



11th International Semantic Web Conference ISWC 2012

November 11-15, 2012
Boston, USA

Proceedings of the ISWC 2012 Posters & Demonstrations Track:

ISWC 2012 Posters & Demos

Collected Abstracts

November 11-15, 2012

Editors

Birte Glimm
David Huynh

Preface

On behalf of the program committee for the ISWC 2012 Posters & Demonstrations Track, it is our great pleasure to present these proceedings, which form a collection of abstracts describing the posters and demos presented at the conference. The posters and demonstrations track of ISWC 2012 continues the established tradition of providing an interaction and connection opportunity for researchers and practitioners to present and demonstrate their new and innovative work-in-progress. The track gives conference attendees a way to learn about novel on-going research projects that might not yet be complete, but whose preliminary results are already interesting. The track also provides presenters with an excellent opportunity to obtain feedback from their peers in an informal setting from knowledgeable sources. As in previous years, we encouraged authors of accepted full research or in-use papers to present a practical demonstration or poster with additional results.

In total, there were 63 submissions, of which we accepted 18 demonstrations and 13 posters. We thank the authors of all submissions for their contributing to the program of ISWC 2012. As in 2011, the authors of accepted submissions will have the opportunity to briefly introduce their posters and demonstrations in a “minute madness” session to the audience of the conference.

We further would like to thank all the members of the program committee as well as the additional reviewers who have spent lots of their valuable time within a very tight schedule in prime holiday time. We thank all of these dedicated people for their valuable discussions and feedback, and we wholeheartedly appreciate their voluntary and enthusiastic cooperation. We are convinced to have arrived at an inspiring mix of posters and demos which tackle the Semantic Web idea from various angles and are looking forward to an exciting session at the conference.

Lastly, we want to thank our fellow organizers of ISWC, foremost the general chair, Abraham Bernstein, and Lalana Kagal for the local organization.

November 2012

Birte Glimm & David Huynh

Posters & Demonstrations Track – Organization

Program Chairs

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

Program Committee

Eva Blomqvist	Nadeschda Nikitina
Gianluca Correndo	Alexandre Passant Axel Polleres
Mathieu D'Aquin	Bernhard Schandl
Klaas Dellschaft	Wolf Siberski
Miriam Fernandez	Frantisek Simancik Giorgos Stoilos
Peter Haase	Nenad Stojanovic
Michael Hausenblas	Raphael Troncy
Pascal Hitzler	Tania Tudorache
Aidan Hogan	Jürgen Umbrich
Matthew Horridge	Maria Esther Vidal
Yue Ma	Haofen Wang
Yuan Ni	

Additional Reviewers

Berry, David	Krisnadhi, Adila A.
Carral Martínez, David	Pinkel, Christoph
Färber, Michael	Preusse, Julia
Huetter, Christian	Ye, Yuxin
Joshi, Amit	

Table of Contents

Part I

Demonstrations

Mining Patterns from Clinical Trial Annotated Datasets by Exploiting the NCI Thesaurus	1
<i>Joseph Benik, Guillermo Palma, Louiqa Raschid, Andreas Thor and Maria-Esther Vidal</i>	
The Linked Data Visualization Model	5
<i>Josep Maria Brunetti Fernández, Sören Auer and Roberto Garcia</i>	
QAKiS: an Open Domain QA System based on Relational Patterns	9
<i>Elena Cabrio, Julien Cojan, Alessio Palmero Aprosio, Bernardo Magnini, Alberto Lavelli and Fabien Gandon</i>	
DiscOU: A Flexible Discovery Engine for Open Educational Resources Using Semantic Indexing and Relationship Summaries	13
<i>Mathieu D'Aquin, Carlo Allocca and Trevor Collins</i>	
ourSpaces - A Semantic Virtual Research Environment	17
<i>Peter Edwards, Edoardo Pignotti, Alan Eckhardt, Kapila Ponnampeluma, Chris Mellish and Thomas Bouttaz</i>	
SPARQLoid - a Querying System using Own Ontology and Ontology Mappings with Reliability	21
<i>Takahisa Fujino and Naoki Fukuta</i>	
Simplifying MIREOT; a MIREOT Protege Plugin	25
<i>Josh Hanna, Chen Cheng, Alex Crow, Roger Hall, Jie Liu, Tejaswini Pendurthi, Trent Schmidt, Steven Jennings, Mathias Brochhausen and William Hogan</i>	
Browsing Causal Chains in a Disease Ontology	29
<i>Kouji Kozaki, Hiroko Kou, Yuki Yamagata, Takeshi Imai, Kazuhiko Ohe and Riichiro Mizoguchi</i>	
Real Time Fire Monitoring Using Semantic Web and Linked Data Technologies	33
<i>Kostis Kyzirakos, Manos Karpathiotakis, George Garbis, Charalampos Nikolaou, Konstantina Bereta, Michael Sioutis, Ioannis Papoutsis, Themistoklis Herekakis, Dimitrios Michail, Manolis Koubarakis and Charis Kontoes</i>	

Demonstrating Blank Node Matching and RDF/S Comparison Functions <i>Christina Lantzaki, Yannis Tzitzikas and Dimitris Zeginis</i>	37
Creating Enriched YouTube Media Fragments With NERD Using Timed-Text <i>Yunjia Li, Giuseppe Rizzo and Raphael Troncy</i>	41
Linked Data Fusion in ODCleanStore <i>Jan Michelfeit and Tomáš Knap</i>	45
Making Sense of Research with Rexplore <i>Enrico Motta and Francesco Osborne</i>	49
Quest: Efficient SPARQL-to-SQL for RDF and OWL <i>Mariano Rodriguez-Muro, Josef Hardi and Diego Calvanese</i>	53
A Prototype for Semantic based Diagnosis of Road Traffic Congestions .. <i>Marco Luca Sbodio, Freddy Lecue and Anika Schumann</i>	57
TwikiMe! - User profiles that make sense. <i>Patrick Siehndel and Ricardo Kawase</i>	61
Adding Realtime Coverage to the Google Knowledge Graph <i>Thomas Steiner, Ruben Verborgh, Raphael Troncy, Joaquim Gabarro and Rik Van de Walle</i>	65
Everything is Connected: Using Linked Data for Multimedia Narration of Connections between Concepts <i>Miel Vander Sande, Ruben Verborgh, Sam Coppens, Tom De Nies, Pedro Debevere, Laurens De Vocht, Pieterjan De Potter, Davy Van Deursen, Erik Mannens and Rik Van de Walle</i>	69
Semantic Vernacular System: an Observation-based, Community- powered, and Semantics-enabled Naming System for Organisms <i>Han Wang, Nathan Wilson, Kathryn Dunn and Deborah McGuinness</i>	73
Demo: Efficient Human Attention Detection in Museums based on Semantics and Complex Event Processing <i>Yongchun Xu, Nenad Stojanovic, Ljiljana Stojanovic and Tobias Schuchert</i>	77

Part II

Posters

Extracting Relevant Subgraphs from Graph Navigation <i>Valeria Fionda, Claudio Gutierrez and Giuseppe Pirró</i>	81
--	----

Jena-HBase: A Distributed, Scalable and Efficient RDF Triple Store	85
<i>Vaibhav Khadilkar, Murat Kantarcioglu, Bhavani Thuraisingham and Paolo Castagna</i>	
Applying Multidimensional Navigation and Explanation in Semantic Dataset Summarization	89
<i>James Michaelis, Deborah McGuinness, Cynthia Chang, Joanne Luciano and James Hendler</i>	
Building Large Scale Relation KB from Text	93
<i>Junfeng Pan, Haofen Wang and Yong Yu</i>	
Reasoning in RDFS is Inherently Serial, at least in the worst case	97
<i>Peter Patel-Schneider</i>	
INSTANS: High-Performance Event Processing with Standard RDF and SPARQL	101
<i>Mikko Rinne, Esko Nuutila and Seppo Törmä</i>	
RIO: Minimizing User Interaction in Ontology Debugging	105
<i>Patrick Rodler, Kostyantyn Shchekotykhin, Philipp Fleiss and Gerhard Friedrich</i>	
On direct debugging of aligned ontologies	109
<i>Kostyantyn Shchekotykhin, Patrick Rodler, Philipp Fleiss and Gerhard Friedrich</i>	
Queries, the Missing Link in Automatic Data Integration	113
<i>Aibo Tian, Juan F. Sequeda and Daniel Miranker</i>	
MeDetect: Domain Entity Annotation in Biomedical References Using Linked Open Data	117
<i>Li Tian, Weinan Zhang, Haofen Wang, Chenyang Wu, Yuan Ni, Feng Cao and Yong Yu</i>	
Towards Licenses Compatibility and Composition in the Web of Data . . .	121
<i>Serena Villata and Fabien Gandon</i>	

Mining Patterns from Clinical Trial Annotated Datasets by Exploiting the NCI Thesaurus

Joseph Benik¹, Guillermo Palma², Louiqa Raschid¹, Andreas Thor¹, and Maria-Esther Vidal²

¹ University of Maryland, College Park, USA
josephbenik@gmail.com, {louiqa,thor}@umiacs.umd.edu

² Universidad Simón Bolívar, Venezuela
{gpalma, mvidal}@ldc.usb.ve

Abstract. Annotations of clinical trials with controlled vocabularies of drugs and diseases, encode scientific knowledge that can be mined to discover relationships between scientific concepts. We present PAnG (Patterns in Annotation Graphs), a tool that relies on dense subgraphs, graph summarization and taxonomic distance metrics, computed using the NCI Thesaurus, to identify patterns.

1 Introduction

Linked Open Data has made available a diversity of collections and can facilitate scientists to mine semantically annotated datasets. These annotations represent scientific knowledge, for example, genes, proteins, drugs and diseases are annotated with controlled vocabulary terms (CV terms) from ontologies. Annotations describe properties of these concepts, and they are useful as a basis for focused literature review, and further, to plan a wet-lab experiment or a clinical trial. Annotation graphs as well as the ontologies are rich and complex, for example, the NCI Thesaurus version 12.05d has 93,788 terms. Thus, the challenge is to explore the potentially large number of annotations and to discover patterns. Automatic techniques and tools are therefore needed to support the scientist. These tools could range from making simple but valuable link predictions, e.g., predicting new gene functional annotation, to discovering complex patterns of annotation across multiple disease conditions and drug interventions.

We present PAnG (Patterns in Annotation Graphs), a tool that allows scientists to identify patterns in annotated graph datasets. PAnG is based on a complementary methodology of graph summarization (GS) and dense subgraphs (DSG) [3, 4]. DSG shows particular benefit in creating a promising subgraph, when the input graph is large and includes a diversity of ontology terms, or when the graph has sparse annotations. PAnG uses a taxonomic distance metric, d_{tax} [2] to compute distances between terms, e.g., in the NCI Thesaurus. Patterns are represented as graph summaries that consist of node partitions (super-nodes). Patterns can include super-edges between super-nodes as well as edges between individual nodes. Patterns provide a better visualization and understanding of the overall structure of the underlying graph.

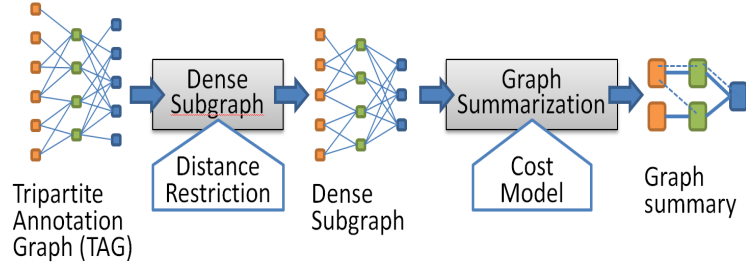


Fig. 1. The PAnG System Workflow.

Select	Description	Date Added	Visualize
S9b	Sirota et al - Intervention: Verapamil The clinical trials matching the intervention search: Verapamil	April 16, 2012	Visualize
S9d	Sirota et al - Intervention: Gefitinib The clinical trials matching the intervention search: Gefitinib	April 16, 2012	Visualize
S12	alemtuzumab The clinical trials matching the search: alemtuzumab	June 8, 2012	Visualize

1

2 Limit

Condition	Intervention
1 Specification of clinical trials.	
2 Search for conditions and interventions.	
3 Dense subgraph configuration (e.g., distance restrictions).	
4 Graph summarization configuration.	

3 ☒ **Dense Subgraph (DSG)**

Distance between Condition:

No distance restriction: ☐

Distance between Intervention:

No distance restriction: ☐

Triples: ☐ Required ☒ Not Require

4 ☒ **Graph Summarization**

Allow combining heterogenous nodes: ☐

Remove singletons: ☒

Fig. 2. PAnG's GUI for the LinkedCT dataset.

Further, the pattern captures semantic knowledge not only about individual nodes and their connections, but also about groups of related nodes. LinkedCT (LinkedCT.org) is a Linked Open Data dataset from the clinical trial site (ClinicalTrial.gov). Conditions represent diseases and are typically described using the NCI Thesaurus. Interventions include a (unique) name, a description and a type, e.g., a drug, device, procedure, etc. PAnG for LinkedCT.org is available at <http://pang.umiacs.umd.edu/iswc2012demo>.

2 The PAnG System

Figure 1 illustrates the overall workflow of PAnG. The input is a tripartite annotated graph G , and the output is a graph summary. Our workflow consists of two steps. The first step is optional and deals with the identification of dense subgraphs, i.e., highly connected subgraphs of G that are (almost) cliques. The

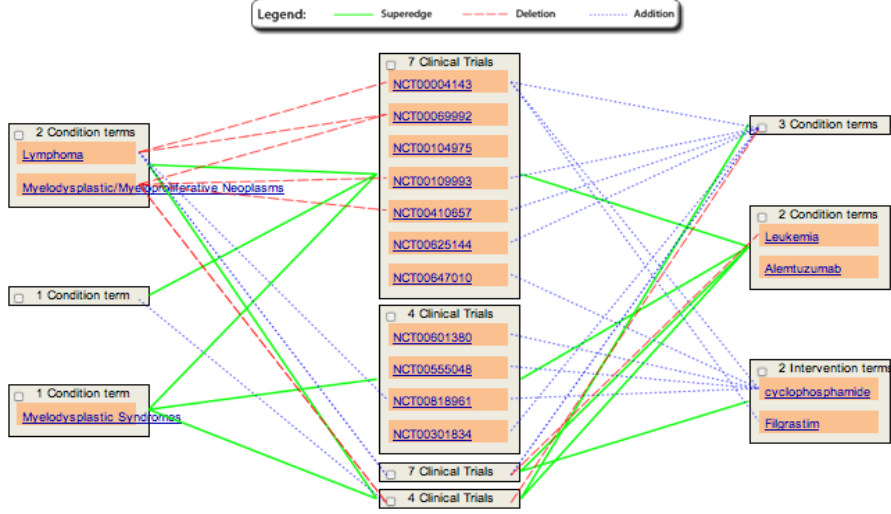


Fig. 3. Graph Summary for Alemtuzumab. PAnG configuration: S12 (alemtuzumab), DSG+GS, Triples Not Required, Distance between Condition: 0.5, Distance between Intervention: 0.5, Allowing combining heterogenous nodes, Remove singletons.

goal is to identify interesting regions of the graph by extracting a relevant sub-graph. Graph summarization transforms the graph into an equivalent compact graph representation. Graph summaries are made up of the following elements: (1) supernodes; (2) superedges; (3) deletion and addition edges (corrections). Figure 2 shows the PAnG interface for the LinkedCT annotation graph dataset. A scientist can search on conditions or interventions and create a subset of clinical trials. She can then choose the DSG option, with a threshold for d_{tax} for conditions or interventions or both. She can also skip the DSG option or choose a DSG without a distance restriction. She can also require that the DSG option favor the selection of triplets across the tri-partite graph, or favor the selection of independent doublets across the two bi-partite graphs. Figure 3 presents a possible summary graph. There are 10 supernodes; five supernodes cluster conditions, four supernodes cluster clinical trials and one supernode clusters the two interventions cyclophosphamide, and Filgrastim. A superedge is a solid edge and occurs between two supernodes. It represents that all nodes in both supernodes are fully connected, for example, the superedge between the supernode with the condition Myelodysplastic Syndromes and the supernode of 4 clinical trials. The summary reflects the basic pattern (structure) of the graph and is accompanied by a list of corrections, i.e., deletions and additions, that express differences between the graph and its simplified pattern. For example, a deletion edge to a condition that occurs within a supernode indicates that the specific condition *was not* studied in a particular clinical trial, whereas the conditions within the supernode, without deletion edges, were studied. In Figure 3, the condition Lymphoma (top left-hand supernode) was not studied in the clinical trial NCT00004143 \dots (top middle supernode).

3 Demonstration of Use Cases

As of September 2011, LinkedCT contains 106,308 trials, 2.7 million entities and over 25 million RDF triples. We demonstrate two use cases:

Single Drug: We commence with a single drug and create a dataset of all clinical trials associated with that drug, and all other conditions and interventions associated with these trials. We use the taxonomic distance metric, d_{tax} [1], and the NCI Thesaurus version 12.05d to compute pairwise (condition-condition) or (intervention-intervention) distance. Because, this is a poly-hierarchy, if there are alternate paths, the shortest path is chosen. Three drugs **Alemtuzumab**, **Getfinib**, and **Verapamil** are used to illustrate the effect of different configurations. For example, **Getfinib** treats certain types of cancers, e.g., breast or lung cancer. With no DSG, PAnG cannot discern any patterns. If DSG+GS are chosen, with no distance restriction and triples required, PAnG produces a very simple summary of a supernode of clinical trials for **Breast Cancer**, another supernode for **Lung Cancer**, with only **Getfinib** as the intervention, and one clinical trial covering both cancers. However, with DSG+GS, no distance restriction, no triples required, a different graph summary is generated. For example, with a distances threshold of 0.3 for both conditions and interventions, **Esophageal Cancer** is related to a supernode of **Radiation Therapy**, **conventional surgery** and **neoadjuvant therapy**; this suggests that this disease is related to these three procedures, in addition to treatments with **Getfinib**.

Drug Family: The NCI Thesaurus is used to explore drug families and their properties. Starting with **Alemtuzumab** as an exemplar, we retrieve the intersection of **Monoclonal antibodies** and **Antineoplastic agents**. This creates a dataset of 12 drugs: **Alemtuzumab**, **Bevacizumab**, **Brentuximab vedotin**, **Cetuximab**, **Catumaxomab**, **Edrecolomab**, **Gemtuzumab**, **Ipilimumab**, **Ofatumumab**, **Panitumumab**, **Rituximab**, and **Trastuzumab**. We use the pairwise annotation similarity based on the set of interventions associated with each drug to select interesting pairs of drugs[2] and then further analyze using PAnG.

References

1. J. Benik, C. Chang, L. Raschid, M.-E. Vidal, G. Palma, and A. Thor. Finding cross genome patterns in annotation graphs. In *DILS*, pages 21–36, 2012.
2. G. Palma, E. Haag, L. Raschid, A. Thor, and M.-E. Vidal. An Evaluation of Metrics to Compute Concept Similarity Based on Evidence from Ontologies. *Technical Report, University of Maryland UMIACS*, 2012.
3. B. Saha, A. Hoch, S. Khuller, L. Raschid, and X.-N. Zhang. Dense subgraphs with restrictions and applications to gene annotation graphs. In *Conference on Research on Computational Molecular Biology (RECOMB)*, 2010.
4. A. Thor, P. Anderson, L. Raschid, S. Navlakha, B. Saha, S. Khuller, and X.-N. Zhang. Link prediction for annotation graphs using graph summarization. In *Proc. of International Semantic Web Conference (ISWC)*, 2011.

The Linked Data Visualization Model

Josep Maria Brunetti¹, Sören Auer², Roberto García¹

¹ GRIHO, Universitat de Lleida, Jaume II, 69. 25001 Lleida, Spain
{josepmb Brunetti, rgarcia}@diei.udl.cat, <http://griho.udl.cat/>

² AKSW, Computer Science, University of Leipzig, Germany
auer@informatik.uni-leipzig.de, <http://aksw.org/>

Abstract. The potential of the semantic data available in the Web is enormous but in most cases it is very difficult for users to explore and use this data. Applying information visualization techniques to the Semantic Web helps users to easily explore large amounts of data and interact with them. We devise a formal Linked Data Visualization model (LDVM), which allows to *dynamically* connect data with visualizations.

1 Introduction

In the last years, the amount of semantic data available on the Web has increased dramatically, especially thanks to initiatives like Linked Open Data (LOD). The potential of this vast amount of data is enormous but in most cases it is very difficult *and* cumbersome for users to visualize, explore and use this data, especially for lay-users without experience with Semantic Web technologies.

Applying information visualization techniques to the Semantic Web helps users to explore large amounts of data and interact with them. Visualizations are useful for obtaining an overview of the datasets, their main types, properties and the relationships between them. Compared to prior information visualization strategies, we have a unique opportunity on the Data Web. The unified RDF data model being prevalent on the Data Web enables us to bind data to visualizations in an *unforeseen* and *dynamic* way. An information visualization technique requires certain data structures to be present. When we can derive and generate these data structures automatically from reused vocabularies or semantic representations, we are able to realize a largely automatic visualization workflow. This will enable users to explore datasets even if the publisher of the data does not provide any exploration or visualization means.

The *Linked Data Visualization Model* (LDVM) we propose allows to connect different datasets with different visualizations in a dynamic way. In order to achieve such flexibility and a high degree of automation the LDVM is based on a visualization workflow incorporating analytical extraction and visual abstraction steps. Each of the visualization workflow steps comprises a number of transformation operators, which can be defined in a declarative way. As a result, the LDVM balances between flexibility of visualization options and efficiency of implementation or configuration.

2 Linked Data Visualization Model

We use the *Data State Reference Model* (DSRM) proposed by Chi [1] as conceptual framework for our *Linked Data Visualization Model* (LDVM). While the DSRM describes the visualization process in a generic way, we instantiate and adopt this model with LDVM for the visualization of RDF and Linked Data. The names of the stages, transformations and operators have been adapted to the context of Linked Data and RDF. Figure 1 shows an overview of LDVM. It can be seen as a pipeline, which originates in one end with raw data and results in the other end with the visualization.

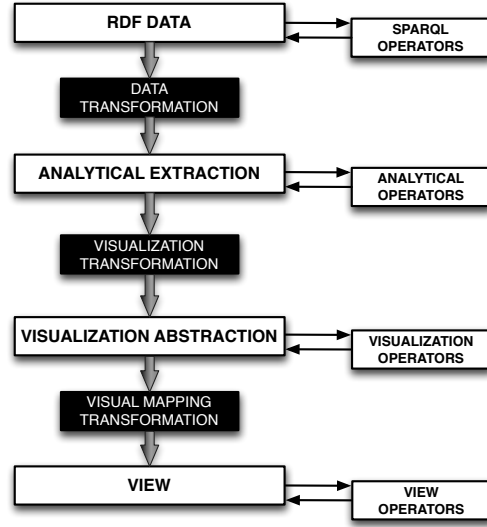


Fig. 1. High level overview of the Linked Data Visualization Model.

The LDVM pipeline is organized in four stages that data needs to pass through:

1. *RDF Data*: the raw data, which can be all kinds of information adhering to the RDF data model, e.g. instance data, taxonomies, vocabularies, ontologies, etc.
2. *Analytical extraction*: data extractions obtained from raw data, e.g. calculating aggregated values.
3. *Visual abstraction*: information that is visualizable on the screen using a visualization technique.
4. *View*: the result of the process presented to the user, e.g. plot, treemap, map, timeline, etc.

Data is propagated through the pipeline from one stage to another by applying three types of transformation operators:

1. *Data transformation*: transforms raw data values into analytical extractions declaratively (using SPARQL query templates).
2. *Visualization transformation*: takes analytical extractions and transforms them into a visualization abstraction. The goal of this transformation is to condense the data into a displayable size and create a suitable data structure for particular visualizations.
3. *Visual mapping transformation*: processes the visualization abstractions in order to obtain a visual representation.

As illustrated in Figure 2, our model allows to connect different RDF datasets and different data extractions with different visualization techniques. Not all datasets are compatible with all data extractions and each data extraction is only compatible with some visual configurations.

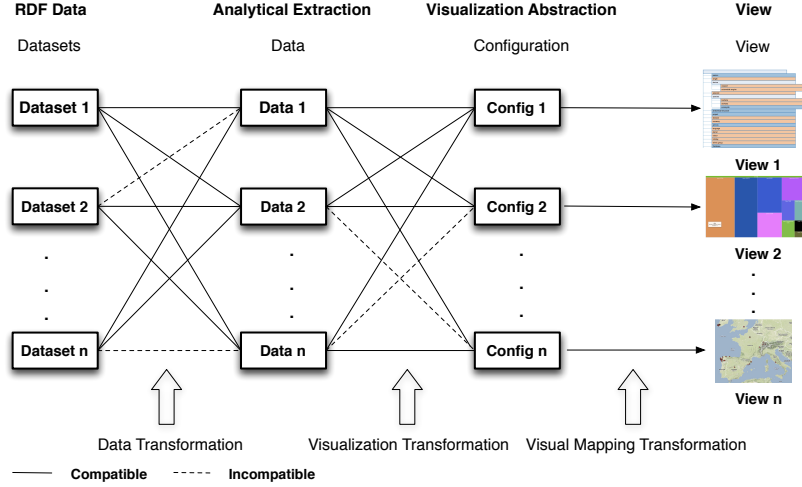


Fig. 2. Linked Data Visualization Model ecosystem, which allows to dynamically connect datasets with visualizations.

Each dataset offers different data structures to be extracted, e.g. class hierarchy, property hierarchy, geospatial data, etc. Each data extraction can be visualized with different configurations, which contain information such as the visualization technique to use, colors, etc. Then, a concrete visualization is generated depending on the data extraction and the visual configuration.

In summary, the model is divided in two main areas: data space and visual space. The RDF data stage, analytical extraction stage and data transformation

belong to the data space, while visual abstraction stage, view stage and visual mapping transformation belong to the visual space. These two main blocks are connected by a visualization transformation.

3 Demonstration

We have implemented a prototype called *LODVisualization*¹ that supports the Linked Data Visualization Model proposed. It allows to explore and interact with the Data Web through different visualizations. This way, our prototype serves not only as a proof-of-concept of our LDVM but also provides useful visualizations of RDF. These visualizations allow users to obtain an overview of RDF datasets and realize what the data is about: their main types, properties, etc.

LODVisualization is compatible with most of SPARQL endpoints as long as they support JSON and SPARQL 1.1. We have evaluated our implementation of the Linked Data Visualization Model with different datasets, data extractions and visualizations. The goal of our evaluation was to prove that the LDVM can be applied to different datasets providing different data visualizations in real time. All the visualization examples are available on the website and it easy to create new ones.

4 Related Work

Some of the existing tools available to explore and visualize Linked Data have been analyzed in [2]. However, only very few provide visualizations and they are focused on concrete data types or domains.

5 Conclusions

The Linked Data Visualization Model (LDVM) can be applied to rapidly create visualizations of RDF data. It allows to connect different datasets, different data extractions and different visualizations in a dynamic way. Applying this model, developers and designers can obtain a better understanding of the visualization process with data stages, transformations and operators. The LDVM offers user guidance on how to create visualizations for RDF data.

References

1. Ed H. Chi. A taxonomy of visualization techniques using the data state reference model. In *Proceedings of the IEEE Symposium on Information Visualization 2000*, INFOVIS '00, pages 69–, Washington, DC, USA, 2000. IEEE Computer Society.
2. Aba-Sah Dadzie and Matthew Rowe. Approaches to visualising linked data: A survey. *Semantic Web*, 2(2):89–124, 2011.

¹ <http://lodvisualization.appspot.com/>

QAKiS: an Open Domain QA System based on Relational Patterns

Elena Cabrio¹, Julien Cojan^{1*}, Alessio Palmero Aprosio^{2,3},
Bernardo Magnini², Alberto Lavelli², and Fabien Gandon¹

¹ INRIA, 2004 Route des Lucioles, Sophia Antipolis, France
{elena.cabrio, julien.cojan, fabien.gandon}@inria.fr

² FBK-Irst, Via Sommarive 18, Povo-Trento, Italy
{aprosio, magnini, lavelli}@fbk.eu

³ Università degli Studi di Milano, Via Comelico 39/41, Milano, Italy

Abstract. We present QAKiS, a system for open domain Question Answering over linked data. It addresses the problem of question interpretation as a relation-based match, where fragments of the question are matched to binary relations of the triple store, using relational textual patterns automatically collected. For the demo, the relational patterns are automatically extracted from Wikipedia, while DBpedia is the RDF data set to be queried using a natural language interface.

1 Introduction

To enhance users interactions with the web of data, query interfaces providing a flexible mapping between natural language expressions, and concepts and relations in structured knowledge bases are becoming particularly relevant. This demonstration presents QAKiS (Question Answering wiKiframework-based System), that allows end users to submit a query to an RDF triple store in English and obtain the answer in the same language, hiding the complexity of the non intuitive formal query languages involved in the resolution process. At the same time, the expressiveness of these standards is exploited to scale to the huge amounts of available semantic data. In its current implementation, QAKiS addresses the task of QA over structured Knowledge Bases (KBs) (e.g. DBpedia) where the relevant information is expressed also in unstructured form (e.g. Wikipedia pages). Its major novelty is to implement a relation-based match for question interpretation, to convert the user question into a query language (e.g. SPARQL). Most of the current approaches (for an overview, see [2]) base this conversion on some form of flexible matching between words of the question and concepts and relations of a triple store, disregarding the relevant context around a word, without which the match might be wrong. QAKiS tries instead first to establish a matching between fragments of the question and relational textual patterns automatically collected from Wikipedia. The underlying intuition is that a relation-based matching would provide more precision with respect to matching on single tokens, as done by current QA systems.

* Acknowledges the French Minister of Culture for supporting the DBpedia Fr project.

2 QAKiS system description

QAKiS demo⁴ (Fig. 1) is based on Wikipedia for patterns extraction. DBpedia is the RDF data set to be queried using a natural language interface.

The screenshot shows the QAKiS demo interface. At the top, there is a search bar with the text "Which river does the Brooklyn Bridge cross?". To the right of the search bar are two buttons: "Get answers!" and "Clear". Below the search bar, the system's response is displayed. It starts with "Your asked: Which river does the Brooklyn Bridge cross?". Then, under "Pattern matching:", it shows the typed question "[River] does [Bridge] cross ?", the best match pattern "crosses [D:Bridge] international border cross [R:River]", and the score "10.1366333333333". Next, under "The query generated is:", it shows a SPARQL query. Finally, there is a table with a green header row labeled "result" and one data row labeled "East River".

QA
KiS
Question Answering
wikiFramework-based
System

Which river does the Brooklyn Bridge cross?

Your asked: Which river does the Brooklyn Bridge cross?

Pattern matching:
typed question: [River] does [Bridge] cross ?
had a best match with pattern: crosses [D:Bridge] international border cross [R:River]
with score 10.1366333333333

The query generated is:
select distinct * where { <http://dbpedia.org/resource/Brooklyn_Bridge> <http://dbpedia.org/ontology/crosses> ?v . ?v rdf:type <http://dbpedia.org/ontology/River> . OPTIONAL { ?v <http://www.w3.org/2000/01/rdf-schema#label> ?l filter (lang(?l)='en')} } limit 20

#	result
1	East River

Fig. 1. QAKiS demo interface. The user can either write a question (or select among a list of examples) and click on *Get Answers!*. QAKiS outputs: *i*) the user question (the recognized Named Entity (NE) is linked to its DBpedia page), *ii*) the generated typed question (see Section 2.1), *iii*) the pattern matched, *iv*) the SPARQL query sent to the DBpedia SPARQL endpoint, and *v*) the answer (below the green rectangle *results*).

QAKiS makes use of relational patterns (automatically extracted from Wikipedia and collected in the WikiFramework repository [3]), that capture different ways to express a certain relation in a given language. For instance, the relation **crosses**(**Bridge**,**River**) can be expressed in English, among the others, by the following relational patterns: [**Bridge** crosses the **River**] and [**Bridge** spans over the **River**]. Assuming that there is a high probability that the information in the Infobox is also expressed in the same Wikipedia page, the WikiFramework establishes a 4-step methodology to collect relational patterns in several languages for the DBpedia ontology relations (similarly to [1],[4]): *i*) a DBpedia relation is mapped with all the Wikipedia pages in which such relation is reported in the Infobox; *ii*) in such pages we collect all the sentences containing both the domain and the range of the relation; *iii*) all sentences for a given relation are extracted and the domain and range are replaced by the corresponding DBpedia ontology classes; *iv*) the patterns for each relation are clustered according to the lemmas between the domain and the range, and sorted according to their frequency.

2.1 System architecture

QAKiS is composed of two main modules (Fig. 2): *i*) the **query generator** takes the user question as input, generates the typed questions, and then generates the SPARQL queries from the retrieved patterns; *ii*) the **pattern matcher** takes as input a typed question, and retrieves the patterns (among those stored in the pattern repository) matching it with the highest similarity.

⁴ Available at <http://dbpedia.inria.fr/qakis/>

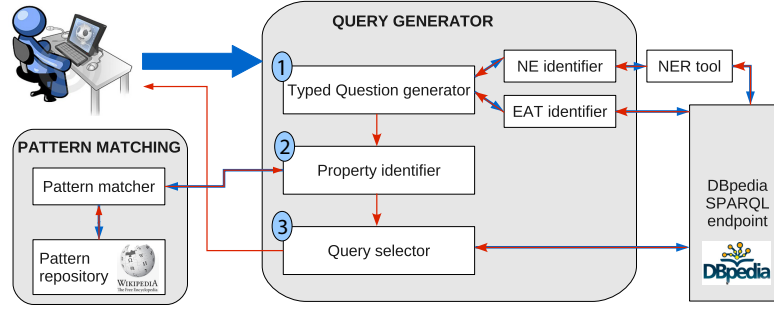


Fig. 2. QAKiS workflow

The current version of QAKiS targets questions containing a NE related to the answer through one property of the ontology, as *Which river does the Brooklyn Bridge cross?*. Each question matches a single pattern (i.e. one relation).

