 748
 Muslea, Maria 375
- Nagy-Rothengass, Márta 6
 Nazemi, Kawa 633
 Nederstigt, Lennart J. 300
 Ngonga Ngomo, Axel-Cyrille 149
 Nickel, Maximilian 164
 Nikolov, Andriy 119
- Oguchi, Kentaro 703
 Ohsuga, Akihiko 575
- Padó, Sebastian 210
 Palmonari, Matteo 179
 Palomares, Fabien 618
 Parundekar, Rahul 703
 Paulheim, Heiko 560
 Peristeras, Vassilios 778
 Picalausa, François 406
 Polo, Luis 195, 590
 Poveda-Villalón, María 833
 Powell, Gavin 687
- Raimond, Yves 255
 Rettinger, Achim 134
 Rohrer, Edelweis 818
 Rosati, Riccardo 360
 Rospocher, Marco 793
 Rowe, Matthew 39
 Rubiera, Emilio 195
 Rudolph, Sebastian 210
 Ruotsalo, Tuukka 422
- Sah, Melike 103
 Sandler, Mark 255
 Scheglmann, Stefan 659
 Scherp, Ansgar 659
 Schreiber, Guus 733
 Schulte, Stefan 499
 Servant, François-Paul 603
 Setz, Jochen 674
 Skjæveland, Martin G. 72
 Solnon, Christine 391
 Sonntag, Daniel 674
 Sperber, Wolfram 763
 Staab, Steffen 659
 Stab, Christian 633
 Stadler, Claus 87
 Stampouli, Dafni 687
 Stankovic, Milan 24, 39
 Steinmetz, Ralf 499
 Stevens, Robert 270
 Stolpe, Audun 72
 Stroe, Cosmin 179
 Studer, Rudi 56
 Szekely, Pedro 375
- Taheriyani, Mohsen 375
 Tejo-Alonso, Carlos 590
 Thalhammer, Andreas 823
 Trainotti, Michele 793
 Tresp, Volker 164

Triantafillou, Peter 545
Trippel, Thorsten 285

Vaisman, Alejandro A. 469
van de Laar, Julius 7
Vandić, Damir 300
van Gent, Judith 733
van Ossenbruggen, Jacco 733
Vansummeren, Stijn 406

Wade, Vincent 103
Wagner, Andreas 56
Walshe, Brian 849
Wielemaker, Jan 733
Wolff, Annika 748

Zillner, Sonja 674
Zinn, Claus 285
Zouaq, Amal 315