Expected Answer Type (EAT) and NE identification. Before running the *pattern matcher* component, we identify the target of the question with a NER tool. We apply the Stanford Core NLP NE Recognizer together with a set of strategies based on the comparison with the labels of the instances in the DBpedia ontology. We plan to test the use of other NER tools in the future. At the same time, simple heuristics are applied to infer the EAT from the question keyword, e.g. if the question starts with “When”, the EAT is [Date] or [Time], with “Who”, the EAT is [Person] or [Organisation] and so on.

Typed questions generation. We generate a *typed question* by replacing the question keywords (e.g. who, where) and the NE by the types and supertypes. Given the question “Who is the husband of Amanda Palmer?” 9 typed questions are generated, since *i)* both [Person] or [Organisation] (subclasses of [owl:Thing]) are considered as EAT, and *ii)* [MusicalArtist], [Artist] and [owl:Thing] are the types of the NE *Amanda Palmer*.

WikiFramework pattern matching. The typed questions are lemmatized, tokenized, and stopwords are removed. A Word Overlap algorithm is then applied to match such typed questions with the patterns for each relation. A similarity score is provided for each match: the highest represents the most likely relation.

Query selector. A set of patterns (max. 5) is retrieved by the pattern matcher component for each typed question, and sorted by decreasing matching score. For each of them, one or two SPARQL queries are generated, either *i)* `select ?s where{?s <property> <NE>}`, *ii)* `select ?s where{<NE> <property> ?s}` or *iii)* both, according to the compatibility between their types and the property domain and range. Such queries are then sent to the SPARQL endpoint for answer retrieval. If the query produces no results, we try with the next pattern, until a satisfactory query is found or no more patterns are retrieved.

2.2 Experimental evaluation

Table 1 reports QAKiS’s results on the QALD-2 data sets⁵ (DBpedia track). For the demo, we focused on code optimization reducing QAKiS average processing time per question from 15 to 2 sec., w.r.t. the version used for the challenge.

	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i># answered</i>	<i># right answ.</i>	<i># partially right</i>
train	0.476	0.479	0.477	40/100	17/40	4/40
test	0.39	0.37	0.38	35/100	11/35	4/35

Table 1. QAKiS performances on DBpedia data sets (participation to QALD-2)

Most of QAKiS’ mistakes concern wrong relation assignment (i.e. wrong pattern matching). We plan to replace the Word Overlap algorithm with approaches considering the syntactic structure of the question. Another issue concerns questions ambiguity, i.e. the same surface forms can in fact refer to different relations in the DBpedia ontology. We plan to cluster relations with several patterns in common, to allow QAKiS to search among all the relations in the cluster. The partially correct answers concern questions involving more than one relation: the actual version of the algorithm detects indeed only one of them. We plan to target questions as *Give me all people that were born in Vienna and died in Berlin* in a short time, since the two relations are easily separable. On the contrary, we need more complex strategies to answer questions with nested relations.

3 Future perspectives

We are currently considering to publish the WikiFramework relational patterns as RDF triples, organized according to a newly defined RDF vocabulary (similarly to [1]). We are also planning improvements on: *i)* the WikiFramework pattern extraction algorithm, following [4]; *ii)* the question-pattern matching algorithm; *iii)* the system coverage, addressing boolean and n-relation questions. We are also exploring QAKiS applicability in real application scenarios.

References

1. Gerber, D., Ngonga Ngomo, A.C. (2011), Bootstrapping the Linked Data Web, in 1st Workshop on Web Scale Knowledge Extraction @ ISWC 2011.
2. Lopez V., Uren V., Sabou, M., Motta, E. (2011), Is Question Answering fit for the Semantic Web?: a Survey, in Semantic Web journal, vol. 2(2), pp. 125-155.
3. Mahendra, R., Wanzare, L., Bernardi, R., Lavelli, A., Magnini, B. (2011), Acquiring Relational Patterns from Wikipedia: A Case Study, in Proc. of LTC2011.
4. Wu F., Weld, D.S. (2010), Open information extraction using Wikipedia, in Proc. of ACL2010, pp. 118-127.

⁵ <http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/>

DiscOU: A Flexible Discovery Engine for Open Educational Resources Using Semantic Indexing and Relationship Summaries

Mathieu d’Aquin, Carlo Allocca and Trevor Collins

Knowledge Media Institute, The Open University, Milton Keynes, UK
{m.daquin, c.allocca, t.d.collins}@open.ac.uk

Abstract. We demonstrate the DiscOU engine implementing a resource discovery approach where the textual components of open educational resources are automatically annotated with relevant entities (using a named entity recognition system), so that these rich annotations can be searched by similarity, based on existing resources of interest.

1 Introduction / Motivation

There is a growing base of open educational content being made available online. At the Open University, this currently includes 650 units of course material on OpenLearn and 3,800 audio and video podcasts¹. With such content available, discoverability of educational resources becomes a major challenge. The exposure of the metadata for such resources as linked data (see data.open.ac.uk and [1]) is expected to make these resources more directly addressable, together with their general description and the subjects they are covering (see e.g. [2]). Accordingly, linked data is increasingly being adopted in open and distance learning scenarios where discoverability is a main requirement (see [3]). However, relying purely on metadata requires either to stay at a high level of description of the content of resources (through the general topics being covered) or to richly annotate these resources with all the dimensions relevant to their content. Another common approach is therefore to search by similarity based on existing resources of interest (i.e. finding things that are “more like this”). This is however generally limited to the comparison of the textual components of the resources, with obvious limitations, as similarity between texts does not necessarily reflect a useful relationship between resources in a discovery scenario.

Here, we demonstrate an engine implementing a hybrid approach, where the textual components of open educational resources are automatically annotated with relevant entities (using a named entity recognition system), so that these rich annotations can be searched by similarity. This allows us to discover resources based on relationships that are not necessarily explicitly described in their metadata, and to characterise semantically these relationships based on shared entities. This also provides us with a more flexible workflow, compared to typical recommendation engines, where the user can act upon the search for resources, through customising the semantic annotations realised prior to similarity search. We demonstrate a prototype application of the developed services to discover open educational content from the Open University, based on the content of programmes broadcasted by the BBC.

¹ see <http://podcast.open.ac.uk> and <http://openlearn.open.ac.uk>

2 The DiscOU Approach

Figure 1 summarises the architecture of the DiscOU system, which describes the workflow implemented in four RESTful services² for 1- extracting semantic entities from an online resource, 2- indexing these entities, 3- searching by similarity in the index and 4- summarising the relationships between resources.

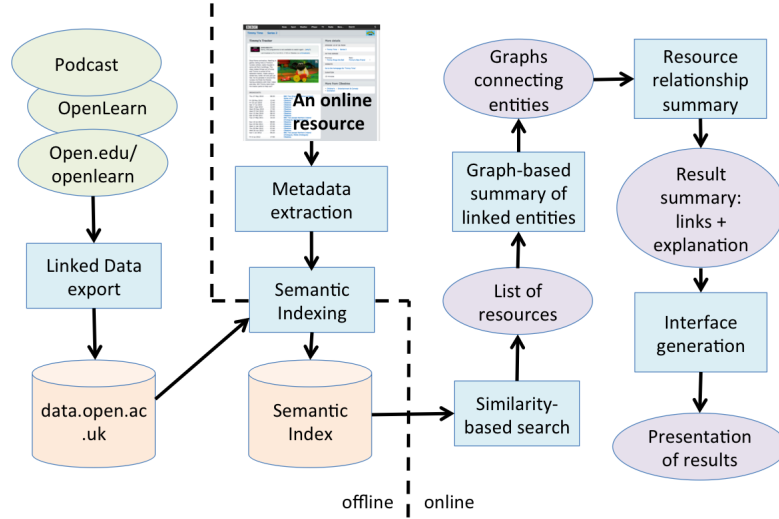


Fig. 1. Overview of the architecture of the DiscOU system.

Semantically Indexing Online Resources. As described in the previous section, the main idea behind DiscOU is to take a hybrid approach where resources can be searched “by similarity” with another existing resource, but where the comparison of resources is based on rich semantic annotations. To generate such rich semantic annotations, we make use of a named entity recognition system, namely DBpedia Spotlight [4]. Textual components are first extracted from the resources to be indexed, based on their description on data.open.ac.uk (using the metadata directly for the title and abstract, and links to the textual content, as online documents for OpenLearn units and PDFs of transcripts for podcasts). The online service provided by DBpedia Spotlight is then used to obtain a list of DBpedia entities with confidence/relevance scores for each of these components.

To index these semantic descriptions, we use the Lucene open source search engine library³. Lucene is however designed to index documents and texts and is based on term-occurrence measures for searching and ranking results (i.e., TF.IDF). The indexing of semantic annotations is therefore realised in such a way that these mechanisms can be used to obtain relevant results when searching on the basis of semantic entities rather than of text. This is achieved simply by

² see <http://discou.info>

³ <http://lucene.apache.org/core/>

transforming the relevance score provided by DBpedia Spotlight into a number of occurrences for the entity, therefore repeating the mention of an entity in the index of a given resource depending on its relevance for the resource. In this way, when searching based on semantic entities, Lucene should return in priority resources for which these entities are highly relevant.

Searching by Similarity. Lucene provides the base technique to search by similarity through a mechanism called “MoreLikeThis”. This mechanism takes as input an indexed document and generates a query that is expected to return other resources having similar indexes. We apply this mechanism through first indexing the external resource used as starting point for the discovery process (in the next section, we use BBC programme webpages) using the same process as described above. Because of the way the index is constructed, resources are returned that share a large part of their semantic annotation.

Summarising Relationships Between Resources. One major advantage of our approach is that the similarity relationship between resources being discovered and the original ‘query resource’ is characterised by the semantic entities shared in their content. Depending on the richness of the considered content however, such lists of shared entities can be too large to be useful summaries of this relationship. To tackle this issue, we developed a mechanism to summarise lists of DPpedia entities. It uses DBpedia links between entities in a list (using a local index of DBpedia, optimised for this specific task) to generate a set of connected graphs. Each of these graphs is expected to represent one major topic of the resources being considered. We therefore select the one which contains the most entities with the highest relevance and, within this graph, the entity that appears to be the most connected and the most relevant.

3 Demonstrator: Finding Open University Content Based on BBC programmes

We implemented a demonstrator using the above mechanisms for a scenario in which a user, having found a BBC programme interesting, wants to obtain links to open educational resources to learn more about the topics covered by the programme (see Figure 2). The interface of the demonstrator is implemented in Javascript, using a bookmarklet to trigger the search. In other terms, being on a BBC programme page (in the example Figure 2, “The Secret Life of Chaos”⁴), the user can click on the DiscOU bookmark to make appear the results of searching for similar resources in Open University content.

This is realised by extracting textual content from the BBC programme page (out of its RDF description on the BBC website), running the semantic indexing service on this content and searching by similarity. The results show the titles and descriptions of the retrieved resources (obtained using the SPARQL endpoint of data.open.ac.uk) as well as the summary of the relationship between each resource and the BBC programme (here, mostly that they are about Chaos Theory and Economy). One interesting aspect of this demonstrator is that the

⁴ <http://www.bbc.co.uk/programmes/b00pv1c3>

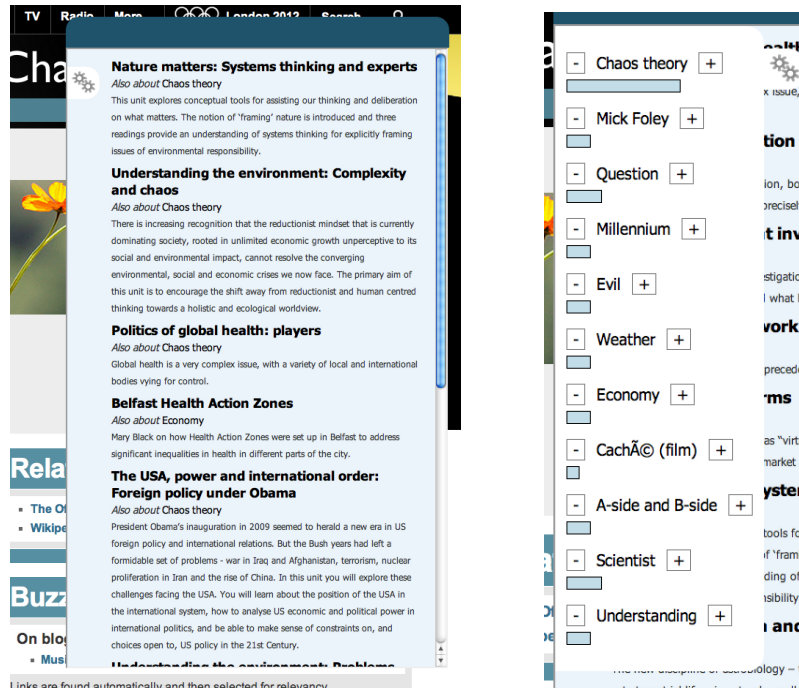


Fig. 2. Default results obtained with the BBC programme “The Secret Life of Chaos” (left) and interface to customise the semantic annotations for this programme (right).

user can customise the ‘query’ by changing the weights of the entities extracted as semantic annotations for the BBC programme (see right part of Figure 2). Once the weights are customised, the search is triggered again, showing results that are related to the personalised semantic annotations of the BBC programme.

While the results obtained are not always relevant, the fact that some level of explanation is provided together with the ability to refine the automatically generated ‘query’ makes the issue of incorrect results less critical. It is worth mentioning in particular that only a very small part of the system is specific to BBC programmes. The engine is used by the demonstrator as a set of RESTful services, with its functionalities being highly reusable in other scenarios.

References

1. d’Aquin, M.: Putting linked data to use in a large higher-education organisation. In: Interacting with Linked Data workshop. (2012)
2. Heath, T., Singer, R., Shabir, N., Clarke, C., Leavesley, J.: Assembling and applying an education graph based on learning resources in universities. In: Linked Learning (LILE) Workshop. (2012)
3. d’Aquin, M.: Linked data for open and distance learning. Commonwealth of Learning report (2012)
4. Mendes, P., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia spotlight: shedding light on the web of documents. In: International Conference on Semantic Systems. (2011)

ourSpaces - A Semantic Virtual Research Environment

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

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

Abstract. In this demo we present *ourSpaces*, a semantic virtual research environment designed to support inter-disciplinary research teams. The system utilizes technologies such as OWL, RDF and a rule-based reasoner to support the management of provenance information, social networks, online communication and policy enforcement within the VRE.

Keywords: provenance, virtual research environment, eResearch

1 Introduction

In recent years, scientific research has become increasingly interdisciplinary in nature and a range of information and communication technologies have been adopted by researchers to support collaboration, and to facilitate transfer of ideas, knowledge and resources. Web-based virtual research environments (VREs)¹ have been proposed as one way to help researchers in all disciplines to manage the increasingly complex range of tasks involved in carrying out research. Semantic web technologies are seen as crucial in this context in order to provide a common framework to allow intelligent applications and services to make use of information about data resources and other (research) objects held in a VRE.

The *ourSpaces* virtual research environment (described in Edwards et al. [1]) has been developed by the PolicyGrid² Digital Social Research project to provide a collaboration space for interdisciplinary academic research communities using state-of-the-art Semantic Web technologies. Groups using *ourSpaces* work in socio-environmental and health-related domains and there are currently around 183 registered users. A screenshot of the *ourSpaces* web interface is presented in Figure 1.

Provenance in *ourSpaces* is crucial in order to support transparency and accountability of the research process by documenting the derivation history of research artefacts. The system utilizes technologies such as OWL³, RDF⁴ and a

¹ <http://www.jisc.ac.uk/publications/reports/2010/vrelandscapestudy.aspx>

² This work is supported by the UK Economic & Social Research Council (ESRC) under the Digital Social Research programme; award RES-149-25-1075.

³ <http://www.w3.org/TR/owl-ref/>

⁴ <http://www.w3.org/RDF/>

SPIN-based⁵ reasoner to support the following system functionalities: representing and tracking the provenance of digital artefacts and processes; capturing the provenance associated with a user's social network; managing policies associated with use and re-use of data, security and privacy; controlling the behaviour of services within the application using policy-based reasoning; visualising provenance information using different modalities (natural language or graphical).



Fig. 1. A screenshot of the *ourSpaces* VRE showing a user's home space, an open upload form and the graphical provenance visualiser.

During the demonstration, we will illustrate how semantic web technologies have been deployed to support key research activities within the system such as uploading and annotating research artefacts; managing project information e.g. membership, sub-projects, data, notifications; writing comments and blogs; and visualising information about research artefacts and their provenance.

⁵ <http://spinrdf.org/>

2 A Semantic Framework for Provenance

At the heart of *ourSpaces* is an ontological framework (see Figure 2) describing different aspects of the provenance of the research process. In order to support basic provenance we use a Web Ontology Language (OWL) representation of the Open Provenance Model [2]. This ontology defines entities such as *Artefact*, *Agent* and *Process* and causal relationships between them (e.g. *wasGeneratedBy*, *used* and *wasControlledBy*). OPM is a generic solution and as a result, our framework supports additional domain-specific provenance ontologies that are created by extending the concepts defined in the OPM ontology with domain-specific classes. Using these ontologies it is possible, for example, to describe a physical research activity (e.g. an interview) as an `opm:Process`, and how such an activity causes an `opm:Artifact` to be generated (e.g. interview notes).

For research groups utilising *ourSpaces*, it is important to situate research artefacts and processes alongside people and their associated organisational structures. The current OPM specification supports limited information about a person (agent) controlling a process. Friend-of-a-Friend⁶ (FOAF) is an established RDF vocabulary for describing people and their social networks and we have opted to utilise this within our framework; a `foaf:Profile` is thus a subclass of `opm:Agent`.

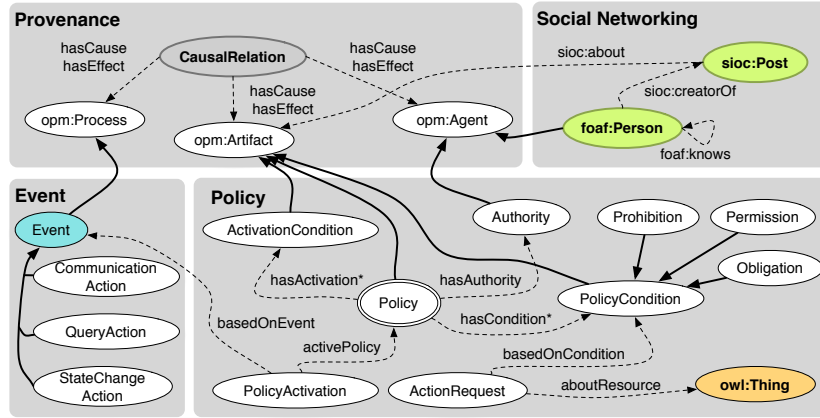


Fig. 2. *ourSpaces* ontological framework.

In an environment like *ourSpaces*, online communication is often used to comment about research artefacts or to discuss research issues. Documenting such interactions within in the VRE is a crucial requirement for achieving a full and transparent provenance representation. The SIOC⁷ (Semantically-Interlinked On-

⁶ <http://www.foaf-project.org/>

⁷ <http://sioc-project.org/>

line Communities) ontology is designed to describe aspects of online communication by providing a model to express user-generated content such as posting a message in a blog or posting a comment. We have also integrated this vocabulary within our provenance framework, e.g. a `sioc:post` generated by a `foaf:user` can be associated with an `opm:Artifact`, `opm:Process` or `opm:Agent`.

Within the system we have developed a service enabling users to visualise short textual descriptions of the provenance of resources. This service translates RDF statements into English sentences using a Natural Language Generation algorithm based on the approach described by Bouttaz et al. [3].

Within the environment, there is also a need to manage users and their behaviours so that they comply with certain policies. For example, a user may impose certain access constraints on digital artefacts that he/she owns, e.g. an artefact may only be accessible to people within that user's social network. We have extended our ontological framework to define such policies as a combination of conditions such as obligations, prohibitions or permissions [4]. We make use of the SPIN ontology ⁸ to support the use of the SPARQL query language to specify rules and logical constraints necessary to reason about policies. Policies in *ourSpaces* are activated based on events taking place in the environment, e.g. download/upload artefact, add/remove metadata, etc. When an activity is detected, an event manager initiates a policy reasoning task. Using this approach in *ourSpaces* we were able to implement a number of policies for use by the project teams using the system.

References

1. Edwards, P., Pignotti, E., Eckhardt, A., Ponnampereuma, K., Chris, Bouttaz, T.: *ourspaces - design and deployment of a semantic virtual research environment*. In: International Semantic Web Conference (ISWC 2012) - Semantic Web in Use Track, Springer-Verlag (2012) to appear.
2. Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y., Groth, P., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., Plale, B., Simmhan, Y., Stephan, E., den Bussche, J.V.: *The open provenance model core specification (v1.1)*. Future Generation Computer Systems (July 2010)
3. Bouttaz, T., Pignotti, E., Mellish, C., Edwards, P.: A policy-based approach to context dependent natural language generation. In: Proceedings of the 13th European Workshop on Natural Language Generation, Nancy, France, Association for Computational Linguistics (September 2011) 151–157
4. Pignotti, E., Edwards, P.: Using web services and policies within a social platform to support collaborative research. In: Working Notes of AAAI 2012 Stanford Spring Symposium on Intelligent Web Services Meet Social Computing. (March 2012)

⁸ <http://spinrdf.org/spin>

SPARQLoid - a Querying System using Own Ontology and Ontology Mappings with Reliability

Takahisa Fujino¹ and Naoki Fukuta²

¹ Graduate School of Informatics, Shizuoka University

² Faculty of Informatics, Shizuoka University
{gs12033@s, fukuta@cs}.inf.shizuoka.ac.jp

Abstract. The heterogeneity of ontologies on the web of data is a crucial problem. This paper presents an application called SPARQLoid that uses a query rewriting method to enable users to query any SPARQL endpoint with the user's own ontology, even when their ontology mappings are not reliable enough. Often ontology matching methods produce mappings with their reliability degree. Based on the given reliability degrees of those mappings, SPARQLoid allows users to query data in the target SPARQL endpoints by using their own (or a specified certain) ontology under a control of sorting order based on their mapping reliability. In this paper, we show a brief demonstration and a comparison to related work.

Keywords: SPARQL, heterogeneous ontology, query translation.

1 Introduction

In this paper, we propose SPARQLoid³, which can add functionality to SPARQL queries for utilizing the reliability degree in mappings. SPARQLoid enables users to make a query without using the ontologies prepared for the target endpoints; it can also control the resulting outputs based on the reliability degrees in the mappings.

Our developed system uses a query rewriting approach⁴ to allow translated SPARQL queries to be reused in user applications. In SPARQLoid, users can use special primitives to specify the criteria to control the results (e.g., controlling sorting orders, placing a threshold to remove unreliable results, etc., see Table 1) based on the mapping reliability to the concepts used in the query. To shorten the generated query strings and to avoid storing large amounts of mapping data in the user applications, the system can generate a query that refers to an external endpoint that stores necessary ontology mappings.

Figure 1 shows our SPARQLoid architecture. We implemented SPARQLoid using Jena 2.6.4⁵ and ARQ 2.8.8 in Jena. The translator engine can use ontology alignment data that are produced by a standard Ontology Alignment API 4.4⁶.

³ A demo video is available at <http://whitebear.cs.inf.shizuoka.ac.jp/sparqloid-demo/>

⁴ A preliminary idea about this approach is presented in [1].

⁵ http://jena.apache.org/about_jena/about.html

⁶ <http://alignapi.gforge.inria.fr/>

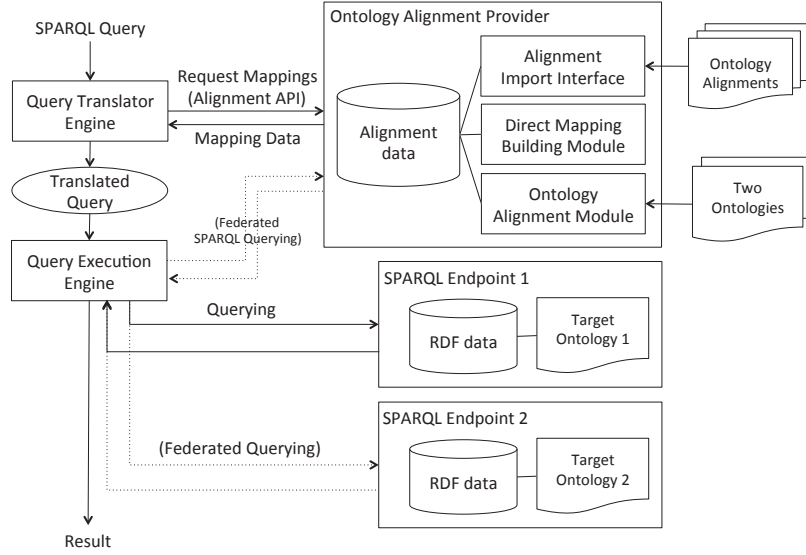


Fig. 1. SPARQLoid Architecture

2 Example

Figure 2 shows a working example of SPARQLoid. First, the user chooses ontology(s) that will be used in the query (Fig. 2(1)). Then, the user enters the endpoint(s) to retrieve data by the query (Fig. 2(2)). The user can see the ontologies that have been already registered in SPARQLoid and confirm whether appropriate mappings exist (Fig. 2(3)) or generate new mappings by APIs (Fig. 2(4)). In the next tab (Figure 2(5)), the user writes a query based on the ontologies chosen in the first step. The query translator engine translates the entered query into a translated query that is written in ordinary SPARQL syntax (Fig. 2(6)). The translated query can be executed on the system (Fig. 2(7)). Furthermore, the user can get programming-friendly code (e.g., Java, PHP, etc.) of the query (Fig. 2(8)).

3 Related Work and Comparison

Mosto[2,3] is a tool to perform data translation using automatically generated SPARQL executable mappings. The SPARQL-RW[4] framework provides transparent query access over mapped RDF datasets by using Expressive and Declarative Ontology Alignment Language (EDOAL)⁷. The R2R framework [5] au-

⁷ <http://alignapi.gforge.inria.fr/edoal.html>

Expressions in RANKING	Ranking Criterion
A+B+C	sim=a+b+c
max(A)+max(B)+max(C)	sim=max(a)+max(b)+max(c), max(a) means maximum reliability degree when multiple mappings are available for a.
A=0.6,B=0.4,C=0.3	sim = a*0.6+b*0.4+c*0.3
A>B>C	a is used as the primary key for sorting, b is used as the sub-key, and c is used as the sub-subkey
Expressions in THRESHOLD	Filtering Criterion
A=0.3,B=0.2,C=0.4	Solutions using a > 0.3, b > 0.2, c > 0.4
max(A)=0.5, max(B)=0.4,max(C)=0.6	Solutions with max(a) > 0.5, max(b) > 0.4, max(c) > 0.6
A:3,B:3,C:3	At most top 3 relevant mappings will be used for A, B, and C

Table 1. Example Syntax of RANKING and THRESHOLD clauses

	SPARQLoid	Mosto[2,3]	SPARQL-RW[4]	R2R[5]
ability to generate SPARQL-compatible queries	✓		✓	
use of mapping reliability	✓			✓
control syntax for mapping reliability	✓			
ability to translate data		✓		✓
ability to integrate data		✓		✓
ability to generate mappings	*	✓		
support for complex mappings		✓	✓	✓
ability to publish mappings	**			✓
support for interlinking mappings				✓
application embeddability	✓		✓	✓

✓implemented, * partially implemented, ** under development

Table 2. Comparison to Related Work

tomatically discovers and publishes mappings among Linked Data sources for better integration. Table 2 briefly compares our work and the above works.

References

1. Fujino, T. and Fukuta, N.: A SPARQL Query Rewriting Approach on Heterogeneous Ontologies with Mapping Reliability, *The 3rd IIAI International Conference on e-Services and Knowledge Management (IIAI ESKM 2012)*, (2012). (to appear)
2. Rivero, C., Hernandez,I., Ruiz,D., and Corchuelo, R.: Mosto: Generating SPARQL Executable Mappings Between Ontologies, *The 30th International Conference on Conceptual Modeling Demos and Posters*, (2011).
3. Rivero, C., Hernandez,I., Ruiz,D., and Corchuelo, R.: Generating SPARQL Executable Mappings to Integrate Ontologies, *Proc. of the 30th International Conference on Conceptual Modeling*, pp. 118–131, (2011).
4. Makris, K., Bikakis, N., Gioldasis, N., and Christodoulakis, S.: SPARQL-RW: Transparent Query Access over Mapped RDF Data Sources, *The 15th International Conference on Extending Database Technology (EDBT2012)*, (2012).
5. Bizer, C. and Schultz, A.: The R2R Framework: Publishing and Discovering Mappings on the Web, *The 1st International Workshop on Consuming Linked Data (COLID 2010)*, (2010).

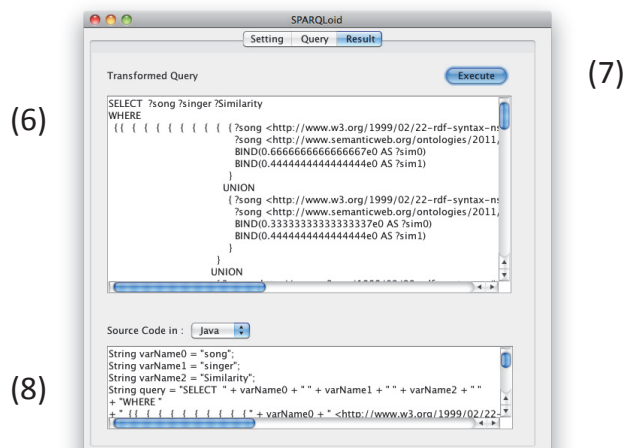
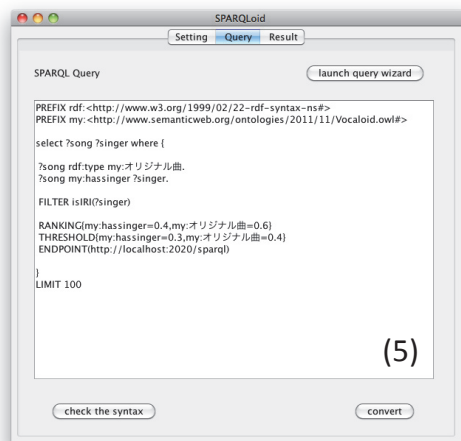
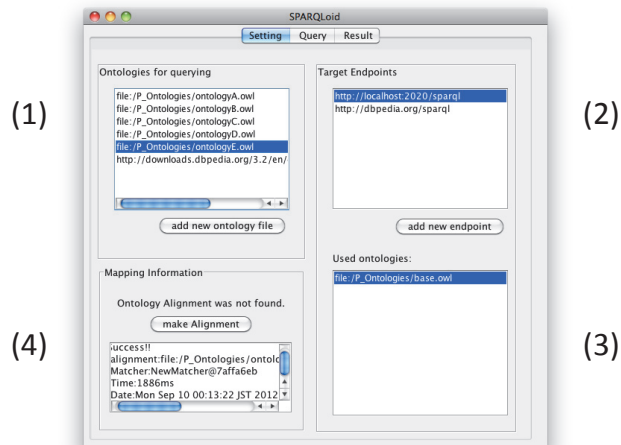


Fig. 2. System Overview

Simplifying MIREOT: a MIREOT Protégé Plugin

Josh Hanna^{1,2}, Cheng Chen², W. Alex Crow², Roger Hall², Jie Liu², Tejaswini Pendurthi², Trent Schmidt², Steven F. Jennings², Mathias Brochhausen¹, William Hogan¹

¹ University of Arkansas for Medical Sciences
jhanna@uams.edu

² University of Arkansas at Little Rock

Abstract. The Web Ontology Language (OWL) [1] is a commonly used standard for creating ontology artifacts. However, its capabilities for reusing existing OWL artifacts in the creation of new artifacts is limited to the import of whole ontologies, even when only a small handful of classes, object properties, and so on (which we refer to generically as “OWL components”) are relevant. This situation can result in extremely large and unwieldy, or even broken, ontologies.

To address this problem while still promoting ontology reuse, the OBI Consortium [2] has elucidated the Minimum Information to Reference an External Ontology Term (MIREOT) [3]. We provide a suite of plugins to the Protégé [4] editor that greatly simplifies the use of MIREOT principles during ontology creation and editing.

1 Introduction

Dealing with the ever-increasing amount of data being generated has been an extremely difficult problem. Researchers have spent a lot of time categorizing this information and the semantic web is one response to this problem. Ontologies, in particular, are one of the technologies that the semantic web uses to help organize information. To make ontologies the basis of semantic integration across multiple data sources, we need to facilitate re-use of ontologies – and representational units from those ontologies – across communities.

However, the semantic-web standard for ontologies, the Web Ontology Language (OWL) [1], has limited built-in facilities for reuse. The only way an OWL-based ontology can re-use work done in other OWL ontologies is by importing the entire ontology. Given the considerable size of some ontologies (e.g. Chemical Entities of Biological Interest [5] and the Ontology of Biomedical Investigations [2]), this can be a major obstacle to ontology re-use among ontology developers.

For this reason, the Ontology for Biomedical Investigations (OBI) Consortium [2] has defined the Minimum Information to Reference an External Ontology Term (MIREOT) [3] to facilitate sharing of components of OWL ontologies such as individual classes rather than entire imports of ontologies. Xiang et al [6]

describe an implementation of the MIREOT principles in OntoFox, a web service for creating subsets of the classes and object properties of a source ontology for import into an ontology under development. However, for users of ontology editors like Protégé [4], the fact that OntoFox is a web service makes its integration into the ontology development process somewhat unwieldy. The user must go outside Protégé, generate import files using OntoFox, and then import those files into the ontology in Protégé.

While the specific issues we want to resolve are linked to the OWL language, our primary aim is to provide a simple MIREOT solution for the user community of Protégé. For that purpose, we present a cohesive suite of plugins built for Protégé 4.1 to simplify the use of basic MIREOT principles.

2 Background

The OWL language is designed to facilitate data sharing on the web in a machine readable way through the creation of ontologies. All OWL ontologies can be expressed as an RDF [7] graph, an important type of datastore for the Semantic Web. The most current version as of this writing is OWL 2.

Protégé is a popular, open-source ontology editor built and maintained by researchers at Stanford. It provides a graphical interface that uses the OWL API [8] to build and maintain ontologies. It also includes an OSGI-compatible framework that makes it easy to extend via plugins.

A plugin [9] for Protégé 3 with similar functionality was developed by the Stanford Center for Biomedical Informatics Research. However, this plugin has not yet been updated for Protégé 4 and thus lacks support for OWL 2.

3 Methods

The OBI Consortium has defined a set of principles – which we refer to generically as “MIREOT principles” – for importing only one or more OWL components from another ontology. These principles specify the minimum unit of import as follows: the IRI of the source ontology, the IRI of the source component being imported (e.g., class, object property, etc.), and the IRI of the new parent component in the target ontology. Additionally, the Consortium defined a new annotation property in the Information Artifact Ontology (IAO)[10], named “iao:imported_from”. The value of this property on the imported component is the IRI of the source ontology, ideally. Finally, they recommend that certain relevant annotations be imported, such as “rdfs:label” and “iao:definition”.

We have implemented crucial parts of the MIREOT specification in a suite of Protégé plugins. Our implementation of the MIREOT specification differs in the following ways:

1. It pulls every annotation of the source component, not just “rdfs:label” and “iao:definition”.
2. It saves all relevant axioms in the target ontology instead of external files.

Additionally, we made the following assumptions:

1. Only the most recent source ontology is preserved in the "iao:imported_from" annotation property. Any similar annotations are stripped from the component to avoid confusion about which ontology was the direct source of a class.
2. If an ontology lacks an IRI, then the URL from where it was loaded is used instead.
3. If an annotation property that is not found in the target ontology is imported, then its "rdfs:label" is transferred as well, if it exists.

4 Results

The suite of plugins includes one tab plugin that contains three view plugins. A tab plugin tends to be simple; it contains multiple views that should work together to accomplish some overarching purpose. A view plugin represents the building blocks of a tab plugin and implement most of the actual functionality within a tab plugin. The tab plugin does not manage any additional state.

The first and most novel of the three view plugins is the "MIREOT Additional Ontology View". It allows users to load an additional ontology and search it for classes or object properties. The search algorithm matches the query against a configurable set of annotations on each component. The currently searchable annotation properties are "rdfs:label", "rdfs:comment", and "iao:definition". All loading and searching is done in a separate thread to keep the user interface fluid and stable.

The other two view plugins represent the active ontology loaded by Protégé. They display the hierarchy of either the classes or the object properties of the active ontology. Additionally, they allow a user to drag a component from the results panel of the "MIREOT Additional Ontology View" and drop it under the new parent of the component in the target ontology.

A short video demonstrating a subset of the features of the suite can be found online at <http://vimeo.com/46860781>. The suite and installation instructions can be found online at <http://bitbucket.org/sohj/mireot-protege-plugin>.

5 Discussion and Future Work

We have developed a suite of Protégé plugins that implements crucial parts of the MIREOT specification defined by the OBI Consortium. To our knowledge, this is the first Protégé plugin to do so. Although other tools such as OntoFox also implement MIREOT, our plugins place the workflow directly into the most commonly used ontology editor, and thus are more likely to facilitate MIREOT in practice. Although it has face validity, verification of this conjecture requires further study.

Our implementation of the MIREOT principles differ from the approach taken in the OBI specification in several notable ways:

- We import all of the annotations of an imported component. We do this because vital information about some components are included in annotations other than “rdfs:label” and “iao:definition”.
- We keep the axioms in the main ontology. This decreases the friction of development for Protégé users while still maintaining necessary provenance information for editors, consumers, and users.
- We report only the most recent source ontology information. This will prevent confusion based on multiple “iao:imported_from” annotations that do not allow one to distinguish from which ontology a component was imported directly.
- We save the ontology URL as the IRI in the case of ontologies that do not have an IRI. We feel, it is important that the provenance of the imported component is preserved somehow.

One of the major reasons that OBI separates the imported terms from the main ontology is to allow for automated updates. For this reason, we intend to develop a tool which will allow users to update the terms while still keeping them in the main ontology using the saved provenance information.

6 Acknowledgements

This work was funded by award R01GM101151 from the National Institute for General Medical Sciences. This paper does not represent the views of NIGMS or the National Institutes of Health.

References

1. W3C Consortium, <http://www.w3.org/TR/owl2-overview>
2. OBI Consortium, <http://purl.obolibrary.org/obo/obi>
3. Courtot M, Gibson F, Lister AL, Malone J, Schober D, Brinkman RR, et al. MIREOT: The minimum information to reference an external ontology term. *Applied Ontology*. [10.3233/AO-2011-0087]. 2011;6(1):23-33.
4. **Protégé**, <http://protege.stanford.edu>
5. de Matos, P., Alcntara, R., Dekker, A., Ennis, M., Hastings, J., Haug, K., Spiteri, I., Turner, S., and Steinbeck, C. (2009). Chemical entities of biological interest: an update. *Nucleic Acids Res.* in the press.
6. Xiang Z, Courtot M, Brinkman RR, Ruttenberg A, He Y. OntoFox: web-based support for ontology reuse. *BMC research notes*. 2010;3:175.
7. W3C Consortium, <http://www.w3.org/TR/rdf-concepts/>
8. Matthew Horridge, Sean Bechhofer. The OWL API: A Java API for OWL Ontologies. *Semantic Web Journal* 2(1), Special Issue on Semantic Web Tools and Systems, pp. 11-21, 2011
9. Stanford Center for Biomedical Informatics Research, <http://protegewiki.stanford.edu/wiki/BioPortal.Import.Plugin>
10. OBI Consortium, <http://purl.obolibrary.org/obo/iao.owl>

Browsing Causal Chains in a Disease Ontology

Kouji KOZAKI¹, Hiroko KOU¹, Yuki Yamagata¹, Takeshi IMAI², Kazuhiko OHE²
and Riichiro MIZOGUCHI

¹The Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka, 567-0047 Japan

²Department of Medical Informatics, Graduate School of Medicine, The University of Tokyo,
7-3-1, Hongo, Bunkyo-ku, Tokyo, Japan
{miz, kozaki, kou, yamagata}@ei.sanken.osaka-u.ac.jp
{ken, kohe}@hcc.h.u-tokyo.ac.jp

Abstract. In order to realize sophisticated medical information systems, many medical ontologies have been developed. We proposed a definition of disease based on River Flow Model which captures a disease as a causal chain of clinical disorders. We also developed a disease ontology based on the model. It includes definitions of more than 6,000 diseases with 17,000 causal relationships. This demonstration summarizes the disease ontology and a browsing system for causal chains defined in it.

Keywords: disease ontology, causal chain, ontology visualization

1 Introduction

In these days, medical information systems store huge amount of data. Semantic technologies are expected to contribute to effective use of them, and many medical ontologies such as OGMS [1], DOID [2], and IDO [3] have been developed for realizing sophisticated medical information systems. They mainly focus on the ontological definition of disease with related properties. The authors proposed a definition of a disease involving capturing a disease as a causal chain of clinical disorders and a computational model called *River Flow Model of Disease* [4, 5]. Based on the model, we developed a disease ontology which includes definitions of about 6,000 diseases with causal relations between 17,000 clinical disorders (abnormal state). This demonstration shows a system to browse causal chains defined in the disease ontology.

This article is organized as follows. The next section overviews the *River Flow Model of Disease* discussed in [4, 5]. Section 3 summarizes developments of the disease ontology and a browsing system for the disease ontology. Finally, we present concluding remarks together with future work.

2 River flow model of disease

After it begins to exist, a typical disease, as a dependent continuant, enacts extending, branching, and fading processes before it disappears. Thanks to these processes, a

disease can be identified as a continuant that is an enactor of those processes. Such an entity (a disease) can change according to its phase while keeping its identity. On the basis of this observation, we defined a disease as:

Definition 1: Disease [4]

A disease is a dependent continuant constituted of one or more causal chains of clinical disorders appearing in a human body and initiated by at least one disorder.

Note that, although any disease has dynamic flows of the propagation of causality as its internal processes, it is the enactor of its external processes, such as branching and extending its causal chain of disorders. When we collect individual causal chains belonging to a particular disease type (class), we are able to find a common causal chain (partial chain) that appears in all of the instance chains. By generalizing such a partial chain, we obtain the notion of a core causal chain of a disease as follows:

Definition 2: Core causal chain of a disease

A sub-chain of the causal chain of a disease whose instances are included in all the individual chains of all instances of a particular disease type. It corresponds to the essential property of a disease type.

Definition 2 provides a necessary and sufficient condition for determining the disease type to which a given causal chain of clinical disorders belongs. That is, when an individual causal chain of clinical disorders includes instances of the core causal chain of a particular disease type, it belongs to the disease type. We can thus define such a disease type that includes all possible variations of physical chains of clinical disorders observed for patients who contract the disease. According to a standard definition of subsumption, we can introduce an *is-a* relation between diseases using the chain-inclusion relationship between causal chains.

Definition 3: *Is-a* relation between diseases

Disease A is a supertype of disease B if the core causal chain of disease A is included in that of disease B. The inclusion of nodes (clinical disorders) is judged by taking an *is-a* relation between the nodes into account, as well as sameness of the nodes[4].

Definition 3 helps us systematically capture necessary and sufficient conditions of a particular disease which roughly corresponds to the so-called “main pathological/etiological conditions”. Fig. 1 shows the main types of diabetes constituted by corresponding types of causal chains. Assume, for example, that (non-latent) diabetes and type-I diabetes are respec-

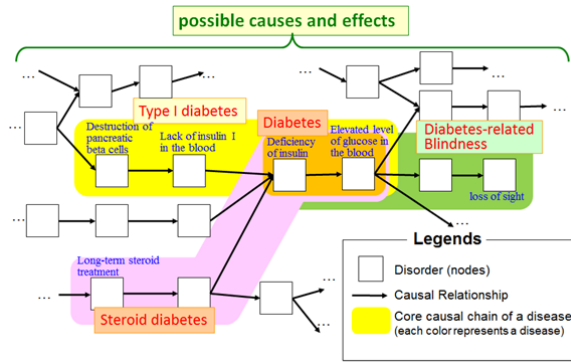


Fig. 1 Types of diabetes constituted of causal chains.

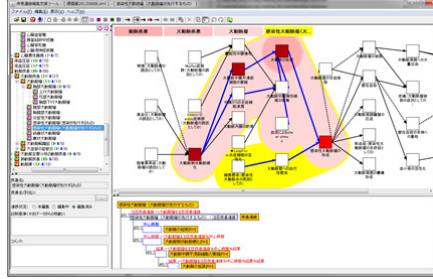


Fig. 2 A visual editing tool for causal chains to define disease concepts.

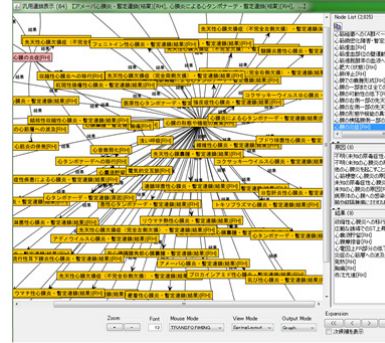


Fig. 3 A browsing tool for causal chains.

tively defined as *<deficiency of insulin → elevated level of glucose in the blood>* and *<destruction of pancreatic beta cells → lack of insulin I in the blood → deficiency of insulin → elevated level of glucose in the blood>*. Then, we get *<type-I diabetes is-a (non-latent) diabetes>* according to Definition 3.

3 Development of a disease ontology and a browsing tool for it

Based on the model discussed in the previous section, we developed a disease ontology. Its design policy and conceptual structures of the ontology were decided through repeated discussions by ontology engineer and medical expert. We defined its upper level concepts (classes) such as clinical disorders (abnormal states), causal chains, causal relationships (cause and effect), etc. based on YAMATO¹. Disease concepts were defined sub-class of these concepts by clinicians in 12 special fields. Although the disease ontology are developed using Hozo², we developed a visual editing tool for it so that clinicians can easily edit the definition of disease concepts. Fig. 2 shows its user interface. It visualizes causal chains defined in a selected disease as directed graph like Fig.1. In the graph, nodes represent clinical disorders and links represents causal relationships between them. When users edit the graph, it automatically translated into ontology in Hozo's format. The ontology could be exported in OWL formats thanks to the export functions of Hozo.

Note here that each clinician defined disease concepts in his/her special field without knowing how other diseases were defined in other filed by others. After they finished defining disease concepts, we collected all causal relationships from all disease concepts defined in the 12 special fields. Then, we combined causal chains which included the same clinical disorder. As the result, we obtained causal chains which include about 17,000 clinical disorders defined in 6,000 diseases. They represent possible causal chains in human body.

In order to browsing these causal chains, we developed a browsing tool (Fig. 3). It visualizes causal chains defined in the disease ontology and the user can browse

¹ http://www.ei.sanken.osaka-u.ac.jp/hozo/onto_library/upperOnto.htm

² <http://www.hozo.jp/>

them through some functions such as searching, tracing, changing layout, zooming etc. Although it is implemented as a client application using Hozo's ontology API, we plan to develop web services version of it. We also consider publishing the disease ontology as Linked Open Data with SPARQL endpoint to get their causal chains.

Currently, we focus on definitions of disease in order to provide a basic knowledge for medial information systems without considering particular applications. When we use the disease ontology for a specific application, we will consider making some extension the ontology and browsing system according to the purpose.

4. Concluding Remarks

We developed a disease ontology based on River Flow Model and a browsing tool for causal chains defined in it. Because the ontology is based on ontological consideration of causal chains, it could capture characteristics of diseases appropriately. The definition of disease as causal could be also very friendly to clinicians since it is similar to their understanding of disease in practice. Moreover, it could include richer information about causal relationships in disease than other disease ontologies or medical terminologies such as SNOMED-CT. Currently we are refining the ontology through reviewing definitions of disease concepts. We are also organizing definitions of clinical disorders into an abnormality ontology based on YAMATO. After these refinement processes, the ontology could become more systematized knowledge. Other future works includes development of a web service for browsing causal chains and publishing the disease ontology as Linked Open Data.

The demonstration is available at the URL: <http://www.hozo.jp/demo/>

Acknowledgement

A part of this research is supported by the Japan Society for the Promotion of Science (JSPS) through its "FIRST Program" and the Ministry of Health, Labour and Welfare, Japan.

References

1. Scheuermann, R. H., Ceusters, W., and Smith, B. (2009) *Toward an Ontological Treatment of Disease and Diagnosis*. Proc. of the 2009 AMIA Summit on Translational Bioinformatics, 116-120, San Francisco, CA.
2. Osborne, J. D., et al. (2009) *Annotating the human genome with Disease Ontology*. BMC Genomics 10(1):S6.
3. Cowell, L. G. and Smith, B (2010) *Infectious Disease Ontology*. Infectious Disease Informatics, Chapter 19, Sintchenko V., 373-395.
4. Mizoguchi, R., et al. (2011) *River Flow Model of Diseases*, Proc. of ICBO2011, 63-70, Buffalo, USA.
5. Kozaki, K., et al. (2012) *Identity Tracking of a Disease as a Causal Chain*, Proc. of ICBO2012

Real Time Fire Monitoring Using Semantic Web and Linked Data Technologies^{*}

K. Kyzirakos¹, M. Karpathiotakis¹, G. Garbis¹, C. Nikolaou¹, K. Bereta¹, M. Sioutis¹, I. Papoutsis², T. Herekakis², D. Michail³, M. Koubarakis¹, and C. Kontoes²

¹ National and Kapodistrian University of Athens

² National Observatory of Athens

³ Harokopio University of Athens

koubarak@di.uoa.gr

1 Introduction

Fire monitoring and management in Mediterranean countries such as Greece is of paramount importance. Almost every summer massive forest fires break out, causing severe destruction and even human life losses. Thus, European initiatives in the area of Earth Observation (EO), such as GMES SAFER⁴, have supported the development of relevant operational infrastructures. In the context of the European project TELEIOS⁵, we aim at developing a fire monitoring service, that goes beyond operational systems currently deployed in various EO data centers, by building on Semantic Web and Linked Data technologies.

In this demonstration we present the fire monitoring service that we have implemented using TELEIOS technologies focusing on its Semantic Web related functionality. The service implements a processing chain where raw satellite images are analyzed and hotspots (pixels of the image corresponding to geographic regions possibly on fire) are detected. The products of this analysis are encoded in RDF, so they can be combined with auxiliary linked geospatial data (e.g., GeoNames, OpenStreetMap). By comparing detected hotspots with auxiliary data their accuracy can be determined. For example, hotspots lying in the sea are retrieved and marked as invalid. Additionally, we can combine diverse information sources and generate added-value thematic maps which are very useful to civil protection agencies and firefighting teams during emergency situations.

In the rest of this demo paper we first describe in short the contributions of project TELEIOS. Then, we present the developed fire monitoring service and its advances compared to relevant deployed services. Finally, we describe how we plan to present this service through a live demonstration.

^{*} This work has been funded by the FP7 project TELEIOS (257662).

⁴ <http://www.emergencyresponse.eu/>

⁵ <http://www.earthobservatory.eu/>

2 TELEIOS Contributions

TELEIOS is a recent European project that addresses the need for scalable access to PBs of EO data and the effective discovery of knowledge hidden in them. TELEIOS started on September 2010 and it will last for 3 years. In the first 18 months of the project, we have made significant progress in the development of state-of-the-art techniques in Scientific Databases, Semantic Web and Image Mining and have applied them to the management of EO data.

We have developed SciQL [6], a new SQL-based query language for scientific applications with arrays as first-class citizens. This allows us to store EO data (e.g., satellite images) in the database, and express low level image processing (e.g., georeferencing) and image content analysis (e.g., pixel classification) in a user-friendly high-level declarative language that provides efficient array manipulation primitives. SciQL is implemented on top of the state of the art column-store DBMS MonetDB⁶, which offers capabilities for scalable querying.

We have also developed the model stRDF, an extension of the W3C standard RDF for representing time-varying geospatial data [1, 2]. The accompanying query language, stSPARQL, is an extension of the query language SPARQL 1.1 and it has been implemented in the semantic geospatial DBMS Strabon⁷, which offers scalability to billions of stRDF triples [4]. In applications, such as the fire monitoring service presented here, stRDF is used to represent satellite image metadata (e.g., time of acquisition), knowledge extracted from satellite images (e.g., spatial extent of hotspots), and auxiliary geospatial data encoded as linked data (e.g., GeoNames). So, rich user queries that cannot be expressed with database technologies of EO data centers can be expressed in stSPARQL. This is illustrated in this demonstration, but also in [3] where some of the knowledge discovery techniques pioneered by TELEIOS are also discussed.

3 The NOA Fire Monitoring Application

The National Observatory of Athens (NOA) operates an MSG/SEVIRI satellite acquisition station, and has developed a real-time fire hotspot detection service for effectively monitoring a fire-front. We present this service graphically in Figure 1 and explain below in some detail the improvements that we have achieved by using TELEIOS technologies.

On a regular basis (5 or 15 minutes) satellite images arrive at the acquisition station and are stored as arrays in MonetDB. The arrays are processed with a series of SciQL queries (for cropping, georeferencing, and hotspot detection) and shapefiles describing the detected hotspots are generated for each acquisition. Because of the low spatial resolution of the SEVIRI instrument, possible errors in image georeferencing, and potential weaknesses of the algorithms in [5], the derived products have limited accuracy for specific scenarios. We increase their accuracy by combining them with linked geospatial data.

⁶ <http://www.monetdb.org/>

⁷ <http://www.strabon.di.uoa.gr/>

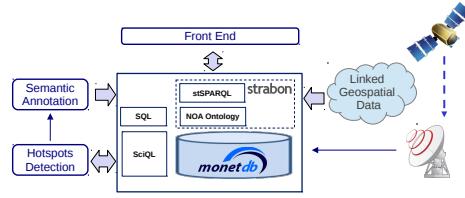


Fig. 1. The NOA fire monitoring service

The main problem with the product accuracy is the existence of false alarms in the fire detection technique. For example, hotspots shown to be occurring in the sea or in locations with inconsistent land use (e.g., urban areas) should be considered false alarms instead of forest fire emergency situations. To query generated data using stSPARQL and combine it with linked data, we derive stRDF triples from the generated shapefiles. The derived triples mainly hold information about the coordinates of detected fire location, the date and time, and the confidence level of the detection for each hotspot. We execute stSPARQL updates which compare the hotspots with two RDF datasets and mark as false positives the hotspots that lie in the sea or in locations with inconsistent land use. The datasets that we use are: (i) a dataset describing the coastline of Greece⁸, and (ii) a dataset describing the Greek environmental landscape⁹.

Another problem is spatial and temporal inconsistencies in hotspots generated by the processing chain due to using a single image acquisition and not using information from previous acquisitions. A simple heuristic we use is retrieving hotspots that were detected at least once during a specific time period (e.g., half hour) but they were not detected in the last acquisition. In this case we add a virtual hotspot for the last acquisition with a confidence level equals to the average confidence level of the real detections during the last half hour.

Finally, the need to generate added-value thematic maps is addressed. The Linked Open Data Cloud supplies an abundance of datasets, in addition to internal EO data, that cover a large variety of geospatial entities, ranging from fine-grained geometric objects like fire stations, to coarser ones like countries. So, instead of manually combining heterogeneous data, a user can pose an stSPARQL query for each layer that she wants to depict in a map and overlay the retrieved data using the ability of Strabon to expose data in KML or GeoJSON. Although this service has been designed for Greece, it can be applied to any geographic area due to the generality of the used technologies (e.g., RDF, linked data, KML).

⁸ This dataset has been compiled in the context of TELEIOS and is available from http://geo.linkedopendata.gr/coastline_gr/.

⁹ <http://geo.linkedopendata.gr/corine/>



Fig. 2. The NOA fire monitoring application GUI

4 Demonstration

The demonstration consists of three parts. First, the user will start an instance of the processing chain described above and browse its results in the GUI of the application (shown in Figure 2). The user can also use the search functionality or pose stSPARQL queries to retrieve fire products of previously executed instances of the processing chain. Second, the demonstration focuses on the improvement of the accuracy of the fire products. We will demonstrate how stSPARQL update statements and linked geospatial data are used in order to increase the accuracy of derived fire products. Finally, the creation of added-value thematic maps by combining information from different data sources will be demonstrated.

References

1. Koubarakis, M., Karpathiotakis, M., Kyzirakos, K., Nikolaou, C., Sioutis, M.: Data Models and Query Languages for Linked Geospatial Data. In: Reasoning Web. Semantic Technologies for Advanced Query Answering. LNCS, vol. 7487, pp. 290–328. Springer (2012)
2. Koubarakis, M., Kyzirakos, K.: Modeling and querying metadata in the semantic sensor web: The model stRDF and the query language stSPARQL. In: ESWC (2010)
3. Koubarakis, M., Sioutis, M., Kyzirakos, K., Karpathiotakis, M., Nikolaou, C., Vassos, S., Garbis, G., Bereta, K., Dumitru, O.C., Molina, D.E., Molch, K., Schwarz, G., Datcu, M.: Building Virtual Earth Observatories using Ontologies, Linked Geospatial Data and Knowledge Discovery Algorithms. In: ODBASE (2012)
4. Kyzirakos, K., Karpathiotakis, M., Koubarakis, M.: Strabon: A Semantic Geospatial DBMS. In: Proceedings of the 11th International Semantic Web Conference (2012)
5. Sifakis, N., Iossifidis, C., Kontoes, C., Keramitsoglou, I.: Wildfire Detection and Tracking over Greece Using MSG-SEVIRI Satellite Data. Remote Sensing 3, 524–538 (2011)
6. Zhang, Y., Kersten, M.L., Ivanova, M., Nes, N.: SciQL: bridging the gap between science and relational DBMS. In: IDEAS (2011)

Demonstrating Blank Node Matching and RDF/S Comparison Functions

Christina Lantzaki, Yannis Tzitzikas, and Dimitris Zeginis*

Institute of Computer Science, FORTH-ICS, GREECE, and
Computer Science Department, University of Crete
{kristi,tzitzik,zeginis}@ics.forth.gr

This demo paper accompanies the ISWC'2012 paper: [7]

Motivation

The ability to compute the differences that exist between two RDF/S Knowledge Bases (for short KBs) is important for aiding humans to understand the evolution of knowledge, and for reducing the amount of data that need to be exchanged and managed over the network in order to build SW synchronization, versioning and replication services [2, 3, 1, 8, 6].

A rather peculiar but quite flexible feature of RDF is that it allows the representation of *blank nodes*: a *blank node* (or anonymous resource or bnode) is a node in an RDF graph which is not identified by a URI and is not a literal. Several KBs rely heavily on blank nodes as they are convenient for representing complex attributes (e.g. an attribute **address**) without having to name explicitly the auxiliary node that connects together the values that constitute the complex value (e.g. the particular **street**, **number** and **postal code** values). Bnodes are also convenient for resources whose identity is unknown but their attributes (either literals or associations with other resources) are known. According to [4], blank nodes is an inevitable reality, e.g. the data fetched from the “hi5.com” domain consist of 87.5% of blank nodes.

The inability to match bnodes increases the delta size and does not assist in detecting the changes between subsequent versions of a KB [4].

Approach

Although there are several works on blank node and comparison functions (for details see [7]), the problem has not been thoroughly studied. To the best of our knowledge, the only work that attempts to establish a bnode mapping for reducing the size of deltas also for the case of non equivalent KBs is [7] (ISWC'12). Finding such a mapping can be considered as a preprocessing step, a task that is carried out before a differential function is applied.

In [7] we prove that finding the optimal mapping is NP-Hard in the general case, and polynomial if there are no directly connected bnodes. Subsequently,

* Current affiliation: Information Systems Lab, University of Macedonia, Thessaloniki, Greece, zeginis@uom.gr

we present two main algorithms: (a) the *AlgHung* algorithm, and (b) the *AlgSign* algorithm. *AlgHung* solves the optimization problem using the *Hungarian algorithm* [5], an algorithm for solving the *assignment problem*. For the cases where there are directly connected bnodes, a variation of *AlgHung* is used for producing an *approximate* solution. The time complexity of the *AlgHung* in any case is in $O(n^3)$, where n is the number of bnodes.

For making the application of this method feasible also to very large KBs, at the cost of probably bigger deltas, [7] also proposed a *signature*-based method, *AlgSign*, whose complexity is in $O(n \log n)$. For these algorithms, the reported experimental results over real and synthetic datasets showed significant reductions of the sizes of the computed deltas.

What will be Demonstrated

We will demonstrate a tool called **BNodeDelta** which supports all algorithms presented at [7]. With this tool, the user (human or other program), specifies the two KBs to be compared (which can be stored in local files or fetched from the network using HTTP), then specifies the bnode mapping algorithm to be used, and then gets back statistics (about the KBs and their delta) and the delta itself (sets of triples to be added and deleted). Furthermore the tool can take as input a *namespace mapping table* (if a namespace *nm1* is mapped to a *nm2* then they are considered equal at the comparison phase).

We will demonstrate the system using two real datasets available in the LOD cloud: the *Swedish open cultural heritage* dataset¹, and the *Italian Museums* dataset², published from LKDI³. We shall also use synthetically generated data.

Figure 1 (left) shows the command line interface which shows the basic statistics for the Italian dataset (more statistics can be placed on demand in a file called "statistics"). Figure 1 (right) shows an excerpt of the file that contains the added triples (assuming the user requested the output *delta* in RDF/XML format).

We will give emphasis on the bnode mapping algorithms, specifically we will show the size of the outcome of the differential function Δ_e (where $\Delta_e(K \rightarrow K') = \{Add(t) \mid t \in K' - K\} \cup \{Del(t) \mid t \in K - K'\}$) for the cases: *AlgHung*, *AlgSign*, a random bnode mapping algorithm, and no bnode mapping at all.

Time Efficiency (comparative results). In both algorithms (*AlgHung* and *AlgSign*) the required time depends on the number of bnodes of the two KBs and the average number of triples to which a bnode participates. *AlgHung* needs 5.4 seconds over datasets of average 3,650 triples and 525 bnodes, and 9.6 minutes for datasets of average 49,900 triples and 6,390 bnodes, whereas *AlgSign* needs only 0.34 seconds and 0.92 seconds respectively. These results show that *AlgSign*

¹ <http://thedatahub.org/dataset/swedish-open-cultural-heritage> used from <http://kringla.nu/kringla/> for providing information on cultural data of Sweden

² <http://thedatahub.org/dataset/museums-in-italy>

³ <http://www.linkedopendata.it/>

can be efficient also in bigger datasets (we will also show that two KBs with 153,600 bnodes can be compared at less than 11 seconds).



Fig. 1. Basic statistics and added triples of delta over Italian datasets

Delta Sizes (comparative results). As regards delta size, in the first dataset without bnode mapping the delta contains 5,771 triples, whereas with *Alg_{Hung}* it contains 311 triples, and with *Alg_{Sign}* 419 triples.

In the second dataset without bnode mapping the delta contains 43,770 triples, whereas with *Alg_{Hung}* it contains 6 triples, and same for *Alg_{Sign}*.

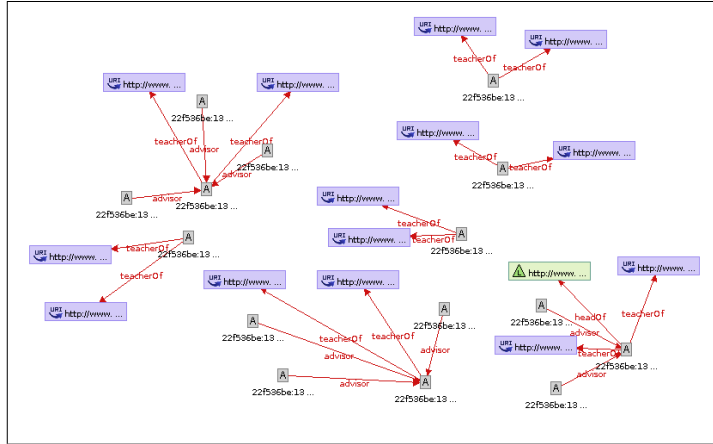


Fig. 2. Visualization of the added triples over synthetic datasets with RDF-Gravity

Delta Visualization. Apart from the benefits in a versioning/synchronization scenario, the achieved delta size reduction makes the visualization and exploration of the delta much easier. For this reason, *BNodeDelta* offers several choices for formatting the output delta in order to aid further processing or visualization. One option returns the delta in two separate files, one containing the *deleted* triples, the other the *added* triples, both in RDF/XML format. Each of these files can be explored and visualized with various RDF/S visualization tools.

For instance, we loaded to RDF-Gravity⁴ the RDF/XML file that contains the *added* triples of the delta over the synthetic dataset. Figure 2 shows the derived visualization

Note that if the delta size is small, *both* added and deleted triples can be visualized as a single graph. In such cases, **BNodeDelta** also returns a graph visualization. For example, Figure 3 shows the graph of delta over the Italian datasets where the added elements are in *green* while the deleted are in *red*.

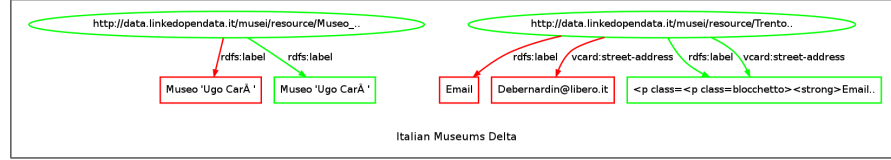


Fig. 3. A graph-based visualization of the delta over Italian datasets (in red the deleted triples, in green the added ones)

The above examples, just show that bnode mapping can reduce the delta to sizes appropriate for graph-based visualization (something not possible without bnode mapping).

Software and datasets are available to download and use from <http://www.ics.forth.gr/isl/BNodeDelta>.

References

1. T. Berners-Lee and D. Connolly. "Delta: An Ontology for the Distribution of Differences Between RDF Graphs", 2004. <http://www.w3.org/DesignIssues/Diff>.
2. J. Heflin, J. Hendler, and S. Luke. "Coping with Changing Ontologies in a Distributed Environment". In *AAAI-99 Workshop on Ontology Management*, 1999.
3. M. Klein, D. Fensel, A. Kiryakov, and D. Ognianov. "Ontology versioning and change detection on the web". In *Procs of EKAW'02*, 2002.
4. A. Mallea, M. Arenas, A. Hogan, and A. Polleres. On blank nodes. In *Procs of the 10th Intern. Semantic Web Conference (ISWC 2011)*. Springer, October 2011.
5. J. Munkres. Algorithms for the assignment and transportation problems. *J-SIAM*, 5(1), 1957.
6. B. Schandl. Replication and versioning of partial rdf graphs. ESWC'10, 2010.
7. Y. Tzitzikas, C. Lantzaki, and D. Zeginis. "Blank Node Matching and RDF/S Comparison Functions". ISWC'12, 2012.
8. D. Zeginis, Y. Tzitzikas, and V. Christophides. "On the Foundations of Computing Deltas Between RDF Models". In *Procs of ISWC-07*, 2007.

⁴ <http://semweb.salzburgresearch.at/apps/rdf-gravity>

Creating Enriched YouTube Media Fragments With NERD Using Timed-Text

Yunjia Li¹, Giuseppe Rizzo², Raphaël Troncy², Mike Wald¹, and Gary Wills¹

¹ University of Southampton, UK

`yl2@ecs.soton.ac.uk, mw@ecs.soton.ac.uk, gbw@ecs.soton.ac.uk`

² EURECOM, Sophia Antipolis, France,

`giuseppe.rizzo@eurecom.fr, raphael.troncy@eurecom.fr`

Abstract. This demo enables the automatic creation of semantically annotated YouTube media fragments. A video is first ingested in the Synote system and a new method enables to retrieve its associated subtitles or closed captions. Next, NERD is used to extract named entities from the transcripts which are then temporally aligned with the video. The entities are disambiguated in the LOD cloud and a user interface enables to browse through the entities detected in a video or get more information. We evaluated our application with 60 videos from 3 YouTube channels.

Keywords: Media fragment, media annotation, NERD

1 Introduction

New W3C standards such as HTML5, Media Fragment URI and the Ontology for Media Resources have finally made videos a first class citizen on the Web. Indexing a video at a fine grained level such as the scene is, however, not yet a common practice on popular video sharing platform. In this demo, we propose to use NERD for extracting named entities from timed text associated to videos in order to generate media fragments annotated with resources from the LOD cloud. Our contributions include a new combined strategy for extracting named entities, temporal alignment of the named entities with the video and a user interface for browsing the enriched videos.

The LEMO multimedia annotation framework provides a unified model to annotate media fragments while the annotations are enriched with contextually relevant information from the LOD cloud [1]. Yovisto provides both automatic video annotations based on video analysis and collaborative user-generated annotations which are further linked to entities in the LOD cloud with the objective to improve the searchability of videos [5]. SemWebVid automatically generates RDF video descriptions using their closed captions [4]. The captions are analyzed by 3 web services (AlchemyAPI, OpenCalais and Zemanta) but chunked into blocks which make loose the context for the NLP tools. In this demo, we propose a new combined strategy using 10 different NER tools based on NERD [3]. In addition, we propose a new method to get the subtitles of a video and to analyze them globally while re-creating the temporal alignment.

2 Technical Architecture

This demo is powered by the integration and extension of two systems: Synote [2] and NERD [3] (Figure 1a). A user creates a new recording in Synote from any



Fig. 1. a) Synote and NERD integration architecture. b) The Synote UI enriched with NERD and DBpedia.

YouTube video. **(1)** The system first extracts the metadata and the subtitles if available using the YouTube API:

```
GET api/timedtext?v=videoid&lang=en&format=srt&name=trackname
```

In this request, four parameters are required: the YouTube video id *v*, the language of the subtitles *lang*, the timed-text format *format* and the track *name*. **(2)** A prior request is necessary for getting the track name since it is specified by the video owner.

```
GET api/timedtext?v=videoid&type=list
```

(3) The timed text is passed to the NERD client API which sends it to the NERD server. The named entity extraction is then performed on the entire context of the SRT file. **(4)** NERD returns a list of named entities with their type and a URI that disambiguates them, and a temporal window reference *startNPT* and *endNPT* corresponding to the SRT block where the entity appears. NERD exploits a combined strategy where 10 different extractors are used together. The named entity types are aligned yielding to a classification in 8 main types plus the general *Thing* concept. **(5)** On receiving the NERD response, Synote constructs media fragment URIs and uses the Jena RDF API to serialize the fragment annotations in RDF. The vocabularies NERD³, Ontology for Media Resource⁴, Open Annotation⁵ and String Ontology in NIF⁶ are used. Finally, the user interface shows the linking between named entities and media fragments, together with the YouTube video and interactive subtitles. The named entities and related metadata extracted from the subtitles are retrieved through SPARQL queries **(6.a, 6.b)**. If a named entity has been disambiguated with a DBpedia URI **(6.c)**, a SPARQL query is sent to get further data about the entity (e.g. label, abstract, depiction) which is displayed alongside with the named entities.

³ <http://nerd.eurecom.fr/ontology>

⁴ <http://www.w3.org/ns/ma-ont>

⁵ <http://www.openannotation.org/spec/core>

⁶ <http://nlp2rdf.lod2.eu/schema/string>

3 Walk Through Demo

A live demo can be found at <http://linkeddata.synote.org>⁷. A user first logged in on Synote. When going to the recording creation page, a user can start the ingestion of a YouTube video. The recording is then available in the recording list. The “NERD Subtitle” button enables to launch the named extraction process. When completed, a “Preview Named Entities” button enables to go to the player page where named entities can be used to seek in particular video fragments.

Figure 1b shows the screenshot of a preview page. The right column displays the named entities found grouped according to the 8 main NERD categories. The YouTube video is included in the left column together with the interactive subtitles. The named entities are highlighted in different colours according to their categories. If a media fragment is used in the preview page URI, the video starts playing from the media fragment start time and stops playing when the end time is reached. When clicking on a named entity, the video jumps to the media fragment that corresponds to the subtitle block where the named entity has been extracted. If a named entity has been disambiguated with a DBpedia URI, the entity is underlined. In addition, when the entity is hover, a pop-up window shows additional information such as the generic label, abstract and depiction properties. For named entities of type Person, the birth date is displayed while latitude and longitude information are given for Location.

4 Evaluation

We filtered the videos which have subtitles for 3 different channels: *People and Blogs*, *Sports* and *Science and Technology* and collected 60 videos in total (the top 20 for each category). Videos have different duration ranging from 32 to 4505 seconds and different popularity ranging from 18 to 2,836,535 views (on July 30th, 2012). The corpora is available at <http://goo.gl/YhchP> and can be visually explored in Synote at <http://goo.gl/XmMqp> after being logged in with the iswc2012 account. The video #16 is the only one discarded because its subtitles are written in Romanian. The evaluation consisted in two steps: *i*) be able to get all subtitles and *ii*) perform entity recognition using NERD. We combined all extractors supported by NERD and we aligned the classification results to 8 main types (*Event* is only supported by OpenCalais in beta) plus the general type *Thing* used as fallback in the case NERD cannot find a more specific type. We define the following variables: number of documents per category n_d ; total number of words n_w ; number of words per document ratio r_w ; total number of entities n_e ; number of entities per document r_e (Table 1). We observe that *Science and Technology* videos tend to be more about people and organizations while *Sports* videos mention more often locations, time and amount. *People and Blogs* videos have less useful information although it is interesting to see that this type of video can be used to train event detection.

5 Conclusion

This demo paper presents a system that creates media fragments from YouTube videos and annotates their subtitles using NERD. The process includes named entities extraction in timed-text documents. Those entities annotate and enrich media fragments with

⁷ As credentials, please insert for both user and password: “iswc2012”.

pointers to the LOD cloud. We provide a lightweight evaluation of the system in order to show that we are effectively able to retrieve the subtitles of YouTube videos and to run named entities extractions. Although a more thorough analysis will be needed, we already show that videos exhibit a very different behavior in terms of named entities depending on their genre.

	People and Blogs	Sports	Science and Technology
n_d	19	20	20
n_w	7,187	21,944	39,661
r_w	378.26	1,097.20	1,983.05
n_e	610	897	1,303
r_e	32.11	44.85	65.15
Thing	6.68	15.35	14.75
Person	4.42	9.75	14.55
Function	0.74	7.35	1.15
Organization	3.63	9.20	12.25
Location	3.89	8.05	6.40
Product	3.26	2.60	6.40
Time	3.95	13.80	3.35
Amount	5.47	9.30	6.30
Event	0.05	0.00	0.00

Table 1. Upper part shows the average number of named entities extracted. Lower part shows the average number of entities for the 8 NERD top categories grouped by video channels.

Acknowledgments

The research leading to this paper was partially supported by the French National Agency under contracts ANR.11.EITS.006.01, “Open Innovation Platform for Semantic Media” (OpenSEM) and the European Union’s 7th Framework Programme via the projects LinkedTV (GA 287911).

References

1. Haslhofer, B., Jochum, W., King, R., Sadilek, C., Schellner, K.: The LEMO annotation framework: weaving multimedia annotations with the web. *International Journal on Digital Libraries* 10(1), 15–32 (2009)
2. Li, Y., Wald, M., Omitola, T., Shadbolt, N., Wills, G.: Synote: Weaving Media Fragments and Linked Data. In: 5th International Workshop on Linked Data on the Web (LDOW’12) (2012)
3. Rizzo, G., Troncy, R.: NERD: A Framework for Unifying Named Entity Recognition and Disambiguation Extraction Tools. In: 13th Conference of the European Chapter of the Association for computational Linguistics (EACL’12) (2012)
4. Steiner, T.: SemWebVid - Making Video a First Class Semantic Web Citizen and a First Class Web Bourgeois. In: 9th International Semantic Web Conference (ISWC’10) (2010)
5. Waitelonis, J., Ludwig, N., Sack, H.: Use what you have: Yovisto video search engine takes a semantic turn. In: 5th International Conference on Semantic and digital media technologies (SAMT’10) (2011)

Linked Data Fusion in ODCleanStore[★]

Jan Michelfeit and Tomáš Knap

Charles University in Prague, Dept. Software Engineering
Malostranské nám. 25, 118 00 Prague, Czech Republic
michelfeit.jan@gmail.com, tomas.knap@mff.cuni.cz

Abstract. As part of LOD2 project and OpenData.cz initiative, we are developing an ODCleanStore framework enabling management of Linked Data. In this paper, we focus on the query-time data fusion in ODCleanStore, which provides data consumers with integrated views on Linked Data; the fused data (1) has solved conflicts according to the preferred conflict resolution policies and (2) is accompanied with provenance and quality scores, so that the consumers can judge the usefulness and trustworthiness of the data for their task at hand.

The advent of Linked Data [1] accelerates the evolution of the Web into an exponentially growing information space (see the linked open data cloud¹) where the unprecedented volume of data will offer information consumers a level of information integration and aggregation agility that has up to now not been possible. Consumers can now “mashup” and readily integrate information for use in a myriad of alternative end uses. Indiscriminate addition of information can, however, come with inherent problems, such as the provision of poor quality, inaccurate, irrelevant or fraudulent information. All will come with an associate cost of the data integration which will ultimately affect data consumer’s benefit and linked data applications usage and uptake.

To overcome these issues, as part of the *OpenData.cz initiative* and *LOD2 project*², we are developing the *ODCleanStore (ODCS) framework*³ (1) enabling management of Linked Data – data cleaning, linking, transformation, and quality assessment – and (2) providing data consumers with a possibility to consume integrated data, which reduces the costs of the web application development.

The overall picture of ODCS is depicted in Figure 1. ODCS processes *RDF data feeds* (collections of RDF quads, one data feed = one named graph⁴) in the *staging area*; feeds can be uploaded to the staging area by any third-party

[★] The work presented in this article has been funded in part by EU ICT FP7 under No.257943 (LOD2 project), the Czech Science Foundation (GAČR, grant number 201/09/H057), and GAUK 3110.

¹ <http://richard.cyganiak.de/2007/10/lo/>

² <http://opendata.cz>, <http://lod2.eu>

³ To download the code, please visit <http://sourceforge.net/p/odcleanstore>

⁴ RDF triples can be extended to *quads* (s, p, o, g) where g is the named graph [3] to which the data belongs. When talking about “data in the named graph g ”, we mean all the quads $(*, *, *, g)$.

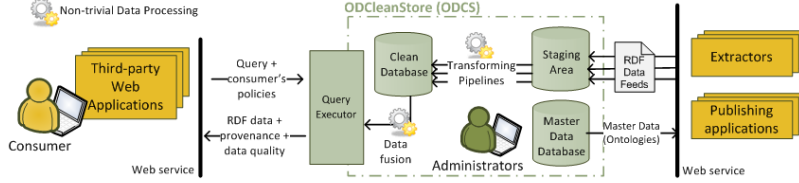


Fig. 1. ODCleanStore Framework

application registered in ODCS, e.g. by various extractors. Based on the identifier of the feed, the appropriate *transforming pipeline* is launched; the pipeline successively executes a defined (and customizable) set of transformers ensuring that data in the processed feed is cleaned, resources deduplicated and linked to already existing resources in the *clean database* or in the linked open data cloud, data is enriched with new resources, arbitrarily transformed, and the quality of the feed (*graph score*) is assessed. When the pipeline finishes, the augmented RDF feed is populated to the clean database together with any auxiliary data and metadata created during the pipeline execution, such as links to other resources or metadata about the feed’s graph score.

Data consumers can query (via third-party applications) the clean database to obtain data about the certain resource (e.g. a city, such as the German city “Berlin”). Since the same resource can be described by various sources (feeds), conflicts may arise when integrating data about that city. To solve this, ODCS applies in the *data fusion algorithm* certain conflict resolution policies which resolve data conflicts in the resulting RDF data; these policies can be customized by the consumer. Furthermore, the resulting integrated RDF data is supplemented with provenance metadata (data origin) and quality scores of the integrated quads, so that data consumers can judge the usefulness and trustworthiness of the resulting data for their task at hand; the quality score is influenced by the quality of the feed the triples originate from (graph score) and by the applied conflict resolution policy [4]. The data fusion algorithm runs during query time, because consumers in different situations can have different requirements on the data.

This paper briefly describes the data fusion algorithm in ODCS in Section 1; the algorithm is fully described in [4]. The practical demonstration⁵ based on the illustrative examples in Section 1 gives further insight into the work of the data fusion algorithm.

To the best of our knowledge, there is just one another linked data fusion software – Sieve – currently under development [5]. Sieve is part of Linked Data Integration Framework⁶. Differently from our approach, Sieve fuses data while

⁵ <http://www.ksi.mff.cuni.cz/~knap/iswc12>

⁶ <http://www4.wiwiiss.fu-berlin.de/bizer/ldif/>

being stored to the clean database and not during execution of queries, thus, provides no data fusion customization during data querying.

1 Linked Data Fusion

Suppose that the clean database of ODCS contains data about the German city Berlin coming from multiple sources – DBpedia, GeoNames, and Freebase⁷. Let us assume that Alice, a data consumer, is an investigative journalist who is writing a story about Berlin; thus, she submits the keyword “Berlin” to the query execution component of ODCS and she would like to get all the information the framework knows about Berlin fused from the available sources.

When fusing data, the data fusion algorithm in ODCS has to deal with *data conflicts*, which happen when two quads have inconsistent object values for a certain subject s and predicate p ; such quads are called *o-conflicting quads* and the conflicting object values of these o-conflicting quads are called *conflicting values*. The solution of the conflicts is prescribed by the conflict resolution policies, which may be specified globally or per predicate. We distinguish two types of conflict resolution policies – *deciding* and *mediating*. Deciding policies select one or more values from the conflicting values, e.g., an arbitrary value (ANY), maximum value (MAX), the value with the highest quality (BEST), or all conflicting values (ALL). Mediating policies compute new value, e.g. an average (AVG) of the conflicting values. For example, Alice may specify she would like to receive in the response all the distinct values for the subject representing Berlin and predicate `rdf:type` (deciding conflict resolution policy ALL). On the other hand, she may want to compute for the same subject average value (AVG) for the values of the predicate `geo:lat`, select the best value with the highest quality (BEST) for `rdfs:label` of Berlin, and select maximum value (MAX) from the values of the predicate `dbprop:populationTotal` of Berlin.

When describing the data fusion algorithm within execution of consumer’s queries in ODCS, we suppose that the typical pre-fusing processes [2] – *schema mapping* (the detection of equivalent schema elements in different sources) and *duplicate detection* (detection of equivalent resources) has already been done. Therefore, we suppose that (1) proper mappings between ontology elements are available in the master data database in Figure 1, e.g. that `geo:lat` and `fb:location.geocode.latitude` are denoted as equivalent predicates holding latitude of Berlin, and (2) `owl:sameAs` links between resources representing the same entity (the German city Berlin) were created by the proper transformers (linkers) on the transforming pipeline.

The input to the data fusion algorithm is (1) a collection of quads from the clean database to be fused – the quads $(x, *, *, *)$, $(*, *, x, *)$, where x is the URI representing Berlin in some source (2) `owl:sameAs` links between URI resources occurring in the quads (output of the deduplication and schema mapping pre-fusion processes), (3) data fusion settings (including set of selected conflict

⁷ Identifiers for the resource Berlin are: <http://dbpedia.org/resource/Berlin>, <http://sws.geonames.org/2950159/>, <http://rdf.freebase.com/ns/en.berlin>

resolution policies), and (4) graph scores of the named graphs (feeds) from which the quads originate. The output is a collection of fused quads enriched with data quality and source named graphs for each fused quad.

The fusion algorithm firstly replaces URIs of resources representing the same concept (i.e. connected by an `owl:sameAs` links) with a single URI and removes duplicate quads⁸. Consequently, quads are grouped to the sets of *comparable quads* – i.e. quads having the same subject and predicate; o-conflicting quads form subset of the corresponding comparable quads. For each set of comparable quads, two steps (Step S1 and S2) are executed: Step S1 chooses and applies a conflict resolution policy determined by the predicate of the comparable quads and Step S2 computes quality of the quads resulting from Step S1. Multiple real-world cases lead us to three factors influencing the computation of the quality of the resulting fused quads (in Step S2): (1) graph scores of the source named graphs containing the processed comparable quads, (2) number of object values within the set of comparable quads which agree on the same object value, and (3) the difference between conflicting values of the comparable and o-conflicting quads. Details of the quality computation are in [4].

2 Conclusions

This paper introduces query-time data fusion algorithm in ODCleanStore – the framework for managing Linked Data. The practical demonstration⁹ shows the maturity of the algorithm and demonstrates its features – application of conflict resolution policies and computation of the quality of the fused quads. Full theoretical background behind the data fusion algorithm is in [4].

References

1. C. Bizer, T. Heath, and T. Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.
2. J. Bleiholder and F. Naumann. Data fusion. *ACM Comput. Surv.*, 41(1):1:1–1:41, Jan. 2009.
3. J. J. Carroll, C. Bizer, P. Hayes, and P. Stickler. Named graphs, Provenance and Trust. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 613–622, New York, NY, USA, 2005. ACM.
4. T. Knap, J. Michelfeit, and N. M. Linked Open Data Aggregation: Conflict Resolution and Aggregate Quality. *METHOD 2012 : The 1st IEEE International Workshop on Methods for Establishing Trust with Open Data, COMPSAC (to appear)*, 2012. <http://www.ksi.mff.cuni.cz/~knap/files/method.pdf>.
5. P. N. Mendes, H. Mühleisen, and C. Bizer. Sieve: Linked Data Quality Assessment and Fusion. In *1st International Workshop on Linked Web Data Management (LWDM 2011) at the 15th International Conference on Extending Database Technology, EDBT 2012*, March.

⁸ Quads having the same subject, predicate, object, and the named graph.

⁹ <http://www.ksi.mff.cuni.cz/~knap/iswc12>

Making Sense of Research with Rexplore

Enrico Motta¹, Francesco Osborne²

¹ Knowledge Media Institute, The Open University, Milton Keynes, UK
e.motta@open.ac.uk

² Dept. of Computer Science, University of Turin, Turin, Italy
osborne@di.unito.it

Abstract. While there are many tools and services which support the exploration of research data, by and large these tend to provide a limited set of functionalities, which cover primarily ranking measures and simple mechanisms for relating authors. To try and improve over the current state of affairs, we are developing a novel tool for exploring research data, which is called Rexplore. Rexplore builds on an intelligent algorithm for automatically identifying hierarchical and equivalence relations between research areas, to provide a variety of functionalities and visualizations to help users to make sense of research data. These include visualizations to detect trends in research; ways to cluster authors according to several dynamic similarity measures; and fine-grained mechanisms for ranking authors, taking into account parameters such as ranking criterion, career stage, calendar years, publication venues, etc.

Keywords: Research Data, Bibliographic Data, Data Visualization, Data Exploration, Visual Analytics, Scholarly Semantic Relations.

1 Introduction

Understanding what goes on in a research area is no easy task. Typically, for a given topic, this sensemaking process may require exploring information about a variety of entities, such as publications, researchers, research groups, projects, events, and others, as well as understanding the relationships which exist between them. In addition, different categories of users tend to be interested in exploring different aspects of this space. For instance, a 1st year PhD student in the Semantic Web area would likely be interested in the main approaches, projects, and publications relevant to her topic of choice. She will also be interested in identifying the key people and research groups, but her exploration needs will certainly be very different from those of a company, who may want to improve their expertise about a specific topic by establishing a relationship with an appropriate research group on the basis of their expertise, status in the field, and geographical location. Research data are also of great interest to research managers, funding bodies and government agencies, who may want to find out about the performance of specific individuals and groups, and compare them with their peers both at national and international level.

There are many tools and services currently available, which already provide a wide variety of functionalities to support exploration of research data. These include bibliographic search engines, such as *Microsoft Academic Search* and *Google Scholar*; large research databases, such as *Sciverse Scopus*, *DBLP* and *PubMed*; reference

management applications, such as *Mendeley*; visual analytics tools, such as *CiteSpace*; tools which focus on mining and visualizing relations between researchers, such as *Arnetminer*; and many others¹. Nevertheless, as Dunne et al. point out [1], there is still a need for an *integrated solution*, where the different functionalities and visualizations are provided in a coherent manner, through an environment able to support a seamless navigation between the different views and functionalities. In addition, we would also argue that there are a number of important functionalities, relevant to the process of making sense of research data, which are currently not well supported. For instance, as discussed in our companion paper accepted for the ISWC 2012 research track [2], semantic relations exist between research areas, which help to structure the data space and make it possible to go beyond visualizations and searches based on a purely syntactic analysis of the data. Let's consider the Semantic Web again as an example. If our aforementioned PhD student is browsing papers related to this area, she may not be necessarily only interested in papers explicitly labeled "Semantic Web", but, e.g., she may also want to consider papers in Ontology Engineering or Linked Data, even though such papers may not be explicitly tagged as Semantic Web papers. Hence, environments for exploring research data need to make use of algorithms, such as the one described in [2], which can automatically discover relations between research areas and make it possible to go beyond purely syntactic approaches to search, while at the same time also addressing the limitations associated with manually constructed taxonomies [2].

Another weakness of current solutions concerns the limited support for identifying and visualizing relations between researchers. These are arguably crucial to the research sensemaking process, because the different ways groups of researchers co-operate, follow similar research trajectories through different topics, and exhibit other kinds of common patterns in the evolution of their careers and publishing behaviours, arguably provide key indicators of the dynamics of a research area. For instance, it may be very useful for a PhD student to be aware that a significant group of researchers has moved over the past 5 years from topic X to topic Y, exhibiting similar publishing behaviours, while not necessarily collaborating explicitly. While some existing systems already provide different ways of visualizing relations between researchers, these tend to cover simple 'static' ones, such as co-authorship.

In sum, it is our view that there is a need to develop new solutions for exploring research data, addressing the two issues discussed above: i) the need for a seamless integration of views and functionalities in the exploration process and ii) the need for new advanced functionalities, able to go beyond the 'document search' paradigm underlying most existing solutions, to provide new ways to discover patterns and relations between the different classes of entities in the research data space.

2 Making Sense of Research with Rexplore

The semantic relationships among authors and topics are at the heart of many new functionalities of Rexplore. In particular they are used for 1) computing novel kinds of

¹ In this short paper it is not possible to do justice to the huge variety of relevant work, hence we only list a few of the best known solutions. It is also important to note that the above classification is only approximate. In practice many tools integrate different functionalities –e.g., most bibliographic search engines and databases also provide visual analytics functionalities.

similarities and ranking metrics that take in consideration the semantic characterization of research areas; 2) improving the ability of Rexplore to interpret user queries; and 3) enabling a novel graph-based navigation technique, which combines both semantic relationships and automatically computed metrics to generate links between the elements of the domain.

Currently, the following functionalities and visualizations are provided²:

- **Author Ranking and Activity.** Author ranking is a standard functionality, which is provided by most systems and, likewise, Rexplore provides a wide variety of ranking mechanisms, including h-index, citations, publications, etc. These rankings can be parameterized with respect to career stage, calendar years, and publication venues, thus providing the user with fine-grained control over the visualizations. For example, not only Rexplore makes it possible to rank Semantic Web authors by number of publications – a functionality already provided by many existing tools, but it also makes it possible, for example, to focus on the ranking of the best early-career researchers over the past n years, taking into account only data related to the top publication venues in the Semantic Web. This is particularly useful in scenarios, such as recruitment, where the focus tends to be on people who are at a specific career stage. Rexplore also makes it possible to plot the impact of an author over time, both in absolute terms and relatively to the default standard for a particular area. Multiple integrated visualizations of an author’s activity are also provided, including the ability to visualize her citations or publications over time, and to parameterize these with respect to the relevant topics.
- **Relations between Authors.** Rexplore makes it possible to visualize a variety of relations between authors, most of which are dynamically constructed on the basis of the patterns emerging from their publishing behaviour and impact over time. For example, Rexplore makes it possible to cluster together researchers who exhibit similar publishing and impact trajectories, whether in the same or different fields. In addition, it is also possible to visualize similarity relations between authors who follow the same research path, by looking at the similarities between their research interests over time. Here we make use of the *Klink* algorithm [2], which ensures that the matching between research areas is ‘semantic’, rather than simply based on keyword matching. This solution is actually very generic and can also be used by applications in other domains, which wish to consider semantic relations when calculating similarity metrics.
- **Topic Evolution.** Rexplore provides a variety of ways to support a user’s understanding of the dynamics of a research area. For example, it makes it possible to visualize *migration patterns* across areas, thus allowing users to understand where people working in a new area are coming from, and whether an area is growing or reducing –i.e., whether there is a gain or loss of researchers between two areas. Another view shows the evolution of a topic over time, highlighting, for example, the main sub-topics, identified automatically using the *Klink* algorithm, which are emerging, as well as those which are decreasing in importance.

² While the ultimate aim of this work is to provide a comprehensive set of functionalities, covering a wide range of entities relevant to the research space, the current version of Rexplore (*alpha v0.9*) only covers authors, groups (of authors), topics, and publications.

Rexplore is implemented mostly in PHP and the visual part of the application uses JavaScript to ensure we do not depend on any external plugin. In particular we use the *Highcharts* library for the charts and a modified version of *JavaScript InfoVis Toolkit* for the graphs. The metadata we use come mainly from Microsoft Academic Search (<http://academic.research.microsoft.com/>) and DBLP (<http://www.informatik.uni-trier.de/~ley/db/>). The first comprises over 30 million papers, while the latter is a database for computer science that covers more than two million articles. As of August 2012, Rexplore contains the metadata regarding 15 million papers, focusing in particular on the Computer Science area. These data are enriched by means of a number of algorithms, which are able to infer new information –e.g., by discovering similarities and patterns in the data, by creating links between research topics, etc.

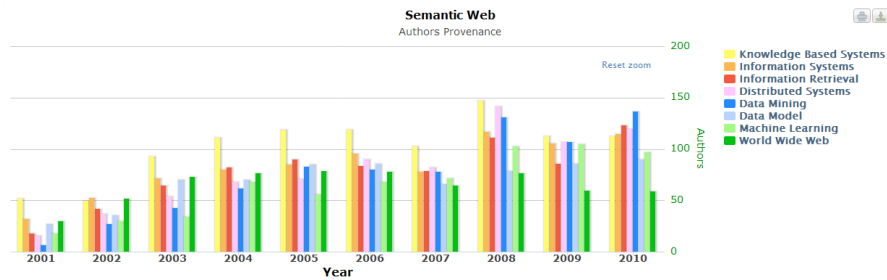


Fig. 1. One of the many visualizations provided by the current version of Rexplore. The snapshot shows the main research areas from which Semantic Web authors originated in the past decade. The figure shows that over the years most newcomers have come from the Knowledge Based Systems area, up until 2010, when for the first time most new authors came from Data Mining.

3 Conclusions

The current version of Rexplore already provides an array of interesting functionalities, many of which go well beyond what is available in other current tools. Nevertheless, we are still at a relatively early stage and many more functionalities are planned. In particular we plan to improve substantially the look and feel of the system, which currently is very much ‘browser-like’. We will also extend the range of inputs to the system, by adding information gathered from social networks and other web sources and we also plan to add geographic visualizations to create maps of research groups and topic tendencies. Finally we are also working on improving interactivity and customization, with the aim of allowing users to customize the topic structures generated by Klink, as well as other aspects of the system.

References

1. Dunne, C., Shneiderman, B., Gove, R., Klavans, J., and Dorr, B. (2012). Rapid Understanding of Scientific Paper Collections: Integrating Statistics, Text Analytics and Visualization. To appear in *JASIST*, 2012.
2. Osborne, F. and Motta, E., (2012). Mining Semantic Relations between Research Areas. *11th International Semantic Web Conference (ISWC 2012)*. Boston, MA.

Quest: Efficient SPARQL-to-SQL for RDF and OWL

Mariano Rodríguez-Muro, Josef Hardi and Diego Calvanese

KRDB Research Centre, Free University of Bozen-Bolzano
{rodriguez, josef, calvanese}@inf.unibz.it

Motivation. One of the most important uses of semantic technology is that of Ontology Based Data Access (OBDA), where the objective is to use shared vocabularies and ontologies as means to access data living in possibly disperse and heterogenous data sources (e.g., relational DBMS, XML databases, spreadsheets, etc.) Today this task often involves an ETL process in which the data is (E)xtracted from the source, (T)ransformed into RDF or OWL datasets in the target vocabulary and (L)oaded into a SPARQL endpoint or an OWL reasoner. This process carries all the issues that come with e.g., the need for synchronisation mechanisms to keep data up-to-date, the extra cost in time and space due to the duplication process, the additional software complexity at the client side, etc. Often it would be better to have live access to the original sources to avoid these issues and to be able to exploit any kind of optimisations that the original source can offer. In the context of relational DBMS and SPARQL queries, there exist several systems that allow for this on-the-fly approach, e.g., the D2RQ engine, Triplify, Spyder, Virtuoso RDF views, etc. However, often these systems fall short either in support for semantics (entailment regimes) and/or in query answering performance, e.g., the systems may send multiple queries to the sources and perform operations in-memory or they may generate complex SQL queries that cannot be planned and executed efficiently by the DBMS. This reality forces the use of the ETL approach, sometimes even in use cases in which an on-the-fly approach would be evidently possible.

Quest. In this demo we introduce Quest [7], a new system that provides SPARQL query answering with support for OWL 2 QL and RDFS. Quest allows to link the vocabulary of an ontology to the content of a relational database through *mapping axioms*. These are used together with the ontology to answer a SPARQL query by means of a single SQL query that is then executed over the database. Quest uses highly-optimised query rewriting techniques to generate the SQL query which not only takes into account the entailments of the ontology and data, but is also 'lean' and simple so that it can be executed efficiently by any SQL engine. Quest supports commercial and open source databases, including database federation tools like Teiid to allow for Ontology Based Data Integration of relational and other sources (e.g., CSV, Excel, XML). Now we will briefly describe Quest's mapping language, the query answering process and the most relevant optimisation techniques used by the system. We will conclude with a brief description of the content of this demo.

Quest Mappings. Quest offers a powerful mapping language that often compensates for the lack of expressivity of RDFS and OWL 2 QL and that enables use cases that are often tackled with OWL 2 EL, OWL 2 RL or SWRL. Mappings axioms are the means to relate the content of a database to the vocabulary of an ontology. Quest's mappings have their formal foundation on GAV-sound mappings as defined in [4],

however, extended to support all the features required from mapping language oriented towards the Semantic Web, e.g., patterns for URI construction, OWL/RDF datatypes, class and property mappings, URI and data constants in the mappings, etc. The result is a language that is equivalent in expressivity to the new W3C recommendation R2RML, but with a softer learning curve due to its simple and flexible Turtle-based syntax.

During the demo we will show the most relevant features of this language by means of a scenario based on the SQL version of IMDB and the Movie Ontology (MO) [3]. In the scenario, we want to query IMDB's data using SPARQL and MO's vocabulary and semantics. The IMDB database is very large and an ETL approach would be costly, i.e., 21 tables containing 43 million rows that generate approx. 42 million triples. Instead, we show how using Quest such scenario is realizable efficiently in an on-the-fly manner using query rewriting and mappings as the following:

```
tgt imdb:movie/{$id} a Movie; title $title;
    dbpedia:productionStartYear $production_year^^xsd:integer .
src SELECT id, title, production_year FROM title
    WHERE kind_id = 1

tgt imdb:movie/{$t.id} belongsToGenre Romance.
src SELECT t.id FROM title t, movie_info m WHERE t.id = m.movie_id
    AND m.info_type_id = 3 and m.info='Romance'
```

In these two mappings we can see that a mapping is composed by two parts, the *source* (**src**), a SQL query that brings some data from the database and a *target* (**tgt**), a template that, intuitively, indicates how to create ABox assertions/data-triples by replacing the column reference in the target with the actual values returned by the source query (in a similar fashion to CONSTRUCT queries in SPARQL). Note that the language offers complete freedom on the SQL queries, the templates, as well as in the way in which the object URIs are constructed. Instances declared in the ontology (e.g., *Romance*) can also be used in mappings, giving the possibility of hybrid data graphs, in which the dynamic and larger part of the data lives in the database and a static and small part is given in the ontology (e.g., in our scenario the ontology states '*Romance* rdf:type *Love*'). Last, the familiar Turtle-based syntax for mappings exposes the semantics of the mappings to the user in a simple way, and doesn't contain any implementation specific features that are often exposed in languages like D2R and R2RML.

SPARQL query answering in Quest. The main service in Quest is SPARQL query answering by means of SQL query rewriting. During query answering Quest will take into account the semantics of the ontology vocabulary (defined in OWL 2 QL or RDFS axioms) as well as the mappings and the metadata of the database. The result of this process is one single SQL query that is executed over the original source. For example, consider the following SPARQL query over the MO ontology asking for

"the name of all directors that are also actors in the cast of a movie directed by themselves such that the movie was produced between the years 2000 and 2001 in Eastern Asia, the movie is a romantic movie and has a user rating higher than 7"

In our use case, we would be able to get this information using the following SPARQL query:

```

SELECT $name $title ?rating WHERE
{ $m a Movie; title ?title; hasActor ?x; hasDirector ?x;
  isProducedBy $y; belongsToGenre $z; dbpedia:rating ?rating;
  dbpedia:productionStartYear ?year.
  $x dbpedia:birthName $name .
  $y hasCompanyLocation [ a Eastern_Asia ] .
  $z a mo:Love .
FILTER (?rating > 7.0 && ?year >= 2000 && ?year <= 2001) }

```

given the axioms in MO and the mappings to the IMDB SQL database Quest would compute the following SQL query:

```

SELECT V8.name AS name, V0.title AS title, V7.info AS rating
FROM title V0, cast_info V1, cast_info V3, company_name V4,
  company_name V4, movie_companies V5, movie_info V6,
  movie_info_idx V7, name V8, movie_companies V9, movie_info V10
WHERE V0.id = V1.movie_id          AND V0.id = V3.movie_id
  AND V0.id = V5.movie_id          AND V0.id = V6.movie_id
  AND V0.id = V7.movie_id          AND V0.id = V10.movie_id
  AND V4.id = V5.company_id         AND V4.id = V9.company_id
  AND V1.person_id = V3.person_id  AND V1.person_id = V8.id
  AND V0.kind_id = 1                AND V3.role_id = 8
  AND (V1.role_id = 1 OR V1.role_id = 2)
  AND (V4.country_code = '[cn]' OR V4.country_code = '[jp]')
  AND V5.company_type_id = 2        AND V6.info_type_id = 3
  AND V7.info_type_id = 101         AND V9.company_type_id = 2
  AND V10.info_type_id = 3          AND V10.info = 'Romance'
  AND V7.info > '7.0'               AND V0.production_year >= 2000
  AND V0.production_year <= 2001

```

note that in MO there is a class (resp. property) hierarchy bellow the class Eastern_Asia (resp. the property hasActor) and hence, Quest includes appropriate OR statements for the values of some of the columns to account for these semantics and in accordance to the mappings of the system.

Inspecting the resulting SQL query some of the benefits of using Quest can be noted. First, as with other OBDA systems, posing the query in terms of the ontology vocabulary is more natural, and follows almost the same logic as the query in natural language. Learning to formulate this query takes a matter of minutes after a fast inspection of the vocabulary, however, formulating such an SQL query for the IMDB database requires a considerable effort since the schema is very complex. With Quest the complexity of the underlying database is hidden and handled by the system. Second, the SQL query generated by Quest is very close to, and often coincides with the query that would be created by an SQL expert. In general, this is not the case in other OBDA/RDB2RDF systems which often issue not one query to the DBMS, but multiple queries that bring data back and forth to compute the answer locally, or in few more advanced system where a single SQL query is achieved, the query is often too complex to be executed efficiently or does not take into account the semantics of the ontology. These are not issues in Quest which is often able to offer dramatic performance improvements w.r.t. to other systems (e.g., on hardware found in an average laptop the previous query takes 1.5s in Quest, while D2RQ cannot return an answer after 5 minutes).

Optimisation. In order to generate these queries **Quest** resources to many optimisation techniques at different levels of the query answering process. Some of these techniques are aimed at efficiently handling the semantics of the ontology while others at generating optimal SQL. Here we briefly mention the major ones. During system initialisation, **Quest** will optimise both TBox and mappings. With respect to the mappings, **Quest** will first parse the SQL queries in the mappings and analyse the database metadata, extracting constraints (e.g., Primary Keys, etc.) and using this information, together with *query containment* checks to optimise the mappings (see T-mappings [5]). The TBox will also be optimised, eliminating redundant vocabulary and detecting any redundancy w.r.t. the mappings (see [6]). At query time, the system will go through 4 key steps: (i) SPARQL-to-Datalog translation, (ii) query rewriting w.r.t. the TBox, (iii) computation of a *partial evaluation* based on the mappings and finally (iv) SQL generation and execution. At each step, **Quest** will resource to query containment-based optimisations to eliminated redundant queries or optimise individual queries (e.g. eliminating redundant joins). As can be intuited, our optimisation theory relies heavily on our ability to analyse SQL to understand its semantics and on query containment for redundancy detection; because of this, as our implementation of the related algorithms improve so will the quality of the SQL generated by the system.

Content of the Demo. We will present the major features of **Quest** using the full version of the IMDB scenario introduced here. We will show how mappings can be created using our plugin for Protege, **-ontopPro-**, and we will query **Quest** through the same interface. We will discuss the features of the mapping language as well as the optimisations that **Quest** performs at the different stages of query answering. Queries will be run on top of a local copy of the IMDB database running on a PostgreSQL server, allowing us to show the performance of the system on inexpensive hardware and to compare it to other similar platforms. Last, we will show how using a data federation tool like Teiid, we can integrate multiple and heterogeneous sources using the same techniques. **Quest** is part of the **-ontop-** framework for Ontology Based Data Access (OBDA) and can be downloaded from its website [2]. A video with a preview of the demo is available online [1].

References

1. **-ontop-** demo. <http://obda.inf.unibz.it/ontop/iswc12/>.
2. **-ontop-** website. <http://obda.inf.unibz.it/protege-plugin/>.
3. A. Bouza. MO the movie ontology, 2010. [Online; 26. Jan. 2010].
4. M. Lenzerini. Data integration: A theoretical perspective. In *PODS*, pages 233–246, 2002.
5. M. Rodríguez-Muro and D. Calvanese. Dependencies: Making ontology based data access work in practice. In *Proc. of AMW 2011*, 2011.
6. M. Rodríguez-Muro and D. Calvanese. High performance query answering over dl-lite ontologies. In *Proc. of the 13th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2012)*, pages 308–318, 2012.
7. M. Rodríguez-Muro and D. Calvanese. Quest, an OWL 2 QL reasoner for Ontology-Based Data Access. In *Proc. of the 9th Int. Workshop on OWL: Experiences and Directions (OWLED 2012)*, volume 849 of *CEUR Electronic Workshop Proceedings*, <http://ceur-ws.org/>, 2012.

A Prototype for Semantic based Diagnosis of Road Traffic Congestions

Marco Luca Sbodio, Freddy Lecue, Anika Schumann

IBM Research - Ireland
Damastown Industrial Estate, Dublin, Ireland
{(firstname.lastname)}@ie.ibm.com}

Abstract. Retrieving the causes of road traffic congestions in quasi real-time is an important task that will enable city managers to get better insight into traffic issues and thus take appropriate corrective actions in a timely way. Our work, accepted at ISWC 2012, tackles this problem by integrating and reasoning over a variety of heterogeneous data sources including data streams. In this paper we present an initial prototype of our work for the city of Dublin, Ireland.

1 Introduction

Consider the case of city planning in anticipation of large events (for a example Republic of Ireland World Cup qualifier match in Croke Park, Dublin), or in reaction to unplanned events (for example a mob assembling in the Dublin Docklands area). By integrating and correlating partial observations from multiple data sources, we could infer that bad weather, coupled with a large number of people assembling in one area of the city on a normal working day, coupled with a lack of public parking, led to traffic chaos that was widely reported in the media, driving strong negative sentiment towards the handling of such events. Whilst such an analysis is a useful tool for understanding “*what went wrong*” and “*what were the causes*” after the event, our work is the first one that can **compute causes** of such unexpected situations **in quasi real-time**; other works focus on detecting, visualizing and analyzing traffic congestions [1]. We achieve this by exploiting semantic representations of historical data (such as traffic congestions data) and feeding them into an AI diagnosis approach. The work is described in a paper accepted at the In-Use track of ISWC 2012 [2].

In this paper we present an initial prototype¹ of our approach that we have developed for the city of Dublin and describe the data sets that we have semantically encoded.

2 Diagnosing Road Traffic Congestions

We start by briefly describing our diagnosis approach [2] shown in Figure 1. First, its heterogeneous input data (see next section for their details) are integrated using semantic web technologies. This then allows AI diagnosis techniques to compute off-line a

¹ see video at <http://www.youtube.com/watch?v=xT5dPpnayZI>

diagnoser representing *historical observations* over a *time window* and their *explanations* (for example Canal street was congested in 2012, May 1st at 6:00pm because of a concert event in Aviva stadium and road works in Bath avenue). Finally, quasi real-time diagnosis consists in combining semantic matching and AI diagnosis techniques for (i) retrieving “*similar*” causes (e.g., roads with heavy traffic of same duration) with “*similar*” conditions (e.g., nearby sport events) which have appeared in the past and (ii) interpreting them in the real-time context.

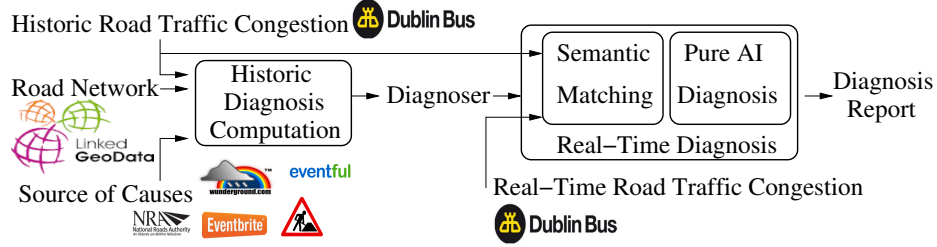


Fig. 1. Overview of the Semantics-Augmented Diagnosis Approach.

3 Data Sets

Table 1 lists the heterogeneous data sets considered in our scenario. Most of these data sources are public, and many are provided by Dublin City Council through *dublinked.ie*² web site, and hosted at IBM.

The **Dublin Bus Data**, encoded according to the SIRI standard (note *a* in Table 1), is transformed into a real-time stream (information about 1000 buses updated every 20 seconds) that is persisted into CSV files (one file per day). Each SIRI record (line in a CSV file) contains information about the current position (latitude and longitude) of a vehicle, its line number, its direction along the bus line, if the bus is in congestion, and if the bus is at a stop point. Information about bus lines and stops is given separately through other CSV files (such information is static, or at least it changes very rarely). We have developed a simple \mathcal{EL}^{++} ontology to represent SIRI data. The actual SIRI records (lines from the CSV files) are modeled as instances of the class *VehicleAtomicUpdate*, which has a property for each field of a SIRI record. Based on a history of 217 SIRI data files (approximately 26 GB), referring to 217 days in 2011 (approximately 122 MB a day), 44.7% of the SIRI records generates 8 triples/record and 47.2% generate 11 triples/record (1 triple to define the type of the RDF resource, and 10 triples to specify the SIRI properties); the other records generate either 9 or 10 triples/record. The varying number of triples per record is due to some missing fields. The instances of *VehicleAtomicUpdate* with missing properties are nevertheless useful to estimate the number of buses in a bounding box in a certain time window (latitude, longitude, and timestamp are always available in SIRI records).

City Events were captured through *Eventful* and *EventBrite* web sites (notes *e*, *f* in Table 1). An average of 187 events a day (i.e., same days as those captured for SIRI data) have been described using some LOD vocabularies e.g., DBpedia, Talis. In addition we

² <http://dublinked.ie/>

Table 1. Data Sets

Data Source	Provider	Format	Size
Dublin Buses Data Stream: vehicle data (GPS location, line number, delay, ...)	Dublin City Council (private)	SIRI ^a (XML)	4-5 GB/day
Wunderground for Dublin: real-time weather information	Wunderground ^b (public)	CSV	0.05-1.5 GB/day
Road & Weather Conditions	NRA ^c (public)	CSV	0.1 GB/day
Road Works & Maintenance	Dublinked ^d (public)	CSV	0.01 GB/day
Events in Dublin	Eventbrite ^e and Eventful ^f (public)	XML	0.001-0.05 GB/day
DBPedia	DBPedia ^g (public)	RDF	3.5×10^6 concepts
Dublin roads: list of road types, junctions and GPS coordinates	Linkedgeodata ^h (public)	RDF	0.1 GB

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

^b <http://www.wunderground.com/weather/api>

^c NRA - National Roads Authority <http://www.nratraffic.ie/weather>

^d <http://www.dublinked.ie/datastore/datastore.php>

^e <https://www.eventbrite.com/api>

^f <http://api.eventful.com>

^g <http://dbpedia.org>

^h <http://linkedgeodata.org>

enriched the events description with \mathcal{EL}^{++} GCIs to capture their categories, which are used for computing not only fine grained matching between historical and new events, but also for computing the diagnosis report. Each event has been described on average through 26 RDF triples.

Similarly an average of 51 **Road Works and Maintenance**³ records a day have also been enriched through 16 RDF triples each. An \mathcal{EL}^{++} enrichment of this raw data ensures that historical and new records can be matched for diagnosis and reporting purposes. We also injected 14,316 \mathcal{EL}^{++} GCIs (6 RDF triples each) to describe 4772 **Roads and their Interconnections**⁴.

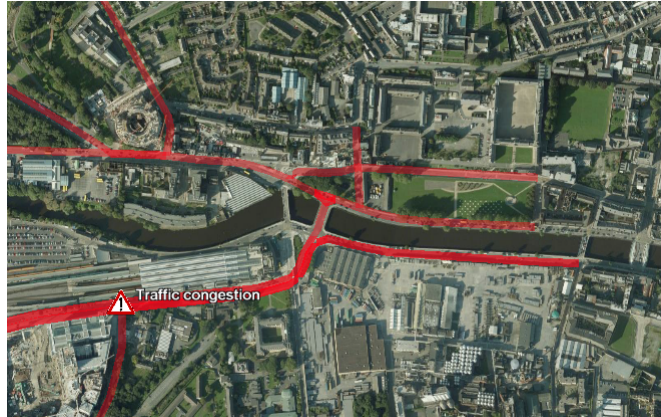
The **Core Static Ontology**, which is used for representing SIRI, events, road works, road weather and Dublin weather data, is composed of 67 concepts with 24 role descriptions (25 concepts subsume the 42 remaining ones with a maximal depth of 4). Finally, a **History of 217 days of the Traffic Congestion Information** was computed based on buses data streams (encoded by more than 1×10^9 RDF triples). Information about past events, road works, weather information and road conditions was stored as 1.1×10^6 RDF triples.

³ CSV sample in <http://www.dublinked.ie/datastore/metadata064.php>

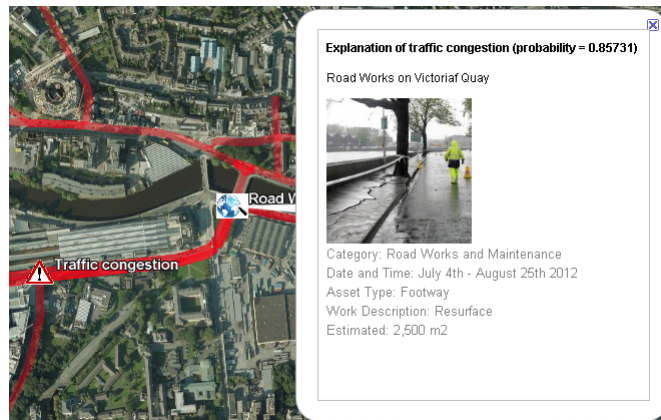
⁴ CSV sample in <http://www.dublinked.ie/datastore/metadata125.php>

4 Prototype

Our initial prototype analyses data of Dublin buses, and displays congested roads on an interactive map (see figure 2(a)). Based on our quasi real-time diagnosis component, the system displays also explanations of selected road traffic congestions (see figure 2(b)). The explanation contains both data about its accuracy, and information extracted from the data source of the identified cause (road maintenance in the case of figure 2(b)).



(a) Detection of traffic congestion



(b) Diagnosis of traffic congestion

Fig. 2. Screenshots of Automated Traffic Diagnosis Prototype

References

1. Biem, A., Bouillet, E., Feng, H., Ranganathan, A., Riabov, A., Verscheure, O., Koutsopoulos, H.N., Moran, C.: Ibm infosphere streams for scalable, real-time, intelligent transportation services. In: SIGMOD. (2010) 1093–1104
2. Lécué, F., Schumann, A., Sbodio, M.: Applying semanticweb technologies for diagnosing road traffic congestions. In: to appear in ISWC. (2012)

TwikiMe!

User profiles that make sense.

Patrick Siehndel, Ricardo Kawase

L3S Research Center, Leibniz University Hannover, Germany
{siehndel, kawase}@L3S.de

Abstract. The use of social media has been rapidly increasing in the last years. Social media, such as Twitter, has become an important source of information for a variety of people. The public availability of data describing some of these social networks has led to a great deal of research in this area. Link prediction, user classification and community detection are some of the main research areas related to social networks. In this paper, we present a user modeling framework that uses Wikipedia to model user interests inside a social network. Our model of user interests reflects the areas a user is interested in, as well as the level of expertise a user has in a certain field.

1 Introduction

In the last years, surfing on social network Web sites has become the most prominent online activity. Together, the top accessed social networks such as Facebook, Twitter and Myspace, to name a few, aggregate over a billion users. Given this level of activity, the research interest on the field of social networks has grown considerably. User modeling, link prediction, sentiment analysis, community analysis, sociology and many other areas of Web Science are examples of research fields exploiting the public (and private) data available in such networks.

In this paper we propose a semantic approach to generate user profiles based on the publicly available Twitter data. In other words, we generate a concise, yet descriptive semantic user profile using Twitter streams. With a semantically generated user profile, one can easily identify the exact topics of interest a user has. In contrast to bag of word approaches, we generate semantically enhanced user profiles that quantify the users' interests in a set of specific categories. Ultimately, our profiling method outputs a semantically enhanced user profile that reflects the real user interest (based on his explicit tweets). The TwikiMe! user profile is a reduced representation of the user interests on a 23-sized vector that is both human and machine comprehensible.

2 Twikifying

We provide TwikiMe! as a framework to generate semantically enhanced profiles for Twitter users. To accomplish it, we use the well established Wikipedia corpus as a semantic knowledge base. Wikipedia is arguably the most accessed reference Web site and each of the over 3.5 million existing articles are manually classified by human

curators to one or more categories. Additionally, categories are organized in a graph where sub-categories reference top level categories. The English Wikipedia has in total 23 top level categories (*Main topic classifications*), which we use to represent a user profile (See Figure 2 for the list of top-level categories). Abel et al. [2] presented similar strategies to enhance Twitter user profiles, however their topic-based profile is built upon topics related to different types of news events. In our work, we consider the topics (categories) of each detected Wikipedia entity, thus the categories describe a wider area of fields.

The creation of semantically enhanced user models consists of three stages (see Figure 1). The first stage, *Extraction*, entities are extracted from the user’s tweets. Given Twitter users streams, we annotate all tweets to detect any mention of entities that can be linked to Wikipedia articles. In this step, we chose to use the WikipediaMiner [1] service to annotate the users’ tweets.

The second stage, *Categorization*, we extract the categories of each entity that has been mentioned in the users Tweets. For each category, we follow the path of all parent categories, up to the root category. In some cases, this procedure results in the assignment of several top level categories to an entity. A weight is calculated for each category by first setting a value of 1 for the detected entity. Following the parent categories (which are closer the root category) we divide the weight of each node by the number of siblings categories, resulting in each entity receiving 23 categories’ scores. In the final stage, *Aggregation*, we perform a linear aggregation over all of the scores for a tweet in order to generate the final user profile.

Our approach of reducing the user profile to 23 topics differs significantly from the work of Michelson and Macskassy [4]. In their work, they propose a similar approach to annotate tweets with Wikipedia articles; but instead, of considering all parent categories, they traverse the category graph only “5 levels deep”. In doing so, they assume that a five stage traversal is sufficient to reach categories that are general enough for a user’s profile. The limitation of their assumption is that a user’s classification may have an unlimited number of categories, thereby preventing profiles from having a normalized length and comparison among all users.

We applied our user profiling method on the Twitter dataset created by Yang and Leskovec [5] containing 476 million tweets from 20 million users. The tweets were collected between 1st of June 2009 and 31st of December 2009. We combined with the social Graph created by Kwak et al. [3] with 41.7 million user profiles and 1.47 billion social relations. We randomly selected 20,000 users with their followers and friends for a total of 630,000 users and 28 million tweets and analyzed how similar the users are to their followers and friends

Additionally, we calculated how much the user profile changes over time by splitting the user’s tweets into two bins and calculating the inter-user similarity. To perform such evaluation we divided the users’ stream in half (by number of tweets), comparing the profile of the earlier tweets against the latest ones. Table 1 shows the results of the similarity measures. As metric we used the cosine similarity.

Our result show that while the categories related to the tweets of one user stay the same over the time (leading to a very high similarity of 0.85), we see that the average similarity between the user and her friends and followers is rather low with 0.26 and

Table 1. Cosine Similarity of the generated user profiles

Comparison	User - User	User - Followers	User - Friends	Friends - Followers
Cosine-Similarity	0.85	0.27	0.26	0.24

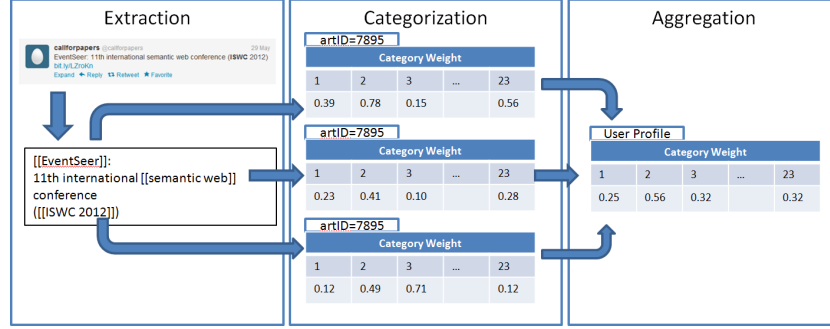


Fig. 1. Different stages for the creation of userprofiles

0.27. Comparing the profiles of the followers and friends we obtained a similarity of 0.24. Based on the high similarity when comparing only tweets from one user, we assume that the topics one user tweets about stays rather constant, and are not necessarily the same as the topics discussed by friends and followers.

3 TwikiMe! Online

To illustrate our approach, we deployed an online version¹ of the TwikiMe! framework. The interface guides the user through the twikifying process. First, one must provide a valid Twitter user name and the amount of past tweets to be annotated (due to the outsourcing of WikipediaMiner service, this step is the most time consuming). Once the process is finished, the user is asked to trigger the Categorization step. Finally, the user is presented with the TwikiMe! chart of the given Twitter user.

To address qualitative comparisons, we also provide a comparative graphical interface as depicted in Figure 2. In the comparative chart, it is possible to visualize the percentage results for two distinct users, side-by-side. Additionally, the interface provides a date picker to restrict the comparisons to a given time interval. In Figure 2, one can see the comparison between the Twitter accounts of the Dalai Lama (@DalaiLama) and president Barack Obama (@BarackObama), during the period of April through December 2011. The resulting user profile depicted in the chart shows a clear dominance of Barack Obama in the topics of "people", "politics" and "business", while Dalai Lama leads on "life", "culture" and "geography" topics.

A closer look into different "known" users (politicians, researchers and technology news), shows that the resulting TwikiMe! profiles provide expected topic identities. While technology news providers lead on the topics of "business", "society" and "technology", researchers lead on the topics of "education" and "applied sciences". We invite

¹ <http://twikime.l3s.uni-hannover.de>

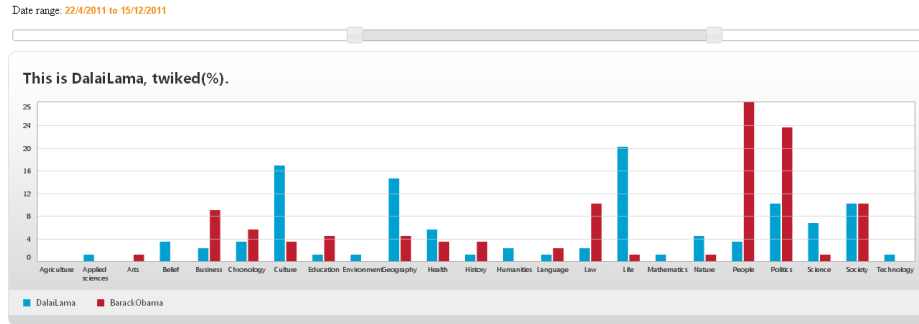


Fig. 2. TwikiMe comparison charts showing the profile of Dalai Lama and Barack Obama.

the readers to try it by themselves our TwikiMe! prototype at <http://twikime.l3s.uni-hannover.de>. The prototype makes available all previously *twikified* users for comparisons.

4 Conclusion

In this paper, we introduced the TwikiMe! A prototype for generating comprehensible user profiles that are represented compactly as a 23-length vector. The profiles quantifies the users’ implicit interest in each of these different categories. We believe that, by exploiting the TwikiMe! profiles, we are able to improve content and user recommendation on Twitter.

As future work, we plan to empirically demonstrate the proof of concept of the generated user profiles as well as improving the strategy of the Categorization step. Finally, we believe that there may be a significant difference between what a user “produces” and what a user “consumes”. We also plan to evaluate this difference by generating TwikiMe! user profiles based on tweets that a user follows, rather than just the tweets that a user writes.

References

1. Wikipediaminer toolkit. website, 2012. <http://wikipedia-miner.cms.waikato.ac.nz/> (accessed july 30, 2012)., 2011.
2. F. Abel, Q. Gao, G.-J. Houben, and K. Tao. Analyzing User Modeling on Twitter for Personalized News Recommendations. In *International Conference on User Modeling, Adaptation and Personalization (UMAP), Girona, Spain*. Springer, July 2011.
3. H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web, WWW ’10*, pages 591–600, New York, NY, USA, 2010. ACM.
4. M. Michelson and S. A. Macskassy. Discovering users’ topics of interest on twitter: a first look. In *Proceedings of the fourth workshop on Analytics for noisy unstructured text data, AND ’10*, pages 73–80, New York, NY, USA, 2010. ACM.
5. J. Yang and J. Leskovec. Patterns of temporal variation in online media. In *Proceedings of the fourth ACM international conference on Web search and data mining, WSDM ’11*, pages 177–186, New York, NY, USA, 2011. ACM.

Adding Realtime Coverage to the Google Knowledge Graph

Thomas Steiner^{1*}, Ruben Verborgh², Raphaël Troncy³,
Joaquim Gabarro¹, and Rik Van de Walle²

¹ Universitat Politècnica de Catalunya – Department LSI, Spain
`{tsteiner,gabarro}@lsi.upc.edu`

² Ghent University – IBBT, ELIS – Multimedia Lab, Belgium
`{ruben.verborgh,rik.vandewalle}@ugent.be`

³ EURECOM, Sophia Antipolis, France
`raphael.troncy@eurecom.fr`

Abstract. In May 2012, the Web search engine Google has introduced the so-called Knowledge Graph, a graph that understands real-world entities and their relationships to one another. Entities covered by the Knowledge Graph include landmarks, celebrities, cities, sports teams, buildings, movies, celestial objects, works of art, and more. The graph enhances Google search in three main ways: by disambiguation of search queries, by search log-based summarization of key facts, and by explorative search suggestions. With this paper, we suggest a fourth way of enhancing Web search: through the addition of realtime coverage of what people say about real-world entities on social networks. We report on a browser extension that seamlessly adds relevant microposts from the social networking sites Google+, Facebook, and Twitter in form of a panel to Knowledge Graph entities. In a true Linked Data fashion, we inter-link detected concepts in microposts with Freebase entities, and evaluate our approach for both relevancy and usefulness. The extension is freely available, we invite the reader to reconstruct the examples of this paper to see how realtime opinions may have changed since time of writing.

1 Introduction

With the introduction of the Knowledge Graph, the search engine Google has made a significant paradigm shift towards “*things, not strings*” [3], as a post on the official Google blog states. With the graph, Google now publicly acknowledges efforts in the direction of understanding the difference between queries like “giants” for the football team (New York Giants) and “giants” for the baseball team (San Francisco Giants). Users now get word sense disambiguation support that allows them to steer the search engine in the right direction.

Google News is a news aggregator portal provided and operated by Google. Earlier this year, a new feature called *realtime coverage* was added to the portal [6], providing signed in US Google+ users with the latest relevant posts from the Google+ community for news stories.

* Full disclosure: the author is also a Google employee.

We have implemented a social network-agnostic approach to add realtime coverage to Knowledge Graph results via a browser extension. The extension is freely available on the Chrome Web Store¹; a free Google API key is required.

2 Methodology

Chrome extensions are small software programs that users can install to enrich their browsing experience. Extensions can inject and modify the contents of Web pages via so-called *content scripts*. We have implemented an extension that gets activated when a user uses Google to search the Web. If the extension detects that the current search engine results page (SERP) has an associated real-world entity in the Knowledge Graph, it first extracts the entity’s name and its URL. The URL typically points to a Wikipedia page of the real-world entity. In some cases—depending on the data source—it can also point to a Google Books page or other resources. Second, the extension performs full-text searches via the search APIs of the popular social networking sites Google+, Twitter, and Facebook and returns the top n results (configurable) of each social network. Third, all microposts get analyzed and annotated via part-of-speech tagging. For each (pair of) nouns, the Freebase search API is used to link these strings with entities, a task commonly known as *entity linking* [4]. In future, the Freebase-linked entities can be checked for relevancy by measuring their relatedness to the extracted Knowledge Graph concept, uniquely identified by its URL. For now, we follow a context-free, most frequent sense approach, which, according to Agirre and Edmonds [1], serves as a surprisingly strong baseline, an observation that we can confirm in our evaluation of micropost *relevancy* in Section 3. In a final step, the resulting set of microposts is ordered chronologically and attached in form of a separate panel to the Knowledge Graph result. A screenshot with exemplary extension output for the entity “*Isabella Stewart Gardner Museum*” is online².

3 Evaluation

For the evaluation, we have considered 100 microposts from 94 unique users, out of which 72 microposts contained outbound links to overall 94 Web pages. We asked a not involved student evaluator to evaluate the extension on 21 real-world entities, starting from the concept “*Boston, MA*”, and then recursively following graph links to related concepts. Our evaluation criteria were *usefulness* of the information contained in both the microposts and the potentially linked-to Web pages, and the *relevancy* of the microposts to the real-world entities in question. We consider a micropost *useful* if it provides additional information on an entity that was not covered by the already existing Knowledge Graph facts, like, *e.g.*, restaurant tips for a location type entity. A micropost is considered *relevant* if the human evaluator could draw a connection to the current real-world entity, like, *e.g.*, a link to a recent news article about a celebrity type entity. We have calculated the Mean Opinion Score (MOS) [2] for both criteria.

¹ Knowledge Graph Socializer extension (<http://goo.gl/rzdVK>)

² Screenshot of the extension (<http://twitpic.com/a8zgiq/full>)

Traditionally, MOS is used for conducting subjective evaluations of telephony network transmission quality. More recently, MOS has also found wider usage in the multimedia community for evaluating *per se* subjective things like perceived quality from the users’ perspective. Therefore, a set of standard, subjective tests are conducted, where users rate the quality of test samples with scores from 1 (worst) to 5 (best). The actual MOS is the arithmetic mean of all individual scores. Given our intrinsically subjective evaluation criteria, MOS provides a meaningful way to judge the quality of our approach. For *relevancy*, the MOS was 4.38. For *usefulness*, we have calculated a MOS of 3.75. Our complete evaluation is available online³. We provide an interpretation of the results below.

Usefulness: The MOS of 3.75 suggests potential for improvement, albeit the information from the microposts and linked-to Web pages was overall still considered useful. Positively rated revealed insights were, among others, recommended restaurants, suggested things to do, scheduled future or past events, special (not necessarily advertisement-like) offers, user-generated photos, news articles, travel tips, or stories of everyday life. On closer inspection, the microposts that teared down the *usefulness* MOS were in the majority of cases still rated relatively high for *relevancy*. We could track down the relevant but not useful microposts to three categories: (i) long microposts (Google+, Facebook) that mentioned the concept somewhere, but that were too long to skim, (ii) so-called @Replies⁴ on Twitter that are messages to other Twitter users, but that lack the context of the conversation, and finally (iii) so-called native check-in messages on Google+, together with shared Foursquare check-in messages on Twitter and Facebook via connected user profiles⁵ that are geotagged (and thus relevant), but that provide no other information besides the fact that a user was at a certain place. By disregarding all three types of microposts, the MOS could be increased to 3.94.

Relevancy: The high MOS of 4.38 shows that the microposts were in the majority of cases considered very relevant. On the one hand, this is due to the well-chosen titles of concepts in the graph, which oftentimes are either unique enough (e.g., “*Faneuil Hall*”), or on the other hand, include some sort of disambiguation aid (e.g., “*Museum of Science, Boston*”). In [4], Spitzkovsky and Chang have collected an extensive set of link titles for Wikipedia concepts and shown that an entirely context-free approach to link strings with concepts works consistently well. Together with the observation above, this justifies our concept title-based full-text search approach on social networking sites, as reflected by the MOS.

4 Future Work and Conclusion

Each concept in the Knowledge Graph has a unique identifier⁶. Exploiting this fact and considering that Knowledge Graph results have already been interlinked via `owl:sameAs` with Freebase entities⁷, we can imagine a comments archive of

³ Evaluation (<http://goo.gl/dbvr4>)

⁴ “What are @Replies and Mentions?” (<http://goo.gl/Ge2RG>)

⁵ “Connecting/sharing to Facebook and Twitter” (<http://goo.gl/NBuCr>)

⁶ A. Thalhammer to the `semantic-web@w3.org` mailing list (<http://goo.gl/2m0zJ>)

⁷ H. Glaser to the `semantic-web@w3.org` mailing list (<http://goo.gl/jj6Sg>)

things people said about real-world entities. Taking it one step further, we can think of not only archiving user comments, but also adding meaning to them [5], which would allow for interesting use cases such as searching for places that are attended by celebrities.

Concluding, we have shown in this paper how realtime coverage can be added to Knowledge Graph results via a browser extension. We have evaluated both the *usefulness* and *relevancy* of the returned microposts and linked-to Web pages, and have given an outlook on how via a potentially enriched comments archive about real-world entities, further use cases can be covered. Through this work, we have demonstrated how the rather static, structured world of real-world facts from the Knowledge Graph (with facts like foundation date, opening hours, spouse, etc.), can be successfully combined for fun and profit with the rather dynamic, *a priori* unstructured world of social network microposts.

Acknowledgments

The research leading to this paper was partially supported by the European Commission via the FP7 projects I-SEARCH (GA 248296) and LinkedTV (GA 287911). R. Verborgh is funded by Ghent University, the Interdisciplinary Institute for Broadband Technology (IBBT), the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT), the Fund for Scientific Research Flanders (FWO Flanders), and the European Union. J. Gabarró is partially supported by SGR 2009–2015 (ALCOM) and TIN-2007–66523 (FORMALISM).

References

1. E. Agirre and P. Edmonds. *Word Sense Disambiguation: Algorithms and Applications*. Text, Speech and Language Technology. Springer, 2007.
2. International Telecommunication Union, Telecommunication Standardization Sector. ITU-T Recommendation P.800: Methods for Subjective Determination of Transmission Quality, Aug. 1998. <http://www.itu.int/rec/T-REC-P.800-199608-I/en>.
3. A. Singhal. “Introducing the Knowledge Graph: things, not strings”, Official Google Blog, May 2012. <http://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html>.
4. V. I. Spitzkovsky and A. X. Chang. A Cross-Lingual Dictionary for English Wikipedia Concepts. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC’12)*, Istanbul, Turkey, May 2012.
5. T. Steiner, R. Verborgh, J. Gabarró Vallés, and R. Van de Walle. Adding Meaning to Social Network Microposts via Multiple Named Entity Disambiguation APIs and Tracking Their Data Provenance. *International Journal of Computer Information Systems and Industrial Management*, 5:69–78, 2013. http://www.mirlabs.org/ijcisim/regular_papers_2013/Paper82.pdf (Accepted for publication).
6. S. Zuccarino. “Updates to Google News US Edition: Larger Images, Realtime Coverage and Discussions”, Google News Blog, May 2012. <http://googlenewsblog.blogspot.com/2012/05/updates-to-google-news-us-edition.html>.

Everything is Connected: Using Linked Data for Multimedia Narration of Connections between Concepts*

Miel Vander Sande, Ruben Verborgh, Sam Coppens, Tom De Nies,
Pedro Debevere, Laurens De Vocht, Pieterjan De Potter,
Davy Van Deursen, Erik Mannens, and Rik Van de Walle

Ghent University - IBBT
Department of Electronics and Information Systems - Multimedia Lab
Gaston Crommenlaan 8 bus 201, B-9050 Ledeborg-Ghent, Belgium
`firstname.lastname@ugent.be`

Abstract. This paper introduces a Linked Data application for automatically generating a story between two concepts in the Web of Data, based on formally described links. A path between two concepts is obtained by querying multiple linked open datasets; the path is then enriched with multimedia presentation material for each node in order to obtain a full multimedia presentation of the found path.

1 Introduction

Interconnectedness is a major topic discussed in philosophy and physics, stating that everything in the world is connected in some way. A popular example is the Six degrees of Separation principle, introduced by S. Milgram [5]. It states that human networks grow in density despite the big physical distances, resulting in a maximal connection of five acquaintances between two people in the whole world. Between academia, this concept is often discussed for not being really scientifically proven [3]. However, with the Web of Data emerging, links between things become well defined and more known [2].

This demonstration relies on Linked Data to illustrate the above described principle (i.e. interconnectedness) by following machine-processable links within today's Web of Linked Data. The presented application will retrieve the connection (with intervening nodes) between two given concepts. Next, for each resource corresponding to a node in the found path, a media presentation is generated (e.g. use of a photo to illustrate a person). Finally, the found path is narrated through a full multimedia presentation, illustrating the intervening resources and their connections (i.e. the links). The aim of this demo is therefore threefold:

- bringing Linked Data closer to inexperienced users;
- a practical test of the link quality in today's Web of Data;
- investigate query/browse strategies for distributed path retrieval.

* The research activities as described in this paper were funded by Ghent University, the Interdisciplinary Institute for Broadband Technology (IBBT), the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT), the Fund for Scientific Research-Flanders (FWO-Flanders), and the European Union.

2 The story between two concepts

The goal of the demo is to automatically generate a story between two concepts, based on formally described links. We obtain such a story in three steps: path generation, node illustration, and path presentation.

2.1 Path generation

Path generation is based on querying Linked Data sources. In this demo, we consider the following linked datasets to obtain a path: DBpedia, Facebook, Freebase, Wiktionary, WordNet, CIA Factbook, Eurostat, GeoNames, MusicBrainz, linkedMDB, and New York Times. DBpedia serves as the interconnection hub between the other datasets; using Facebook adds a personalized touch to the application. Note that we use standard Named Entity and Disambiguation (NERD) techniques to link output from the Open Graph Protocol to DBpedia resources. Also note that we assume that the two input concepts (i.e. the start- and endpoint) are located within our predefined list of data sources.

Since we do not want to load all our datasets into one RDF store for scalability reasons, we introduce *progressive parallel link browsing*. The latter tries to find paths between two resources by combining the following tasks:

- (1) find a path between the start resource and a DBpedia resource (DBpedia start resource);
- (2) find a path between the end resource and a DBpedia resource (DBpedia end resource);
- (3) find a path between the DBpedia start resource and the DBpedia end resource (either inside DBpedia or via another dataset).

This approach enables parallel execution by making the first task independent from the last. Also, since the nodes in the path are discovered sequentially, progressive playback is possible during discovery.

The choice of which links to follow is crucial in finding meaningful paths for the end-user. Therefore, a property prioritization mechanism is introduced. Certain properties (such as `rdfs:seeAlso`, `foaf:primaryTopic`, `dbpedia:wikiPageWikiLink`) contain little semantics in order to explain the link between two resources. Further, following `rdf:type` or `dcterms:subject` might result in links that are too general (e.g. both resources are `owl:Things`). However, when the type is specific enough (e.g., `Category:FC_Bayern_Munich_players`), it can be an interesting link. Therefore, a priority list is needed indicating which properties and classes/categories should be avoided during querying. Note that these ‘blacklists’ might still be used in fallback scenarios when the original attempt did not give any result.

2.2 Node illustration

Given a path between start and end resource, illustrations of the nodes need to be found in order to enable a path presentation (see further). These illustrations can be text, video, audio or images. A first attempt

to provide a human-readable representation of a resource is to follow your nose: lookup the RDF properties of the resources and search for illustrative properties:

- text: `rdfs:label`, `dbpedia-owl:abstract`, `rdfs:comment`, ...;
- image: `foaf:depiction`, `dbpprop:picture`, `lode:illustrate`, ...;
- video/audio: `lode:illustrate`, ...

Alternatively, we use Web services to look up images (e.g. Flickr, Google Images, Bing Images, ImageNet), videos (e.g. YouTube, Vimeo), and audio (e.g. Grooveshark) to provide illustrative material for the resources.

2.3 Path presentation

As a final step, we present the retrieved path by means of a multimedia presentation with voice over. This requires a fluent narrated story, assembled from two types of data: topic descriptions and links. Topic descriptions are pieces of text fetched from certain RDF properties (see previous section) and give a short summary about the current topic. Links between nodes glue these pieces together into one story. However, narrating links is more difficult, since there is no descriptive text available. Different approaches are possible: from a simple display of the property label (e.g. `dbpedia-owl:birthDate` becomes ‘has birth date’) to a property-specific illustrator. The current version of the demo is limited to the simple property label approach.

Using the Acapela text-to-speech service¹, we are able to provide a voice over narrating the textual representation of nodes and links, making the multimedia presentation complete.

3 Implementation

The application runs on a classic client-server architecture where the path generation step is executed on the server, while node illustration and path presentation are performed at the client (i.e. in the browser). A number of performance optimizations were implemented:

- where possible, queries during the path generation are run in parallel;
- when parts of a path are found, these are already send back to the client;
- the client will run the node illustration tasks in the background (i.e. during presentation of the first nodes, information about the following nodes will be fetched).

4 Example scenario

In order to illustrate the idea, we will connect the dummy Facebook user *John Doe* to *Chicago*. Using the Facebook Open Graph protocol, we find out that he likes David Guetta (`ogp:userA ogp:likes dbpedia:David_Guetta`). By looking up `dbpedia:David_Guetta`, we find his birthplace to

¹ <http://www.acapela-vaas.com>

be Paris (`dbpedia:David_Guetta dbpprop:birthPlace dbpedia:Paris`). We derive from this resource that it is the location of the Eiffel tower (`dbpedia:Eiffel_Tower dbpedia-owl:location dbpedia:Paris`). One of the designers of the Eiffel tower was Émile Nougier who died in 1898-02-20 (`dbpedia:Emile_Nougier dbpedia-owl:significantBuilding dbpedia:Eiffel_Tower; dbpprop:deathDate "1898-02-20"`). The latter is also the birthdate of Jimmy Yancey, who died in Chicago (`dbpedia:Jimmy_Yancey dbpprop:birthDate "1898-02-20"; dbpprop:deathPlace dbpedia:Chicago`).

A screencast of the Everything is Connected application is available at <http://youtu.be/FawyGFT5Brs>. In this screencast, we show the resulting multimedia presentation for the above described example. Multimedia content showed in this example comes from Google Images, Google Maps, and YouTube.

5 Related Work

Concerning path generation, some work is done in the field of Social Network search. Given the *degrees of separation* principle, [1] studies changes made by today's popularity of social networks. New search techniques are introduced to browse dynamic Internet-based graphs, resulting in better memory-based search. Similar bidirectional search algorithms are introduced, but only consider the narrow Twitter data domain.

In [4], West et al. perform a large scale study on human navigation behaviour in Wikipedia. They examine the paths followed by users to get from one topic to another, by listing all the visited topics in between. From this extracted knowledge, a probabilistic model is derived for predicting what users are looking for. For characterising the types of topics a user will cross, a classification is introduced based on the connectedness of a topic.

For the Path presentation, QWiki² produces similar output. However, this tool is limited to turning any concept found on Wikipedia into a automatically narrated multimedia presentation based on the Wikipedia abstract.

References

1. Reza Bakhshandeh, Mehdi Samadi, Zohreh Azimifar, and Jonathan Schaeffer. Degrees of separation in social networks. In *SOCS*, 2011.
2. Chris Bizer, Anja Jentzsch, and Richard Cyganiak. State of the LOD Cloud. <http://www4.wiwi.fu-berlin.de/lodcloud/state/>.
3. Judith Kleinfeld. Could It Be A Big World After All? The "Six Degrees of Separation" Myth. *Society*, 2002.
4. Jure Leskovec. Human wayfinding in information networks. In *WWW-12*, 2012.
5. S. Milgram. The small world problem. *Psychology Today*, 61:60–67, 1967.

² Example for the topic Giraffe - <http://www.qwiki.com/q/Giraffe>

Semantic Vernacular System: an Observation-based, Community-powered, and Semantics-enabled Naming System for Organisms

Han Wang¹, Nathan Wilson², Kathryn M. Dunn³, and Deborah L. McGuinness⁴

¹ Tetherless World Constellation, Rensselaer Polytechnic Institute, Troy, NY 12180, USA
wangh17@rpi.edu

² Marine Biological Laboratory, Woods Hole, MA 02556, USA nwilson@eol.org

³ Rensselaer Libraries, Rensselaer Polytechnic Institute, Troy, NY 12180, USA dunnk2@rpi.edu

⁴ Tetherless World Constellation, Rensselaer Polytechnic Institute, Troy, NY 12180, USA dlm@cs.rpi.edu

Abstract. The Semantic Vernacular System is a novel naming system for creating named, machine-interpretable descriptions for groups of organisms. Unlike the traditional scientific naming system, which is based on evolutionary relationships, it emphasizes the observable features of organisms. By independently naming the descriptions composed of sets of observational features, as well as maintaining connections to scientific names, it preserves the observational data used to identify organisms. The system is designed to support a peer-review mechanism for creating new names, and uses a controlled vocabulary encoded in the Web Ontology Language to represent the observational features. A prototype system is under development in collaboration with the Mushroom Observer website. It allows users to propose new names and descriptions, provide feedback on those proposals, and ultimately have them formally approved. This effort aims at offering the mycology community a knowledge base of fungal observational features and a tool for identifying fungal observations and further provides a prototype example of how other communities may use a semantically-enabled portal to evolve community naming systems.

Keywords: observation, scientific name, fungus, ontology

1 Introduction

With focus on the evolutionary relatedness between and within named groups such as species, genera, families, etc., the precise application of scientific names often requires careful microscopic work, or increasingly, genetic sequencing[10]. This is problematic to many audiences, especially field biologists, who often do not have access to the instruments and tools required to make identifications on a microscopic or genetic basis. Also, while a number of existing web-based biodiversity observation systems (e.g. eBird[9], iNaturalist[3], ArtPortalen[1], Mushroom Observer[4], etc.) have collected tens of millions of occurrence records and linked observations to scientific literature through scientific names, it would be desirable to connect scientific names to precise, accurate and ideally machine-interpretable descriptions compatible with the Semantic Web. Currently there are few widely accepted standards for how to create and curate such descriptions.

In order to address these issues, we proposed the Semantic Vernacular System, a new naming system that is based on observational characteristics, authored in a peer-review context, and encoded in an ontology. We used fungi as a test case for the proposed naming system and developed a prototype on the Mushroom Observer website.

2 An Observation-based Naming System

Modern Scientific names are critical for understanding the biological literature and provide a valuable way to understand evolutionary relationships. To validly publish a name, a description is required to separate the group of organisms this name applies to from those other names apply to. The frequent revision of descriptions due to new evolutionary evidences has lead to a fact that a given scientific name may over time have multiple descriptions associated with it and a given published description may apply to multiple scientific names. Because of this many-to-many relationship between scientific names and descriptions, the usage of scientific names as a proxy for descriptions is inevitably ambiguous.

To resolve this ambiguity, the proposed system creates an additional layer of abstraction by naming the descriptions themselves independently from the scientific names with which they are associated. We refer to these descriptions as Semantic Vernacular Descriptions (SVDs). Each SVD is defined by a set of observational features, and is connected to a number of scientific names which match that definitional feature description. The names of such SVDs will be distinct from scientific names and more closely related to common names, e.g. *PineSpike*, *SmoothCordedPuffball*, *OchreAndRedSpongeTruffle*, etc. SVDs provide shortcuts for expressing a recognizable set of observational features, can be used to drive computer assisted identification, and facilitate the understanding of scientific names based on the observational features of associated SVDs.

3 Community-powered Authoring Mechanism

Similar to scientific names, the proposed system provides mechanisms for encouraging peer-review, preventing the reuse of names, applying any agreed upon nomenclatural rules, and avoiding the unintended re-publishing of existing descriptions. There are two stages in the naming process: “under review” and “approved”. Any users can create a new SVD as long as they provide a unique name and a machine-interpretable description using a supported ontology. While the SVD is “under review”, other users are free to propose alternative names or descriptions. Once a consensus is reached with sufficient support, the SVD is formally “approved” and cannot be changed by the users. Exactly what is considered sufficient support as well as a formal appeal process for revising an approved definition are expected to be refined as the system is used. The metadata of each proposal, including the proposer information, will be recorded as provenance by the system for evaluating the trustworthiness of the proposal.

Figure 1 presents the prototype interface for proposing an SVD definition. The format for observation input is a set of fungal features together with their values. The system then shows the SVDs matching the input feature description. Users will therefore know whether there are any existing SVDs for their observations. By further clicking on one of the displayed matched SVDs, users can also discover which scientific name(s) potentially apply to their observations.

4 Semantics-enabled Encoding Methods

One of the key components of the proposed system is a controlled vocabulary that describes various observational features including ranges for allowable values. Using the Web Ontology Language (OWL)[5], which has already been actively used by significant parts of relevant biology communities[6], we proposed an expandable ontology for fungal SVDs. Figure 2 illustrates the logic model

of the current ontology. The core entity of the ontology is the `SemanticVernacularDescription` class which has two attributes `VernacularLabel` and `VernacularFeatureDescription`. The latter encompasses an OWL property restriction that describes the observational features. This part can be substituted for vocabularies representing features of other organisms in different applications.

Being encoded in OWL DL, the vocabulary is machine-interpretable and can be processed by existing computationally efficient algorithms. Using SPARQL[8] queries, RDFS reasoning, and OWL classification inferences, the system not only can return which features are included in each fungal SVD, but is capable of returning SVDs that match observations described using this ontology in essence providing identifications. The system also accepts SVD registrations from users and is able to integrate the new entries into the ontology on the fly. This is enabled by the SPARQL 1.1 Update Service[7], which takes the RDF generated from the user interface and then updates the triple store.

5 Conclusion

The Semantic Vernacular System is an observation-based, community-powered, and semantics-enabled naming system for groups of organisms. This system fills the gap between modern scientific names and observation-based descriptions. It also provides a platform for creating and maintaining machine-interpretable descriptions and observational data. We developed a prototype system on the Mushroom Observer website that builds on a peer-reviewed fungal SVD ontology and empowers computer-aided identification of fungi based on their observational features.

6 Future Work

We plan to allow the system to discover and suggest new SVDs based on patterns in user-provided observational data. We are also working on the system's identification capability so that it can return exact, partial, and near matches. Finally, we will continue with the development of the fungal SVD ontology in collaboration with the community including making connections to various existing biology related vocabularies (e.g. EnvO[2]) to capture and standardize the terminology needed to describe fungal observational features. We expect the evolution of the fungal feature ontology to be community driven. However, the revision process for the ontology will be peer-reviewed and versioned. Thus users will be able to suggest new features or new values for existing features, and possibly use those for describing particular observations. However, they will not be able to use them in defining an SVD on the globally deployed website until they have been formally adopted by the community. We are exploring local definition capabilities for temporary usage during the community review process.

References

1. Artportalen, <http://artportalen.se>
2. The Environment Ontology, <http://environmentontology.org>
3. iNaturalist, <http://www.inaturalist.org>
4. Mushroom Observer, <http://mushroomobserver.org>
5. Owl 2 Web Ontology Language Quick Reference Guide, <http://www.w3.org/TR/2009/REC-owl2-quick-reference-20091027>
6. The Phenotype Ontology Research Coordination Network, <http://www.phenotypercn.org>

7. Sparql 1.1 Update, <http://www.w3.org/TR/sparql11-update>
8. Sparql Query Language for RDF, <http://www.w3.org/TR/rdf-sparql-query>
9. Sullivan, B.L., Wood, C.L., Iliff, M.J., Bonney, R.E., Fink, D., Kelling, S.: eBird: a Citizen-based Bird Observation Network in the Biological Sciences. *Biological Conservation* (142), 2282–2292 (2009)
10. Wilson, N., Dunn, K., Wang, H., McGuinness, D.L.: Application of Semantic Technology to Define Names for Fungi. Tech. rep., Tetherless World Constellation at Rensselaer Polytechnic Institute (2012), <http://tw.rpi.edu/web/doc/ApplicationofSemanticTechnologytoDefineNamesforFungi>

Fig. 1. Interface for proposing a definition for a SVD

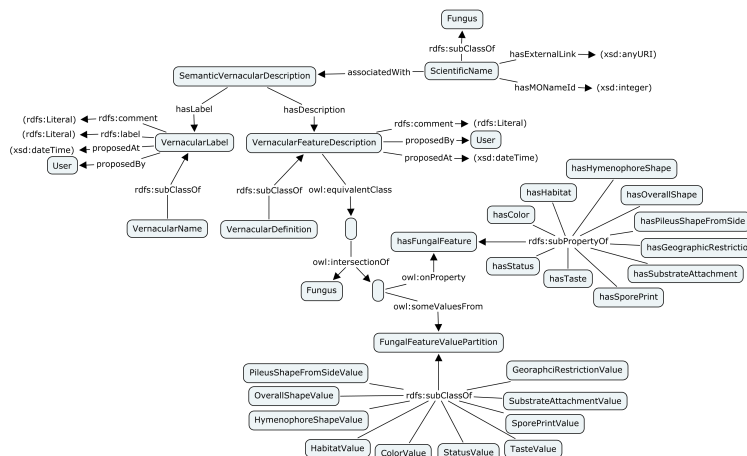


Fig. 2. The logical model of the current fungal SVD ontology

Demo: Efficient Human Attention Detection in Museums based on Semantics and Complex Event Processing

Yongchun Xu¹⁾, Nenad Stojanovic¹⁾, Ljiljana Stojanovic¹⁾, Tobias Schuchert²⁾

¹⁾FZI Research Center for Information Technology
Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany
{name.family name}@fzi.de

²⁾Fraunhofer Institute of Optronics, System Technologies and Image Exploitation (IOSB)
FraunhoferStr. 1, 76131 Karlsruhe, Germany
{name.family name}@iosb.fraunhofer.de

Abstract. In this paper we present a demo for efficient detecting of visitor's attention in museum environment based on the application of intelligent complex event processing and semantic technologies. The detection takes advantage of semantics: (i) in design time for the correlation of sensors' data via modeling of the interesting situations and annotation of artworks and their parts and (ii) in real-time for the more accurate and precise detection of the interesting situation. The results of the proposed approach have been applied in the EU project ARTSENSE.

Keywords: Sensor, Human attention, Complex Event Processing, Ontologies

1 Introduction

In this paper we describe a demo, which shows a semantic-based system providing personalized and adaptive Augmented Reality (A²R) for the visitor, in which the digital contents react depending on the observed artwork and the user's engagement/attention state. In the demo we use semantic technologies for the correlation of sensors' data via modeling the so called interesting situation and use complex event processing to recognize the attention patterns in the event stream.

2 Overview

In order to enable an adaptive experience for the visitor to a museum, the demo is constructed around a four-phase OODA (Observe, Orient, Decide, Act) as shown on **Fig. 1**. In the Observe phase, our approach is concerned with the measurement of covert cues that may indicate the level of interest of the user. In order to consider how a user perceives an artwork, different sensors have been considered: The monitoring of visual behavior will allow the system to identify the focus of attention. The acoustic module should provide important information about environmental influences on patterns of visual attention or psychophysiology. Finally, a video-based hand gesture recognition provides an additional input modality for explicit interaction with the system (e.g., for selecting certain visual items, navigating through menus).

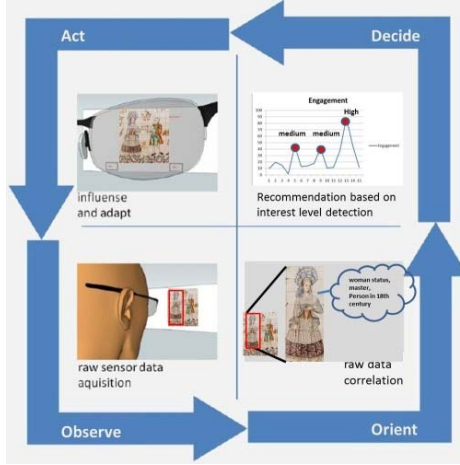


Fig. 1. OODA cycle

All data streams are collected and analyzed in real-time in order to yield a dynamic representation of the user attention state (phase Orient). In the Decide phase, covert physiological cues are used to measure the level of interest or engagement with artwork or with augmented content presented via the AR device. Based on the interpretation of this complex state, the provision of augmented content from a repertoire of available content is made. The presentation of selected content via the AR device (e.g. visual, audio) is subsequently executed during the final Act stage.

3 Semantic based attention

The challenge of this demo is to detect the attention of the visitor in the museum in real time according to the complex metadata. In most situations the attention of the visitors can be determined according to the gaze behavior of the visitors. In some cases the observed object is the attention object of the visitor, while in other cases the visitors pay attention to the information behind the observed objects. Thus, we distinguish between visual attention and content-related attention. **Fig. 2** summaries categories of attentions that are relevant in the museum context, including visual attention, content-based attention and audio disturbance.

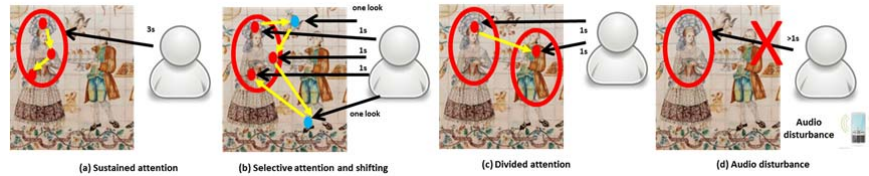


Fig. 2. . Different attention situations relevant for museums: (a) Sustained attention; (b) Selective attention and shifting; (c) Divided attention; and (d) Audio disturbance Implementation

Fig.2.(a) shows the sustained attention: the attention is focused over extended periods of times. Fig.2. (b) is then selective attention and shifting: The visitor changes his/her focus for a very short period and then focuses back on the previously selected artwork. Divided attention (see Fig.2.(c)) means sharing of attention by focusing on more than one relevant object at one time. When a visitor shifts his/her focus between two objects for certain time-period, we can say he/she is interested in the similarity of the objects. In the case that the audio disturbance happens during the visit and the

visitor reacts to this disturbance (see Fig.2.(d)), although the fixation of the visitor is detected, he/she pays no attention to fixated object but to the disturbance. Hence this situation should not be recognized as an attention.

4 The role of Semantic Technologies

The demo is based on knowledge-rich, context-aware, real-time artwork interpretation aimed at providing visitors with a more engaging and more personalized experience. Indeed, we propose to combine annotation of artworks with the time-related aspects as key features to be taken into account when dealing with interpretation of artworks. Thus, the aspects of the museums modeled by ontologies are classified into:

- Static aspects which are related to the structuring of the domain of interest, i.e. describing organization of an artwork and assigning the metadata to it;
- Dynamic aspects which are related to how a visitor's interpretation the elements of the domain of interest (i.e. artworks) evolve over time.

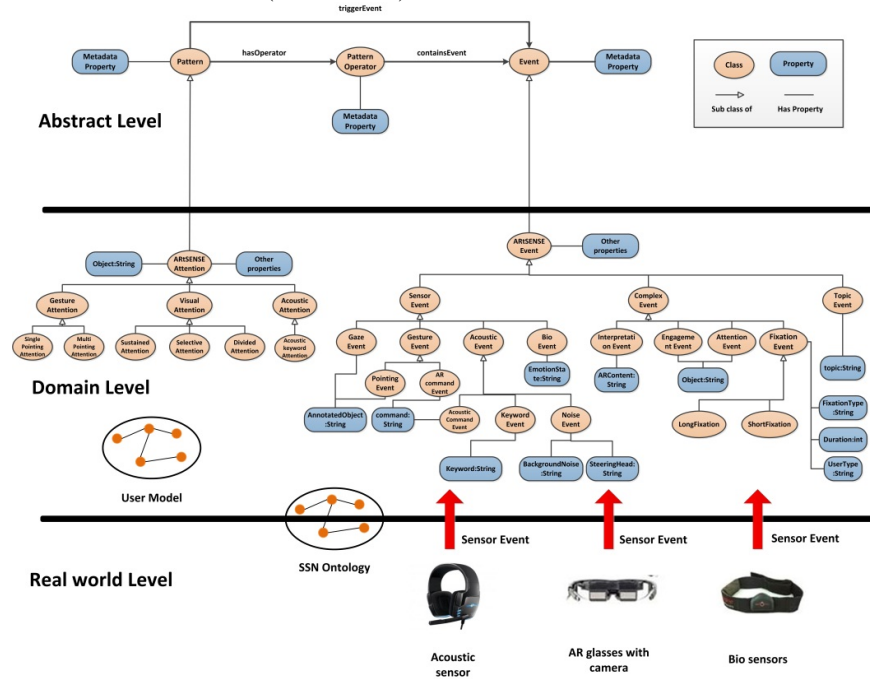


Fig. 3. Ontology for sensor data and patterns of human attention detection

Fig. 3 shows the ontology for sensor data and patterns of human attention detection that is used to describe what happened, when it happened, what the cause was and what the situation meant. We distinguish between three levels. Whereas the abstract

level represents a meta-model for events and patterns, at the domain level¹ we defined the ARtSENSE pattern and event ontology, which describes the types of events and patterns relevant for the ARtSENSE context. For example, the most important situations of interest in ARtSENSE are attentions (see section 3), which are modeled as a pattern hierarchy. The real world level describes the sources of the events. In the case of the ARtSENSE these sources are sensors including acoustic sensors, bio sensors and camera sensors, which can be modeled by the existing sensor ontologies, such as SSN Ontology².

5 Demo setting

The demo will be performed using following hardware equipment:

- A Poster of Valencia Kitchen in MNAD (Museo Nacional de Artes Decorativas, Madrid Spain) as artwork
- Vuzix Star 1200 AR glasses with camera
- M-Audio Fast Track Pro audio card and BEYERDYNAMIC MCE 60.18 mic
- Zephyr HxM Bluetooth Heartrate sensor

6 Demo Implementation

Fig.4. shows the architecture of the ARtSENSE system. The system consists of sensors, sensor adapters, publisher/Subscriber, knowledgebase, metadata annotation tool, ETALIS and Interpretation. All components communicate through ActiveMQ by publishing and/or subscribing to events.

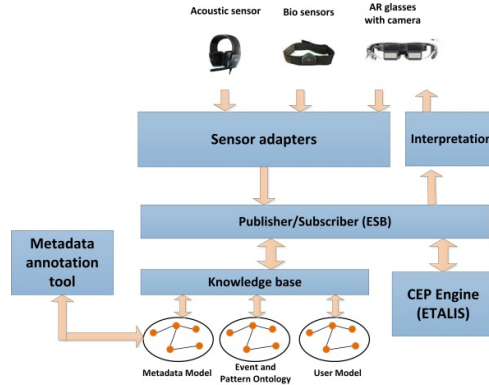


Fig. 4. System architecture

The following sensors are used: see-through glasses with integrated camera that can track the gaze of visitors and display the augmented reality (AR) content to visitors; acoustic sensor senses the acoustic information surrounding visitors such as environment noise or the content that visitors are listening to, and bio sensor observes the biological signals of visitors like heart rate. A recent live demonstration of the system during a workshop in the Louvre museum can be found at <http://www.youtube.com/watch?v=BnbGIlVQMYQ>.

¹ The user model is used to model parameters of visitors such as age, gender, etc.

² W3C Semantic Sensor Network Incubator Group, *Semantic Sensor Network Ontology*

Extracting Relevant Subgraphs from Graph Navigation

Valeria Fionda¹, Claudio Gutierrez², Giuseppe Pirr ¹

¹ KRDB, Free University of Bozen-Bolzano, Bolzano, Italy

² DCC, Universidad de Chile, Santiago, Chile

Abstract. The main goal of current Web navigation languages is to retrieve set of nodes reachable from a given node. No information is provided about the fragments of the Web navigated to reach these nodes. In other words, information about their connections is lost. This paper presents an efficient algorithm to extract relevant parts of these Web fragments and shows the importance of producing subgraphs besides of sets of nodes. We discuss examples with real data using an implementation of the algorithm in the EXPRESS tool.

Key words: Navigation, Subgraph extraction, Linked Open Data

1 Introduction

Daisy is a researcher interested in a new topic. She is aware of a relevant paper and to find other relevant papers she starts looking at the bibliographic references within. After manually browsing for a while this *graph* of citations, for instance by using Google Scholar, and reaching a relevant paper she wonders: how this particular paper has been reached? How it is connected with the original paper? How two other relevant papers encountered during the navigation are linked in this citation graph? The problem becomes more challenging with automatic navigation where Daisy, by using one of the existing navigation languages (e.g., NAUTLOD [3]), can write a complex expression involving the traversal of several data sources. For instance, she wants to collect in the FOAF graph people she knows directly or indirectly that are interested in her new topic. While current graph navigation languages enable to retrieve this set of people, they do not provide information about the structure of the fragment of the FOAF graph where Daisy is linked with these people.

These examples underline the need to augment current navigation languages with capabilities to extract *fragments* (i.e., subgraphs) of the graphs being navigated besides of sets of nodes. Solving this problem is not trivial: there can be an exponential number of paths connecting a seed node with each node reachable according to a navigation expression. Besides, only relevant paths should be kept in the fragment. In a Web setting, which is the focus of this paper, the problem becomes even more challenging since the graph structure is discovered during the navigation.

We present an efficient algorithm to extract Web fragments, describe its pseudo-code and report on its complexity. We present some real world examples to motivate the need of having fragments besides of sets of nodes. Although we focus on the Web, our results are valid for any other graph navigation language. The algorithm has been implemented in the EXPRESS tool available online at <http://swget.wordpress.com/express>.

2 Real World Examples

We present some examples with real data to underline the importance of extracting graph fragments besides of sets of nodes. In Section 3 we describe an efficient algorithm to achieve this goal. The algorithm has been implemented in the EXPRESS tool, which also enables to visualize Web fragments. To illustrate the examples, we use the following subset of the NAUTLOD language [3] through which it is possible to specify navigation expressions with semantic control over the data sources visited.

```

path ::= pred | path[test] | (path)? | (path)* | (path|path) | path/path
pred ::= an RDF predicate
test ::= an ASK-SPARQL query

```

Example 1 (Co-author subgraph). Daisy wants to extract the co-author subgraph of Alberto Mendelzon (AM) up to depth 2 only with publications between 1980 and 1990. The following NAUTLOD expression enables to retrieve the set of AM's co-authors.

```
dblp:Alberto_0._Mendelzon -p (<foaf:maker>/Q/<foaf:maker>)<2-2>
```

where $Q = [\text{ASK } \{?p < \text{dc:issued} > ?y. \text{FILTER}(?y > 1980. ?y < 1990). \}]$ serves to keep only papers, reached by expanding the predicate `<foaf:maker>`, published in the interval considered. The notation `<2-2>` means that the expression within parentheses is repeated two times. From papers reached after evaluating Q it is possible to get their authors, which are direct or indirect co-authors of AM. This expression enables to collect the set of co-authors. However, information about their connections is lost. For instance, while *R. Fagin* and *J. Ullman* appear in the results, it is not possible to know whether they are direct or indirect co-authors of AM or through which sequence of papers they are connected. On the other hand, EXPRESS enables to extract the Web fragment shown in Fig. 1 where it can be observed that AM, *R. Fagin* and *J. Ullman* are direct co-authors in 1982 of a paper appeared in the *TODS* journal.

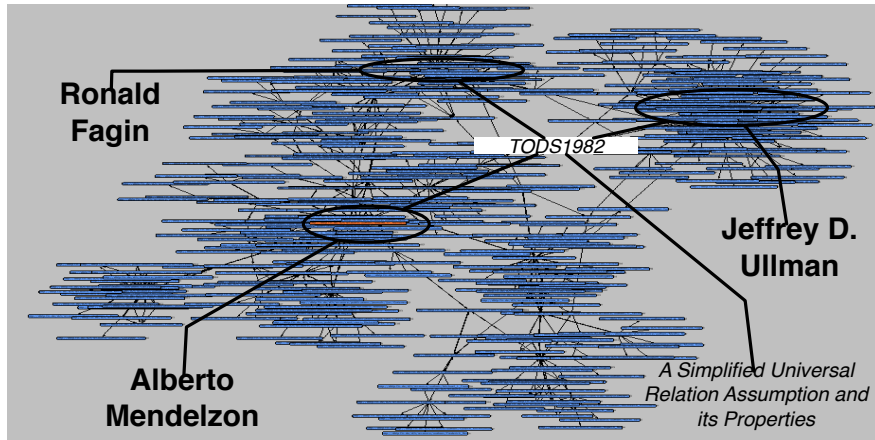


Fig. 1. An excerpt of the co-author subgraph from Alberto Mendelzon.

Example 2 (Related work subgraph). Daisy is interested in query and navigation languages for the Web. She is aware of the paper *Querying the World Wide Web* published in 1998 and writes a navigation expression to find papers up to 2 cite-link away. She decides that only papers having some keywords in the title have to be considered. As citation data are not completely available as linked data, HTML data and links can be exploited. By using Google Scholar identifiers for papers, the following NAUTiLOD expression cast in the standard Web enables to find papers that respect the criteria.

`scholar:scholar?cluster=12837463148248684776 -p(<citedBy>/Q)<2-2>`

where $Q=[\text{title.contains}(\text{"Quer*,Web"})]$ exploits the function `title.contains` to check if the title of a paper contains specific words. The first part of the expression is the URL associated to the paper *Querying the world Wide Web* while `<citedBy>` is the label associated to HTML `href` links pointing to papers citing a given paper. Even in this example, while the paper *Formal Models of Web Queries* is in the result set, Daisy cannot see its connections. EXPRESS enables to extract and visualize the fragment including citation paths that lead from the initial paper to this paper. This fragment provides Daisy with an overview on the topic. For sake of space we do not report the results, which are online available at <http://swget.wordpress.com/express>.

3 Extracting Relevant Subgraphs

As discussed before, there is the need to enhance current graph navigation languages with capabilities to deal with Web fragments. We discuss the case of the NAUTiLOD language even if the same reasoning applies to any other graph query language. The semantics of NAUTiLOD [3] has been modified for outputting subgraphs instead of set of URIs. Each NAUTiLOD expression e has associated a Non Deterministic Finite state Automaton (NFA) A , whose size is linear in the size of the expression $|A| = O(|e|)$. The NFA A is used to recognize the strings (i.e., concatenations of RDF predicates that lead to a result) that belong to the language defined by e . The evaluation of an expression is done in two steps. The first step corresponds to the building of a Web fragment G containing *all* the triples encountered during the navigation (Algorithm 1). NAUTiLOD expressions are memoryless, which means that at each step of their evaluation information about previous “states” is not carried on. Thus, G contains several useless paths, that is, paths for which no final states of the automaton have been reached. Cleaning G in order to maintain only meaningful paths is the aim of the second step performed by the extraction algorithm (Algorithm 2). The desideratum is a subgraph G^* induced only by the set of paths matching e . To achieve this goal two data structures are needed: *i)* a hash-table-like structure S where the key is a uri $u \in G$ and the value is a set of automaton states at which u has been reached; *ii)* a data structure F , the same as S , where only final states are added. To access the set of states for a uri u we use the notation $S(u)$ (resp., $F(u)$). S and F are populated in the *evaluate* algorithm (Algorithm 1, lines 4-6), and are exploited by the *extract* algorithm (Algorithm 2) that keeps only relevant parts of G , thus obtaining G^* .

The space complexity of both S and F is $O(|V| \cdot |e|)$, where V is the number of URIs in G . The overall space complexity is $O(|V| \cdot |e| + |E| + |E_a|)$ where E is the number of edges in G and $|E_a|$ is the number of edges in the NFA A . The overall time complexity is $O(|V| \cdot |e|^2)$.

Algorithm 1: evaluate(NAUtiLOD expression e)

Input: NAUtiLOD expression e , seed URI $seed$
Build: G = RDF graph including all the triples retrieved, Automaton A , Data Structures S and F

```
1   $A$  = build the NFA from  $e$ ;  
2  add the pair  $\langle seed, A.initial \rangle$  to the set of pairs to be looked-up;  
3  while there is a pair  $\langle u, q_k \rangle$  to be looked up s.t.  $\langle u, q_k \rangle$  has not been yet looked up do  
4    add  $q_k$  to  $S(u)$ ;  
5    if  $q_k$  is final then  
6      add  $q_k$  to  $F(u)$ ;  
7     $descr = dereference(u)$   
8    for each transition  $\langle q_k, p, q_j \rangle$  in the NFA  $A$  do  
9      if the triple  $(u, p, u')$  or  $(u', p, u) \in descr$  then  
10       add the triple  $(u, p, u')$  or  $(u', p, u)$  to  $G$ ;  
11       add the pair  $\langle u', q_j \rangle$  to the set of pairs to be looked up;  
12    for each transition  $\langle q_k, \epsilon, q_j \rangle$  in the NFA  $A$  do  
13      add the pair  $\langle u, q_j \rangle$  to the set of pairs to be looked up;  
14    for each transition  $\langle q_k, test, q_j \rangle$  in the NFA  $A$  do  
15      if  $test$  evaluates true over  $descr$  then  
16        add the pair  $\langle u, q_j \rangle$  to the set of pairs to be looked up;
```

Algorithm 2: extract(Graph G , Set S , Set F , Automaton A)

Input : RDF graph G ; Sets of states and final states S, F ; NFA A associated to an expression
Output: G^* = RDF subgraph of G containing only paths from the seed to URIs in the results

```
1  for each pair  $(u, q_k)$  s.t.  $q_k$  belongs to  $F(u)$  and  $(u, q_k)$  has not been already processed do  
2    for each transition  $\langle q_j, p, q_k \rangle$  in the NFA  $A$  do  
3      if the triple  $(u', p, u)$  or  $(u, p, u') \in G$  and the state  $q_j \in S(u')$  then  
4        add  $q_j$  to the set of final states  $F(u')$ ;  
5        add the triple  $(u', p, u)$  or  $(u, p, u')$  to  $G^*$ ;  
6      for each transition  $\langle q_j, \epsilon, q_k \rangle$  in the NFA  $A$  do  
7        add  $q_j$  to the set of final states  $F(u)$ ;  
8      for each transition  $\langle q_j, test, q_k \rangle$  in the NFA  $A$  do  
9        if  $q_j \in S(u)$  then  
10         add  $q_j$  to the set of final states  $F(u)$ ;  
11    return  $G^*$ ;
```

4 Related Work

The main goal of current Web navigation languages is to retrieve set of nodes. Information about the structure of subgraphs spanned by these nodes is lost. Note that SPARQL CONSTRUCT builds graphs from collections of nodes that do not reflect the structure of the navigation. Extracting this subgraph structure is challenging: a naive approach remembering all the paths is not feasible since they can be exponential in number. In a Web context this is even more challenging since the structure of the graph being navigated is not known a priori. We presented an efficient subgraph extraction algorithm for NAUtiLOD that can be extended to any other graph query language. It has been inspired by Earley's parser for Context-free Grammars [2]. It is also related to some work in XPath (e.g., XPlainer [1]). The main difference is that our approach deals with (a priori unknown) graphs instead of (known) trees.

References

1. Consens M. P., Liu J. W. S., Rizzolo F.: XPlainer: Visual Explanations of XPath Queries. In *Proc. of ICDE*, pp. 636-645, (2007).
2. Earley, J.: An Efficient Context-Free Parsing Algorithm. *CACM*, 13(2), pp. 94-102, (1970).
3. Fionda V., Gutierrez C., Pirr6 G.: Semantic Navigation on the Web of Data: Specification of Routes, Web Fragments and Actions. In *Proc. of WWW*, pp. 281-290, (2012).

Jena-HBase: A Distributed, Scalable and Efficient RDF Triple Store

Vaibhav Khadilkar¹, Murat Kantarcioglu¹, Bhavani Thuraisingham¹, and Paolo Castagna²

¹ The University of Texas at Dallas

² Talis Systems Ltd.

Abstract. Lack of scalability is one of the most significant problems faced by single machine RDF data stores. The advent of Cloud Computing has paved a way for a distributed ecosystem of RDF triple stores that can potentially allow up to a planet scale storage along with distributed query processing capabilities. Towards this end, we present Jena-HBase, a HBase backed triple store that can be used with the Jena framework. Jena-HBase provides end-users with a scalable storage and querying solution that supports all features from the RDF specification.

1 Introduction

The simplest way to store RDF triples comprises a relation/table of three columns, one each for *subjects*, *predicates* and *objects*. However, this approach suffers from lack of scalability and abridged query performance, as the single table becomes long and narrow when the number of RDF triples increases [1]. The approach is not scalable since the table is usually located on a single machine. In addition, query performance is diminished since a query requires several self-joins with the same table. There have been several approaches, for *e.g.* [2], which modify the single-table storage schema to improve query performance. Nevertheless, these approaches suffer from the scalability problem, which can be solved by moving from a single-machine to a multi-machine configuration. The cloud computing paradigm has made it possible to harness the processing power of multiple machines in parallel. Tools such as Hadoop and HBase provide advantages such as fault tolerance and optimizations for real time queries. In this paper, we present Jena-HBase³, a HBase backed triple store that can be used with the Jena framework along with a preliminary experimental evaluation of our prototype.

Our work focuses on the creation of a distributed RDF storage framework, thereby mitigating the scalability issue that exists with single-machine systems. The motivation to opt for Jena is its widespread acceptance, and its built-in support for manipulating RDF data as well as developing ontologies. Further, HBase was selected for the storage layer for two reasons: (i) HBase is a column-oriented store and in general, a column-oriented store performs better than row-oriented stores [1]. (ii) Hadoop comprises Hadoop Distributed File System (HDFS), a distributed file system that stores data, and MapReduce, a framework for processing data stored in HDFS. HBase uses HDFS for data storage but does not require MapReduce for accessing data. Thus, Jena-HBase does not require the implementation of a MapReduce-based query engine for executing queries on RDF

³ <https://github.com/castagna/hbase-rdf>, <https://github.com/vaibhavkhadilkar/hbase-rdf>

triples. In contrast, existing systems that use a MapReduce-based query engine for processing RDF data are optimized for query performance, however, they are currently unable to support all features from the RDF specification. Our motivation with Jena-HBase is to provide end-users with a cloud-based RDF storage and querying API that supports all features from the RDF specification.

Our contributions: Jena-HBase provides the following: (a) A variety of custom-built RDF data storage layouts for HBase that provide a tradeoff in terms of query performance/storage. (b) Support for reification, inference and SPARQL processing through the implementation of appropriate Jena interfaces.

2 Jena-HBase Architecture

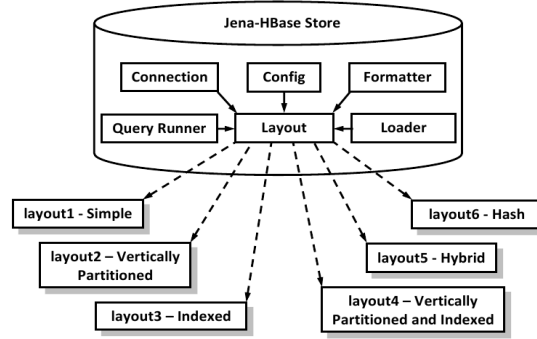


Fig. 1. Jena-HBase - Architectural Overview

Fig. 1 presents an overview of the architecture employed by Jena-HBase. Jena-HBase uses the concept of a store to provide data manipulation capabilities on underlying HBase tables. A store represents a single RDF dataset and can be composed of several RDF graphs, each with its own storage layout. A layout uses several HBase tables with different schemas to store RDF triples; each layout provides a tradeoff in terms of query performance/storage. All operations on a RDF graph are implicitly converted into operations on the underlying layout. These operations include: (a) Formatting a layout, *i.e.*, deleting all triples while preserving tables (**Formatter** block). (b) Loading-unloading triples into a layout (**Loader** block). (c) Querying a layout for triples that match a given $\langle S, P, O \rangle$ pattern (**Query Runner** block). (d) Additional operations include the following: (i) Maintaining an HBase connection (**Connection** block). (ii) Maintaining configuration information for each RDF graph (**Config** block).

We briefly give a summary of the storage schema used by each layout in Table 1. A detailed description of each layout is further given in [3].

3 Experimental Evaluation

We have performed benchmark experiments using SP²Bench (non-inference queries) [4] and LUBM (inference queries) [5] to determine the best layout currently available in Jena-HBase, as well as to compare the performance of the best layout with Jena TDB. We have compared Jena-HBase only with Jena TDB and not with other Hadoop-based systems for the following reasons: (i) Jena TDB gives

Table 1. Storage schemas for Jena-HBase layouts

Layout Type	Storage Schema
Simple	3 tables each indexed by subjects, predicates and objects
Vertically Partitioned (VP)	For every unique <i>predicate</i> , two tables, each indexed by subjects and objects
Indexed	Six tables representing the six possible combinations of a triple namely, SPO, SOP, PSO, POS, OSP and OPS
VP and Indexed	VP layout with additional tables for SPO, OSP and OS
Hybrid	Simple + VP layouts
Hash	Hybrid layout with hash values for nodes and a separate table containing hash-to-node mappings

the best query performance of all available Jena storage subsystems. (ii) The available Hadoop-based systems do not implement all features from the RDF specification. In this section, we show results only for Q1 and Q9 of SP²Bench and Q1 and Q10 of LUBM, however, these results are indicative of the overall trend [3]. Additionally, the figures only show query times and do not include loading times. Finally, as a part of the procedure to determine the best layout, we ran both benchmarks over several graph sizes, but we show results only for a graph of 250137 triples for SP²Bench and for a graph of 5 universities (\approx 560K triples) for LUBM (Fig. 2). Although we used a small graph size, it is still sufficient for determining the best Jena-HBase layout. Since LUBM contains inference queries, we used the Pellet reasoner (v2.3.0) to perform inference.

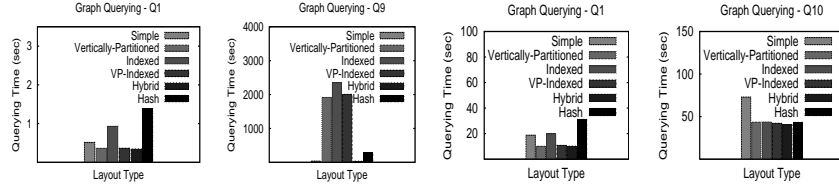
**Fig. 2.** Comparison of layouts for SP²Bench (Q1 and Q9) and LUBM (Q1 and Q10)

Fig. 2 shows a comparison of all layouts for Q1 and Q9 of SP²Bench and Q1 and Q10 of LUBM. We see that the Hybrid layout gives the best results since it combines the advantages of the Simple (Q9 of SP²Bench and Q10 of LUBM) and VP (Q1 of SP²Bench) layouts. The Indexed, VP-Indexed and Hash layouts require longer querying times, since they require multiple row lookups in the SPO, SOP, PSO, POS, OSP and OPS tables (Indexed case) or the SPO, OSP and OS tables (VP-Indexed case) or the mapping table (Hash case).

Fig. 3 shows a comparison of the Hybrid layout with Jena TDB for increasing graph sizes. Note that Jena-HBase values have been scaled down for Q1 (by 1000) and Q9 (by 100) of SP²Bench and for Q1 (by 10) of LUBM for a clear comparison. We see that TDB outperforms the Hybrid layout in the 1M to 25M range for SP²Bench and in the $N = 50$ to 500 range (N is the number of universities) for

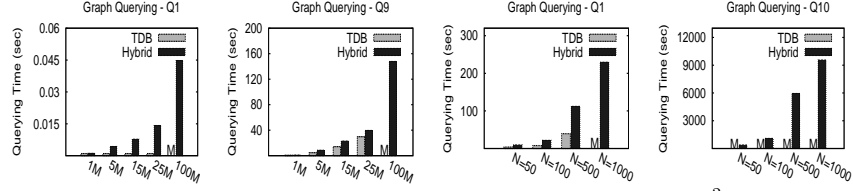


Fig. 3. Comparison of Jena TDB *vs.* Jena-HBase Hybrid layout for SP²Bench (Q1 and Q9) and LUBM (Q1 and Q10). Note that **M** denotes an **Out Of Memory** exception.

Q1 of LUBM. This is expected since for these ranges TDB is able to create and maintain the necessary B+ graph indices in memory, thus resulting in a shorter query execution time. Jena-HBase requires multiple graph pattern matches on increasing graph sizes over a distributed cluster, thus making it slower than TDB. We also observe that TDB fails to execute Q10 of LUBM for the $N = 50$ to 500 range, since the test program runs out of memory during the process of inference for this range. The Hybrid layout successfully executes Q10 for this range, since the reasoner is able to construct the necessary inference related data structures. Finally, we observe that Jena-HBase is more scalable than TDB which fails to construct graphs with 100M triples for SP²Bench and $N = 1000$ universities for LUBM, thereby preventing the execution of any query on these graphs.

4 Conclusion

In this paper, we show that creating a distributed RDF storage framework with existing cloud computing tools results in a scalable data processing solution. Additionally, our solution maintains a reasonable query execution time overhead when compared with a single-machine RDF storage framework (*viz.* Jena TDB).

5 Acknowledgements

This work was partially supported by The Air Force Office of Scientific Research MURI-Grant FA-9550-08-1-0265 and Grant FA-9550-08-1-0260, National Institutes of Health Grant 1R01LM009989, National Science Foundation (NSF) Grant Career-CNS-0845803, and NSF Grants CNS-0964350, CNS-1016343 and CNS-1111529. We thank Dr. Robert Herklotz for his support.

References

1. D. J. Abadi, A. Marcus, S. Madden, and K. J. Hollenbach. Scalable Semantic Web Data Management Using Vertical Partitioning. In *VLDB*, pages 411–422, 2007.
2. K. Wilkinson, C. Sayers, H. Kuno, and D. Reynolds. Efficient RDF Storage and Retrieval in Jena2. Technical report, HP Laboratories, 2003.
3. V. Khadilkar, M. Kantarcioglu, P. Castagna, and B. Thuraisingham. Jena-HBase: A Distributed, Scalable and Efficient RDF Triple Store. Technical report, 2012. <http://www.utdallas.edu/~vvk072000/Research/Jena-HBase-Ext/tech-report.pdf>.
4. M. Schmidt, T. Hornung, G. Lausen, and C. Pinkel. SP²Bench: A SPARQL Performance Benchmark. In *ICDE*, pages 222–233, 2009.
5. Y. Guo, Z. Pan, and J. Heflin. LUBM: A benchmark for OWL knowledge base systems. *J. Web Sem.*, 3(2-3):158–182, 2005.

Applying Multidimensional Navigation and Explanation in Semantic Dataset Summarization

James R. Michaelis, Deborah L. McGuinness, Cynthia Chang, Joanne S. Luciano, and James Hendler

Tetherless World Constellation, Rensselaer Polytechnic Institute,
110 8th Street, Troy, NY 12180
{michaelis,dlm,csc,jluciano,hendler}@cs.rpi.edu

Abstract. A key objective of multidimensional dataset analysis is to reveal patterns of interest to users, but can be difficult to conduct due to the challenges of both presenting and navigating large datasets. This work explores how initial summarizations of multidimensional datasets can be generated (designed to reduce the number of data points which would need to be displayed), using *summarization policies* based on provided dataset values. Additionally, functionality for explaining the derivation of summarizations is being designed in line with prior work on aiding analyst interactions with data processing systems. To help drive development of this work, as well as provide illustrative use cases, we are presently designing a dataset summarization generator as part of greater work being done on an infrastructure for managing evidence of technical emergence in varying research disciplines via automated review of published materials.

Keywords: OLAP, Explanation, Provenance

1 Introduction

A key objective of multidimensional dataset analysis is to reveal patterns of interest to analysts. In many cases, these analyses will involve navigation over a dataset to expose content likely to have interesting patterns. However, multidimensional analysis has been observed to be challenging to analysts for the following reasons [1]:

1. They may be overwhelmed by a data space evidence set if it is too large.
2. They may not have time or expertise to perform extensive navigation.

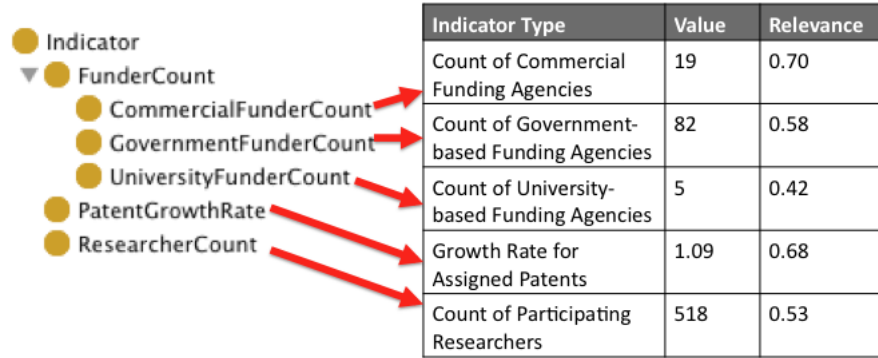
This work explores how initial summarizations of multidimensional datasets can be generated for consuming parties (designed to reduce the number of data points which would need to be displayed) driven by *summarization policies* based on provided dataset values. Focus has been given to RDF-based dataset encodings, due largely to RDFs flexibility in linking to outside data sources (e.g., ontologies for expressing possible data values). Finally, functionality for explaining the derivation of summarizations is being developed - in line with prior work for aiding analyst interactions with data processing systems [2].

2 Evidence Summarization in the ARBITER System

To help drive development of this work, as well as provide illustrative use cases, we are presently developing a dataset summarization generator for the Abductive Reasoning Based on Indicators and Topics of EmeRgence (ARBITER) system - being jointly developed by Rensselaer, BAE Systems, NYU, Brandeis and 1790 Analytics as part of IARPA’s Foresight and Understanding from Scientific Exposition (FUSE) program. ARBITER’s design objective is to scan for signs of technical emergence in published literature - where technical emergence is defined in the FUSE program as [3]: *the process by which research domains appear, mature, and if conditions are favorable, make a significant impact*.

In ARBITER, sets of one or more evidence entries are evaluated to make hypotheses about emergence-related questions for a given topic and time period. For example: *Has a practical application for DNA Microarrays been established in the time period of 2006-2010, based on the document collection PubMed-42?*

In this setting, evidence entries are defined as *emergence indicators*, calculated based on analysis over document collections. Indicators are classified according to an OWL ontology of indicator types, where each indicator is defined to have at least one RDF type, as well as a set of numerical scoring metrics to define relationship of evidence to hypothesis. For brevity, an example is provided with five indicators, each with a single RDF type and two numerical properties (*value* and *relevance to the question answer*, where a higher value is better).



Indicator Type	Value	Relevance
Count of Commercial Funding Agencies	19	0.70
Count of Government-based Funding Agencies	82	0.58
Count of University-based Funding Agencies	5	0.42
Growth Rate for Assigned Patents	1.09	0.68
Count of Participating Researchers	518	0.53

Fig. 1. *Left:* Part of ARBITER’s ontology, which defines indicators. *Right:* A sample of indicator data. Here, indicators are aligned with their corresponding ontology classes.

Currently, these evidence entries are presented as a 2-dimensional spreadsheet. To reduce the number of rows directly presented, policy-based summarization techniques are being explored - deriving from established navigation techniques in OLAP [1]: grouping rows into *collection-based* entries, as well as filtering table entries - each based on specified criteria. For this submission, the following two summarization policies are provided for illustrative purposes:

1. **Grouping:** Group entries together that are SKOS¹ subconcepts of the "FunderCount" class.
2. **Filtering:** Remove entries with relevance scores below 0.55.

Ultimately, the following system conditions are assumed: (i) A maximum number of *summary rows* will be specified, which will appear in the presented summary; (ii) A pre-defined collection of policies will be accessible by ARBITER, along with a pre-defined ordering for their execution; (iii) Policies will be sequentially applied to the evidence set until the summary row count is reached, or all policies have been applied. Initially, an evidence dataset D_0 will represent content directly generated by evidence gathering routines in ARBITER. Each policy execution will yield a transformed dataset view $D_{1...n}$, up until condition (iii) is satisfied.

While initial summarization can be a powerful aid for analyst users, care has to be taken in their usage, since one summarization strategy may not be appropriate for all users and information-seeking tasks. To help analysts keep track of applied strategies, summaries will be accompanied by explanations of their derivation - accessible for individual entries. In Figure 2, an example summary view - along with a supporting explanation - is provided.

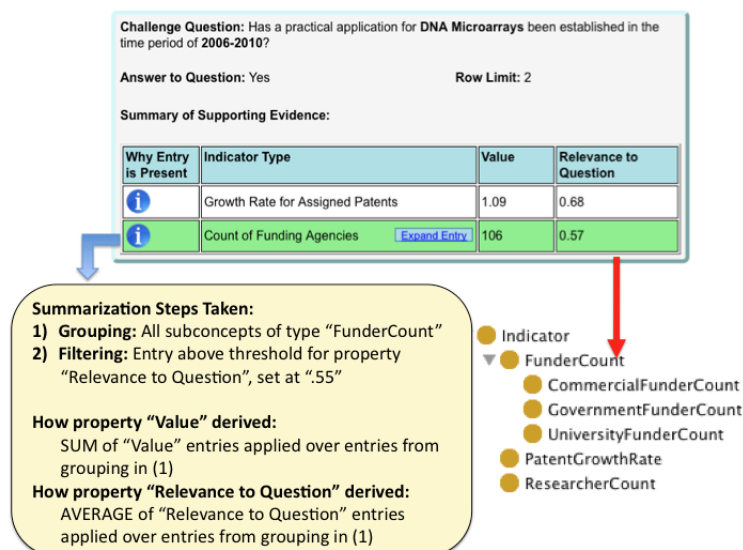


Fig. 2. Here, a summary view is displayed, with a collection-based entry highlighted in green (which aggregates the 3 funding entries from Figure 1). An explanation of why this entry is present - and how column values were derived - can be accessed by clicking the "i" icon.

¹ RDF SKOS Vocabulary: <http://www.w3.org/TR/skos-reference/>

System Development: ARBITERs summary generator is being designed to take three inputs: (i) A set of fine-grained evidence; (ii) A set of SPARQL-encoded preference policies, along with an accompanying execution order; and (iii) Corresponding ontologies for encoding the preference and evidence data. For encoding evidence, we are now exploring use of the RDF Datacube² vocabulary - given its support for representing multidimensional data.

Upcoming Directions: In upcoming work, focus will be given to the following three issues: (i) selection of summarization policies which align with an analysts perceived preferences, (ii) based on the summarization explanations provided, enabling analysts to tweak applied strategies to generate new summarizations, and (iii) enabling analysts to identify source documents used to create evidence entries (similar to efforts discussed in [2]). For situations where significant numbers of evidence entries are presented (e.g., over 100), all three issues are expected to need addressing.

3 Acknowledgements

We would like to thank our collaborators at BAE Systems, Sean Stromsten, Dan Hunter and Olga Babko-Malaya for their assistance in this work. Support has been provided by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20154. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

Disclaimer: 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, DoI/NBC, or the U.S. Government.

References

1. Giacometti, A. and Marcel, P. and Negre, E. A framework for recommending OLAP queries. 11th International Workshop on Data Warehousing and OLAP (DOLAP08), 73-80, 2008.
2. Murdock, J., McGuinness, D., Pinheiro da Silva, P., Welty, C., and Ferrucci, D. Explaining conclusions from diverse knowledge sources. Proceedings of ISWC 2006, 861-872, 2006.
3. Foresight and Understanding from Scientific Exposition (FUSE) Program - Broad Agency Announcement (BAA) [IARPA-BAA-10-06]. Retrieved from: <http://www.iarpa.gov/solicitations.fuse.html>. Date Last Accessed: 07/28/2012.

² RDF Datacube Vocabulary: <http://www.w3.org/TR/vocab-data-cube/>

Building Large Scale Relation KB from Text

Junfeng Pan, Haofen Wang, and Yong Yu

APEX Data & Knowledge Management Lab
Shanghai Jiao Tong University
800 Dongchuan Rd., Shanghai 200240, China
{panjf, whfcarter, yyu}@apex.sjtu.edu.cn

Abstract. Recently more and more structured data in form of RDF triples have been published and integrated into Linked Open Data (LOD). While the current LOD contains hundreds of data sources with billions of triples, it has a small number of distinct relations compared with the large number of entities. On the other hand, Web pages are growing rapidly, which results in much larger number of textual contents to be exploited. With the popularity and wide adoption of open information extraction technology, extracting entities and relations among them from text at the Web scale is possible. In this paper, we present an approach to extract the subject individuals and the object counterparts for the relations from text and determine the most appropriate domain and range as well as the most confident dependency path patterns for the given relation based on the EM algorithm. As a preliminary results, we built a knowledge base for relations extracted from Chinese encyclopedias. The experimental results show the effectiveness of our approach to extract relations with reasonable domain, range and path pattern restrictions as well as high-quality triples.

Keywords: Linked Data, Relation Extraction, Expectation Maximization.

1 Introduction

In recent years, many knowledge bases, such as DBpedia[1], YAGO[6], Zhishi.me[5], have been published on the Web in the form of linked data, which are very useful for both human reading and machine consumption. Comparing with the number of different entities in these knowledge bases, there are only a few distinct relations. Furthermore, these knowledge bases only extract data from structured or semi-structured data sources without considering implicit knowledge from unstructured text, which is in a huge and increasing amount on the Web. On the other hand, open information extraction, such as Machine Reading[4] and Never-Ending Language Learning[2], focuses on extracting entities and their relations from text at the Web scale.

In this paper, we are motivated to build a knowledge base of relations extracted from text by leveraging open information extraction techniques. Moreover, for each relation, we extract not only subject-object examples but also high level restrictions such as the domains and ranges from text. Both information are quite useful to describe relations, which can be used for further natural language processing training or high-quality ontologies for additional extraction. To extract such information, we adapt a novel algorithm based on Expectation Maximization (EM). And an experimental relation knowledge base is built to show the effectiveness of our algorithm.

2 Building the Relation Knowledge Base

As shown in Figure 1(a), the process of building relation knowledge base has three main steps: text annotation, candidates extraction and iterative score estimation.

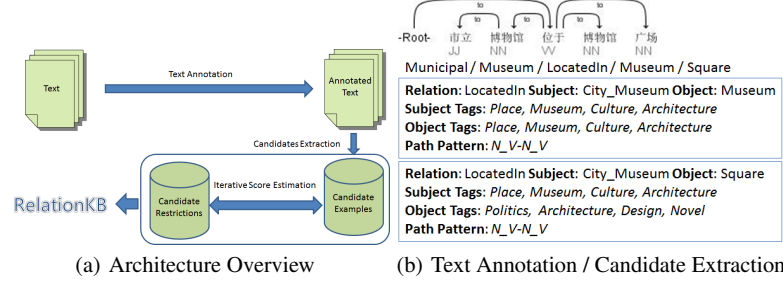


Fig. 1. Building Relation KB

The text annotation step includes tokenization (word segmentation for Chinese), POS tagging and dependency parsing on top of the raw input text. The annotated text is used as the input for the candidate extraction step.

According to the concept of dependency grammar, we can directly extract candidate examples from the annotated text. We identify the relations in the sentences according to POS tags, and for each relation, we enumerate nouns in its subtrees as subject or object candidates. We also extract candidate restrictions in the candidates extraction step. For each candidate example, we tag the head nouns of the subjects and objects to form the candidate domain and range of the relations. We also extract candidate path patterns from subject-relation-object paths on the dependency trees. Figure 1(b) shows two candidates, including examples and restrictions, are extracted from an annotated Chinese sentence means *the Municipal Museum is Located in the Museum Square*.

In the score estimation step, we group the candidate examples and restrictions by relation, and process each relation independently. For each relation v , we group the candidates from one sentence into one candidate group. Suppose there are n candidate groups for v and m_i candidates in group c_i . In one group, each candidate c_{ij} contains the extracted candidate subject s_{ij} with tags $T(s_{ij})$, object o_{ij} with tags $T(o_{ij})$, and path pattern p_{ij} . Figure 1(b) can be seen as a two-candidate group for relation *LocatedIn*.

We need a score function $f_E(s_{ij}, o_{ij}|v)$ to judge whether the example (s_{ij}, o_{ij}) is a credible subject-object pair for v . It is reasonable that the credibility of the example is positively correlated to the credibility of its tags and path pattern so we use Equation (1) to estimate the score. Here $N_w(t)$ is the number of words having tag t . And we also introduce three functions $f_D(t_d|v)$, $f_R(t_r|v)$, $f_P(p|v)$ to indicate the credibility of the tag t_d in the domain, the tag t_r in the range, and the path pattern p for v .

$$f_E(s_{ij}, o_{ij}|v) = f_P(p_{ij}|v) \left(\sum_{t_d \in T(s_{ij})} \frac{f_D(t_d|v)}{N_w(t_d)} \right) \left(\sum_{t_r \in T(o_{ij})} \frac{f_R(t_r|v)}{N_w(t_r)} \right) \quad (1)$$

Intuitively, $f_D(t_d|v)$ should be the real count of t_d over the maximum possible count. We assume that each candidate group is worth 1 count (which implies the maximum possible count is n) and allocate it for each candidate in one group according to $f_E(s_{ij}, o_{ij}|v)$. The way to compute f_R and f_P is almost the same as f_D . Details are given in Equation (2).

$$f_L(t|v) = \frac{1}{n} \sum_{cond} \frac{f_E(s_{ij}, o_{ij}|v)}{\sum_{k=1}^{m_i} f_E(s_{ik}, o_{ik}|v)} \quad (2)$$

$$(L, t, cond) \in \{(D, t_d, t_d \in T(s_{ij})), (R, t_r, t_r \in T(o_{ij})), (P, p, p = p_{ij})\}$$

According to Equation (1) and (2), the scores of the examples and the restrictions are interdependent. It is natural to design an EM[3] algorithm to estimate them simultaneously:

1. Initialize $f_E(s_{ij}, o_{ij}|v) = \frac{1}{m_i}$.
2. Update f_D, f_R, f_P, f_E iteratively until convergence:
 - (a) **E-step:** For every tag and path pattern, update $f_D(t_d|v), f_R(t_r|v), f_P(p|v)$ using Equation (2).
 - (b) **M-step:** For every example, update $f_E(s_{ij}, o_{ij}|v)$ using Equation (1).

Finally we can build the relation knowledge base by the examples (subject-object pairs) and the restrictions (tags for domain and range as well as path patterns) for each relation according to the scores of these candidates.

3 Preliminary Results on Chinese Encyclopedias and Zhishi.me

We built an experimental relation knowledge base from the abstract text of all the entries in three online Chinese encyclopedias, Wikipedia in Chinese, Baidu Baike and Hudong Baike. There are 2,517,826 pieces of text and 20,637,524 sentences in total.

In addition to the text, we use the property `zhishi:category` at a Chinese Linked Open Data, Zhishi.me which extracts the categories for the entities from the above three online encyclopedias. We used them to tag the extracted entities. There are a total of 985 tags and 3,109,448 words that have at least one of these tags.

Our experimental relation knowledge base contains 7,097 relations from the text in total. For one relation, there are 107.46 tags in domain, 103.13 tags in range, 9.68 path patterns and 276.03 subject-object pairs in average.

We sampled 40 relations for human evaluation. For each relation, the domain, the range and the examples can be seen as three ranked lists according to the scores of the elements in them. We use the average precision (average of the precision value obtained for the top k elements, each time a relevant/correct element is retrieved) as the metric. The mean average precision is 0.788 for the domain (top 5), 0.865 for the range (top 5), and 0.657 for the examples (top 15). The average precision distribution on the sample relations is shown in Figure 2.

We have found errors are mainly caused by the following reasons: (1) Some words has wrong tags; (2) Some relation phrases are not actually relations (maybe just modifiers to nouns) in the sentence; (3) Some extracted noun phrases are incomplete or linked

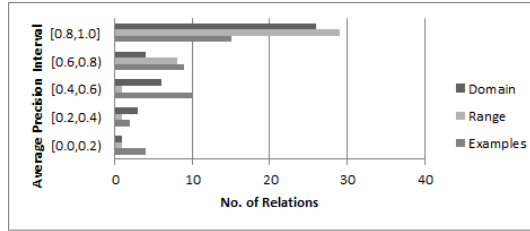


Fig. 2. The Average Precision Distribution on the Sample Relations

to wrong entities. These kinds of errors indicate how to improve the quality of our relation knowledge base in the future (e.g. to improve tag cleaning, relation identification, entity linking and disambiguation, etc).

We also provide a simple web site for users to browse our experimental relation knowledge base at <http://relationkb.apexlab.org>. There are two services in the web site, a lookup service and a SPARQL query service.

4 Conclusion and Future Work

In this paper, we leverage open information extraction and propose an EM algorithm to build a knowledge base which contains the examples and restrictions of the relations from text. Also we give some preliminary results in Chinese to show effectiveness of our algorithm. In the future, we are planning to use advanced method to identify the relations and better entity linking algorithm to improve the quality of the relation knowledge base. In additional, it would be better if we added some structures about the relations, such as relation clusters or relation hierarchies. Finally we are also planning to use our relation knowledge to populate more and more linked data from text and update the relation knowledge base itself when populating.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: a nucleus for a web of open data. In: ISWC/ASWC (2007)
2. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Jr., E.R.H., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: AAAI (2010)
3. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* (1977)
4. Etzioni, O., Banko, M., Cafarella, M.J.: Machine reading. In: AAAI Spring Symposium (2007)
5. Niu, X., Sun, X., Wang, H., Rong, S., Qi, G., Yu, Y.: Zhishi.me: weaving chinese linking open data. In: ISWC (2011)
6. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: WWW (2007)

Reasoning in RDFS is Inherently Serial, at least in the worst case

Peter F. Patel-Schneider

`pfpschneider@gmail.com`

There have recently been several papers presenting scalable distributed inference systems for the W3C Resource Description Framework Schema language (RDFS). These papers have made claims to the effect that they can produce the RDFS closure for billions of RDF triples using parallel hardware.

For example, Urbani *et al* claim “a distributed technique to perform materialization under the RDFS [...] semantics using the MapReduce programming model” [4]. Their initial algorithm used a small number of simple MapReduce passes in sequence to perform distributed materialization. In each pass all triples are distributed according to the processing performed during that pass, but as well all *schema triples* (triples with predicate `rdfs:domain`, `rdfs:range`, `rdfs:subPropertyOf`, or `rdfs:subClassOf`) are sent to all reduce nodes. To handle some of the issues described here, later versions of their algorithm looped through the passes until no new inferences were made.

Similarly, Weaver and Hendler claim to be “materializing the *complete* [emphasis added] finite RDFS closure in a scalable manner” [6]. Their algorithm uses a single MapReduce pass and sends *ontological triples* (schema triples plus triples with predicate `rdf:type` and object `rdfs:Datatype`, `rdfs:Class`, or `rdfs:ContainerMembershipProperty`) to all reduce nodes. However, the algorithm distributes the other triples evenly, without regard to contents. Each reduce node then produces the finite RDFS closure of its input and these results are finally combined. They call their algorithm “embarrassingly parallel” because the inference part can be split into many independent pieces.

These claims seem believable, as on first glance there appears to be nothing to prevent the effective parallelization of RDFS inference. However, RDFS inference does have some aspects that make it less easy than one might expect.

First, RDFS has a formal model-theoretic semantics [1]. The model-theoretic semantics provides an unequivocal definition for inference and related notions in RDFS. The definition is not, however, in terms of a simple set of rules of inference, but is instead in terms of entailment, a model-theoretic notion. Claims of computing RDFS entailment, inference, closure, or materialization must thus be compared with this model-theoretic definition.

Second, RDFS has a infinite number of axioms. This trivially makes the RDFS closure of any RDF graph infinite. Necessarily these axioms are very simple and ter Horst [3] has discovered how to produce a finite subset of the RDFS closure that contains all the useful triples, and from which the remaining triples can be very easily computed.

Third, RDFS inherits the peculiarities of RDF, particularly that its internal vocabulary for defining ontologies can be manipulated in the same way as domain vocabularies can. This provides some power, but also provides considerable opportunities for mischief. One particularly mischievous way of manipulating the RDF and RDFS internal vocabulary is to use it to make extra connections within the internal vocabulary itself, as in

```
rdf:type rdfs:subPropertyOf rdfs:subPropertyOf .
```

Although this kind of manipulation is allowable, the problems it produces wildly outweigh its utility. Another mischievous way of manipulating the RDFS internal vocabulary is to subordinate it to non-RDFS vocabulary, as in

```
rdfs:subClassOf rdfs:subPropertyOf ex:interclassRelationship .
```

Although it may be interesting to create such “higher-level” vocabulary, such pursuits are generally best left to philosophers.

However, some manipulation of the RDF and RDFS vocabulary can be useful under certain circumstances. For example, one might want to make some differentiation between different kinds of properties and use a specific property for the sub-property relationship for them, as in

```
ex:partProperty rdfs:subClassOf rdf:Property .
ex:partSubPropertyOf rdfs:subPropertyOf rdfs:subPropertyOf .
ex:partSubPropertyOf rdfs:domain ex:partProperty .
ex:partSubPropertyOf rdfs:range ex:partProperty .
```

Although this ability to extend the RDF and RDFS ontology vocabulary does not appear to be much used in practice, with only two sub-properties of `rdfs:subPropertyOf` and three sub-properties of `rdfs:subClassOf` in the 2011 Billion Triple Challenge Dataset, it nonetheless can be useful.

When one is claiming to perform RDFS inference one needs to be very clear how one is handling these tricky issues. If one does not explicitly disclaim completeness then the RDF model-theoretic semantics provides the standard that must be adhered to. If one is claiming to produce RDF closures or perform RDFS materialization then one has to provide some way of handling the infinite axiomatization of RDFS. If one does not explicitly state which RDF graphs are not permitted as input then all RDF graphs must be allowed.

As noted above Urbani *et al* prominently claim to be performing RDFS materialization, and do so without any caveats in several places. It is only on close examination of their 2009 paper [5] that one finds that they ignore several inferences where domain classes or properties are made superclasses or superproperties of RDF or RDFS classes or properties and some inferences on container membership properties. These are the only qualifications related to RDFS materialization to be found in this paper. In their 2011 paper [4], there are some comments in the body of the paper about ignoring hijacking cases. There is also a comment about adding one extra step to handle special cases related to container membership properties and datatypes. Finally, there is a comment indicating that they also do not handle RDF and RDFS ontology vocabulary extension, but that it could be somehow handled by adding an extra loop. In

essence in this latter paper Urbani *et al* eliminate the possibility of modifying or extending the RDF and RDFS vocabulary and thus excluding the situations that make RDF and RDFS different from a very cut down version of OWL 2 DL [2]. What they do end up computing, provided that one ignores several bugs in the algorithms and fills in a lot of missing information, is fairly close to complete when their exclusions are taken into account. Their work does point the way to using a fixed number of simple MapReduce passes to materialize the *finite* RDFS closure when the RDF and RDFS vocabulary is neither modified nor extended.

Weaver and Hendler do state very clearly that they are only producing finite RDFS closures, but then very clearly state that they are complete. It is only in the middle of the paper that they state that they assume that there are no modifications or extension of the RDF and RDFS vocabulary. So, although Weaver and Hendler are up-front about the finite closure, counter to their claims of completeness they only consider the RDFS fragment of OWL 2 DL. The result of this algorithm appears to be nearly complete when their exclusions are taken into account, although this is somewhat hard to tell, as there are some bugs in the arguments in the paper. This work points the way to a simple distributed, single-pass method to materialize the *finite* RDFS closure when the RDF and RDFS vocabulary is neither modified nor extended.

So if the RDF and RDFS ontology vocabulary is neither modified nor extended it is possible to easily compute RDFS closures. However, just extending the RDF and RDFS ontology vocabulary produces problems. Consider

```
ex:physicalSubPropertyOf rdfs:subPropertyOf rdfs:subPropertyOf .
ex:physicalPartOf ex:physicalSubPropertyOf ex:partOf .
ex:wheelOf ex:physicalSubPropertyOf ex:physicalPartOf .
```

The closure of this RDF graph includes

```
ex:wheelOf rdfs:subPropertyOf ex:partOf .
```

which is not necessarily produced by Urbani *et al* or Weaver and Hendler. The basic problem here is that it is possible to infer new schema or ontological triples, and these triples can then produce further triples, even further ontological triples.

It might be the case that these failures are simply due to a bug or limitation in the algorithms, and that the general approach in these papers, or a slight generalization of it, can be used to perform finite RDFS materialization. However, this is not the case. Instead, it turns out that there are proofs in RDFS that are arbitrarily large but only depend on a fixed number of triples that contain any of the RDF or RDFS vocabularies.

An example of this comes from RDF graphs of the following form:

```
sp1 rdfs:subPropertyOf rdfs:subPropertyOf .
sp2 sp1 sp1 .
sp3 sp2 sp2 .
...
spn spn-1 spn-1 .
```

This graph RDFS-entails

```
spn rdfs:subPropertyOf rdfs:subPropertyOf .
```

Any proof process that produces these results will need to perform an arbitrarily large amount of work involving triples that do not mention the RDF or RDFS vocabularies. Even worse, there is no commonality between the triples beyond the chaining from one triple to the next. This means that any distribution process will not be able to combine any significant fraction of the triples together, except by using this chain. The end result is that it is not possible to produce these proofs in a fixed number of finite parallel steps, even if determining the closure of all triples that include any RDF or RDFS vocabulary can be done in a fixed amount of time.

In fact, a simple reduction from the Monotone Circuit Problem shows that RDFS entailment is inherently serial, so it is unlikely that any significant speedup can be achieved using a polynomial number of processors, at least in the worst case. In cases like the one above a parallel implementation will have to produce inferences one by one down the chain. Of course, this example is rather contrived, but a system that performs RDFS materialization is not allowed to pick and choose its inputs; it has to handle all valid RDF graphs that have not been explicitly excluded.

So what then does this say about the empirical difficulty of computing finite RDFS closures using distributed computing, as opposed to the difficulty of describing just which are the problematic cases? Actually very little, as the example above demonstrates the worst case, and the general pattern is very unlikely. (The situation is, however, very different for OWL, as chaining of information along domain properties is quite common in OWL.) The investigations by Urbani *et al* and Weaver and Hendler show that computing the RDFS closure could be usually quite easy, particularly when excluding the mischievous cases that no sane person should argue for. Further, it would not be hard to detect cases where the RDF and RDFS vocabulary is being extended and to handle these cases specially. Although this would be slow in comparison to the normal processing, this is the price that users of the extension facility must pay.

References

1. Patrick Hayes. RDF semantics. W3C Recommendation, <http://www.w3.org/TR/rdf-mt/>, 2004.
2. Boris Motik, Peter F. Patel-Schneider, and Bijan Parsia. OWL 2 web ontology language: Structural specification and functional-style syntax. W3C Recommendation, <http://www.w3.org/TR/owl2-syntax/>, 2009.
3. Herman J. ter Horst. Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary. *Journal of Web Semantics*, 3(2-3):79–115, 2005.
4. Jacopo Urbani, Spyros Kotoulas, Jason Maassen, Frank van Harmelen, and Henri Bal. WebPIE: A web-scale parallel inference engine using MapReduce. *Journal of Web Semantics*, 10:59–75, 2012.
5. Jacopo Urbani, Spyros Kotoulas, Eyal Oren, and Frank van Harmelen. Scalable distributed reasoning using MapReduce. *ISWC-2009*, pages 634–649.
6. Jesse Weaver and James A. Hendler. Parallel materialization of the finite RDFS closure for hundreds of millions of triples. *ISWC-2009*, pages 682–697.

INSTANS: High-Performance Event Processing with Standard RDF and SPARQL

Mikko Rinne, Esko Nuutila, and Seppo Törmä

Department of Computer Science and Engineering,
Aalto University, School of Science, Finland
`firstname.lastname@aalto.fi`

Abstract. Smart environments require collaboration of multi-platform sensors operated by multiple parties. Proprietary event processing solutions lack interoperation flexibility, leading to overlapping functions that can waste hardware and communication resources. Our goal is to show the applicability of standard RDF and SPARQL – including SPARQL 1.1 Update – for complex event processing tasks. If found feasible, event processing would enjoy the benefits of semantic web technologies: cross-domain interoperability, flexible representation and query capabilities, interrelating disjoint vocabularies, reasoning over event content, and enriching events with linked data. To enable event processing with standard RDF/SPARQL we have created INSTANS, a high-performance Rete-based platform for continuous execution of interconnected SPARQL queries.

Keywords: Rete, SPARQL, RDF, Complex event processing

1 Introduction

Complex event processing is currently more dominated by proprietary systems and vertical products than open technologies. In the future, however, internet-connected people and things moving between smart spaces in smart cities will create a huge volume of events in a multi-actor, multi-platform environment.

Semantic web technologies enable flexible representation of events in RDF and advanced specification of event patterns with SPARQL. They provide possibilities to reason about event content and to enrich events with linked open data available in the web. Semantic web standards have clear potential to improve the interoperability and offer new capabilities in complex event processing.

A major event processing application can hardly be created out of a single SPARQL query. The INSERT operation in SPARQL 1.1 Update introduced a critical new property: By inserting data into a graph, collaborating SPARQL queries can store intermediate results and communicate with each other. On an environment supporting simultaneous, continuous evaluation of multiple queries, SPARQL can be used to create entire event processing applications [6].

After finding no other platform for incremental processing of multiple SPARQL 1.1 queries, we created INSTANS. Based on the tried and tested Rete-algorithm [3], INSTANS shares equivalent parts of queries, caches intermediate matches

and provides results immediately, when all the conditions of a query have been matched. In addition to being competitive in SPARQL query processing [1], our studies show qualitative and quantitative benefits compared to SPARQL-based systems using repeated execution of queries over windows on event streams [6].

Here we extend the discussion in [5] by adding further information on the INSTANS implementation of continuous incremental SPARQL query processing.

2 INSTANS Event Processing Platform

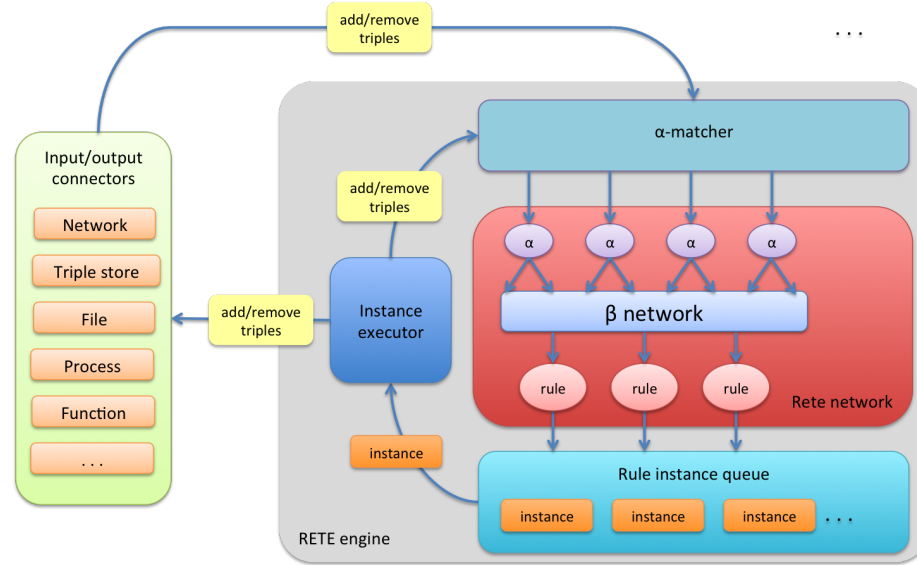


Fig. 1: INSTANS Structure

INSTANS¹ [6] is an incremental engine for near-real-time processing of complex, layered, heterogeneous events. Based on the Rete-algorithm [3], INSTANS performs continuous evaluation of incoming RDF data against multiple SPARQL queries. Intermediate results are stored into a β -node network. When all the conditions of a query are matched, the result is instantly available.

The structure of INSTANS is illustrated in Fig. 1. The system consists of the Rete engine and the input and output connectors, which can interface with the network, triple stores, files or other processes. The Rete engine has four components: 1. Rete network, 2. α -matcher, 3. Rule instance queue, 4. Instance executor. The α -matcher and the Rete network are capable of finding all SPARQL rule conditions satisfying the current set of triples. During runtime the α -matcher receives commands to add and remove triples. The matcher finds the α -nodes of the Rete that match the triples and calls the add or remove methods of those nodes. The changes propagate through the β -network and eventually fully satisfied rule conditions enter the rule nodes, which add new rule instances (with

¹ Incremental eEngine for STANding Sparql, <http://cse.aalto.fi/instans/>

variable bindings) to the rule instance queue. The instance executor executes the rule instances, which causes add and remove triple commands to be fed into the output connectors. The rule instance execution also feeds add and remove triple commands to the α -matcher, resulting in new rule instances. INSTANS operation over an example query is illustrated in Fig. 2. The query selects events occurring between 10 and 11 am. The asynchronous nature of INSTANS means that all input

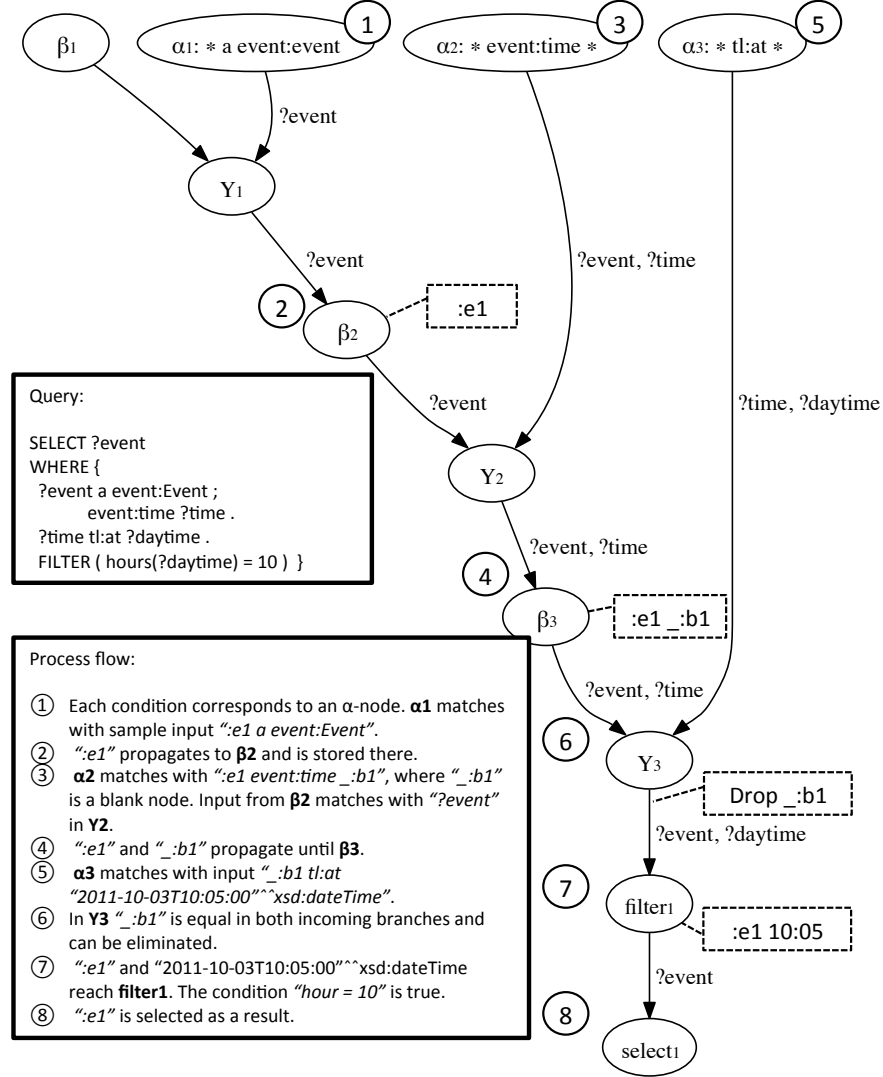


Fig. 2: Example of SPARQL query processing in a Rete-net

is processed when it arrives. To manage periodic actions and missing events, the concept of *timed events* is introduced [7]. When a new timer is started, an *actor* is used to schedule wakeup, at which time a predicate of the timer is changed. A

SPARQL query matching such a triple reacts to the change and carries out the defined actions. No extensions to SPARQL are needed to support timed events.

Performance of INSTANS in terms of notification delay was compared to C-SPARQL [2] using an example application described in [6]. INSTANS yielded average notification delays of 12 ms on a 2.26 GHz Intel Core 2 Duo Mac. In C-SPARQL average query processing delay varied between 12 - 253 ms for window sizes of 5-60 events, respectively, resulting in the window repetition rate being the dominant component of the notification delay for any window repetition rate longer than a second. Using repetition rates of 5-60 seconds with 1 event per second inter-arrival time C-SPARQL notification delay was measured at 1.34-25.90 seconds. Further details are available on the INSTANS project website. Comparison with CQELS [4] is waiting for the availability of a generic version.

3 Conclusions

The feasibility of the central paradigm of INSTANS – continuous incremental matching of multiple SPARQL queries supporting inter-query communication – has so far been supported by empirical tests. When complemented with support for timed events, we have found no showstopper problems which would render the approach unusable for any complex event processing task.

The performance of INSTANS is higher compared to systems based on repeated execution of queries at fixed time intervals (or triple counts); they cannot practically compete with INSTANS whose notification delays are in the order of milliseconds. INSTANS avoids redundant computation: each event is processed immediately on arrival and only once through the Rete network, network structures are shared across similar queries, and intermediate results are memorized.

References

1. Abdullah, H., Rinne, M., Törmä, S., Nuutila, E.: Efficient matching of SPARQL subscriptions using Rete. In: Proceedings of the 27th Symposium On Applied Computing (Mar 2012)
2. Barbieri, D.F., Braga, D., Ceri, S., Grossniklaus, M.: An execution environment for C-SPARQL queries. In: Proceedings of the 13th International Conference on Extending Database Technology - EDBT '10. p. 441. Lausanne, Switzerland (2010)
3. Forgy, C.L.: Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence* 19(1), 17–37 (Sep 1982)
4. Le-Phuoc, D., Dao-Tran, M., Parreira, J.X., Hauswirth, M.: A native and adaptive approach for unified processing of linked streams and linked data. In: ISWC'11. pp. 370–388. Springer-Verlag Berlin (Oct 2011)
5. Rinne, M.: DC Short Paper: SPARQL Update for Complex Event Processing. In: ISWC 2012. Springer-Verlag, Boston, MA (2012)
6. Rinne, M., Abdullah, H., Törmä, S., Nuutila, E.: Processing Heterogeneous RDF Events with Standing SPARQL Update Rules. In: Meersman, R., Dillon, T. (eds.) OTM 2012 Conferences, Part II. pp. 793–802. Springer-Verlag (2012)
7. Rinne, M., Törmä, S., Nuutila, E.: SPARQL-Based Applications for RDF-Encoded Sensor Data. In: 5th International Workshop on Semantic Sensor Networks (2012)

RIO: Minimizing User Interaction in Ontology Debugging

Patrick Rodler, Kostyantyn Shchekotykhin, Philipp Fleiss, and Gerhard Friedrich

Alpen-Adria Universität, Klagenfurt, 9020 Austria
firstname.lastname@aaau.at

Abstract. Interactive ontology debugging incorporates a user who answers queries about entailments of their intended ontology. In order to minimize the amount of user interaction in a debugging session, a user must choose an appropriate query selection strategy. However, the choice of an unsuitable strategy may result in tremendous overhead in terms of time and cost. We present a learning method for query selection which unites the advantages of existing approaches while overcoming their flaws. Our tests show the utility of our approach when applied to a large set of real-world ontologies, its scalability and adequate reaction time allowing for continuous interactivity.

Motivation and Concept:

Efficient ontology debugging is a cornerstone for many activities in the context of the Semantic Web, especially when automatic tools produce (parts of) ontologies such as in the field of ontology matching. Ontology matching aims at generating a set of semantic links, called alignment, between elements of two standalone ontologies describing related domains. The two ontologies together with the produced alignment, called the aligned ontology, may exhibit a very complex fault structure as a consequence of (1) adding many links between the single ontologies at once and since (2) the actual fault may be located in the produced alignment and/or in one or both of the single ontologies, e.g. if a correct link between two concepts “activates” a source of inconsistency in one of the single ontologies. This makes evident that adequate tool assistance in debugging of such ontologies is indispensable.

Ontology debugging deals with the following problem: Given an ontology \mathcal{O} which does not meet postulated requirements R ,¹ the task is to find a subset of axioms in \mathcal{O} , called diagnosis, that needs to be altered or eliminated from the ontology in order to meet the given requirements. Generally, there are many alternative diagnoses for one and the same faulty ontology \mathcal{O} . The problem is then to figure out the single diagnosis, called target diagnosis \mathcal{D}_t , that enables to formulate the target ontology \mathcal{O}_t featuring the user-intended semantics in terms of entailments and non-entailments. The target ontology can be understood as \mathcal{O} minus the axioms of \mathcal{D}_t plus additional axioms $EX_{\mathcal{D}_t}$ which can be added in order to regain desired entailments which might have been eliminated together with axioms in \mathcal{D}_t .

In interactive ontology debugging we assume a user, e.g. the author of the faulty ontology or a domain expert, interacting with an ontology debugging system by answering queries about entailments of the desired target ontology \mathcal{O}_t . Roughly speaking, each query is a set of axioms and the user is queried whether the conjunction of these axioms is entailed by \mathcal{O}_t . Every positively (negatively) answered query constitutes a positive/entailed (negative/non-entailed) test case fulfilled by \mathcal{O}_t . Test cases can

¹ Throughout this work $R = \{\text{consistency, coherency}\}$.

be seen as constraints \mathcal{O}_t must satisfy and are therefore used to gradually reduce the search space for valid diagnoses. Simply put, the overall procedure consists of (1) computing a predefined fixed number of diagnoses (the set of leading diagnoses \mathbf{D} , usually $|\mathbf{D}| \approx 10$) as an approximation of all diagnoses, (2) gathering additional information by querying the user, i.e. adding a positive or negative test case, (3) incorporating this information to cut irrelevant areas off the search space, i.e. eliminating diagnoses not complying with the newly specified test case. This loop is continued until the search space is reduced to a single (target) diagnosis \mathcal{D}_t . The goal is to achieve this with a minimal number of queries to the user.

The best currently known interactive debugging systems pursue active learning strategies for query generation exploiting meta information in terms of fault probabilities of the user who formulates the ontology. Such a system is described in [1] where fault probabilities are used to calculate for each diagnosis the probability of being the target diagnosis. At each step, the query is selected which minimizes the expected entropy of the set of leading diagnoses \mathbf{D} after the query is answered. This means that the expected uncertainty is minimized and the expected information gain is maximized. This entropy-based strategy (ENT) can speed up the debugging procedure if probabilities are specified appropriately, but can also have substantial negative impact on the performance in case of unreasonable probabilities. The problem is that assessment of probabilities is only possible a-posteriori. Consequently, as long as the actual fault is unknown, there is always some risk of suboptimal query selection.

As an alternative, one might prefer to rely on an approach with constant performance which pursues a no-risk strategy without taking into account any meta information. One such strategy is split-in-half (SPL) [1], which selects the query which eliminates half of the leading diagnoses, independent of the answer to the query. In this case, however, possibly well-chosen fault probabilities cannot be exploited, resulting again in inefficient debugging actions. To sum up, the user may choose between a strategy with high potential and high risk and a strategy with no risk and no potential.

Therefore, we introduce a method with high potential and low risk, which can be seen as a hybrid risk optimization method (RIO) exploiting positive aspects of both ENT and SPL. On the one hand, our method takes advantage of the given probabilities as long as good performance is achieved. On the other hand, it gradually gets more independent of meta information if suboptimal behavior is measured. This is accomplished by constantly adapting a reinforcement learning parameter $c \in [0, 0.5]$, which can be seen as the minimal postulated "cautiousness" of the next selected query. The cautiousness of a query is equivalent to its worst case elimination rate w.r.t. the set of leading diagnoses \mathbf{D} . E.g., if $|\mathbf{D}| = 10$ and a query Q_1 eliminates 1 (9) leading diagnoses for positive (negative) answer, the cautiousness of Q_1 is $\frac{1}{10}$, whereas a query Q_2 with 5 (5) has an elimination rate of $\frac{5}{10}$. W.r.t., e.g. $c = 0.3$, Q_1 would be a high-risk-query since it eliminates less than $0.3 * 100\%$ of leading diagnoses in the worst case. Thus, it would be dismissed by RIO as a candidate for the next query. By contrast, Q_2 is a non-high-risk-query as it eliminates more diagnoses than claimed by c anyway. Actually, Q_2 is even a no-risk-query because it eliminates 50% of diagnoses in \mathbf{D} in any case. Given a query Q_3 with 7 (3), we call Q_3 more cautious than Q_1 and less cautious than Q_2 .

More concretely, RIO works as follows: Select the same query Q_{ENT} as ENT would select, if the cautiousness of Q_{ENT} is greater or equal c . Otherwise, select the query with best entropy-measure among all (if more than one) least cautious non-high-risk-queries. In the (rare) situation that no such query exists, select Q_{ENT} . E.g., let current $c = 0.3$

and $Q_{\text{ENT}} = Q_1$, then RIO would select Q_3 since it has cautiousness $0.3 = c$ and is thus the only least cautious non-high-risk-query. After each answered query, the new information is taken into account by updating the diagnosis probabilities according to the Bayesian rule [1]. Additionally, the cautiousness parameter $c \leftarrow c + a$ is adjusted by a value a which is proportional to 0.5 minus the actual achieved elimination rate of the current query. So, for a an elimination of more than half of the leading diagnoses, RIO gets a bonus ($a < 0$) allowing it to take more risk in the next iteration. Otherwise, a penalty is imposed implying more cautious successive behavior.

The following evaluation will demonstrate that, independently of the quality of specified meta information, RIO exhibits superior average performance compared to ENT and SPL w.r.t. the amount of user interaction required. Furthermore, experiments will show that RIO scales well and that the reaction time measured is well suited for an interactive debugging approach.

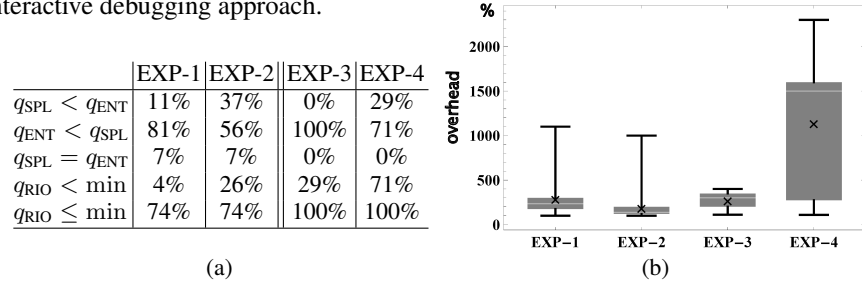


Fig. 1. (a) Percentage rates indicating which strategy performed better w.r.t. number of queries. q_{str} denotes the number of queries needed by strategy str and \min is an abbreviation for $\min(q_{\text{SPL}}, q_{\text{ENT}})$. (b) Box-Whisker Plots presenting the distribution of overhead $(q_w - q_b)/q_b * 100$ (in %) per debugging session of the worse strategy $q_w := \max(q_{\text{SPL}}, q_{\text{ENT}})$ compared to the better strategy $q_b := \min(q_{\text{SPL}}, q_{\text{ENT}})$.

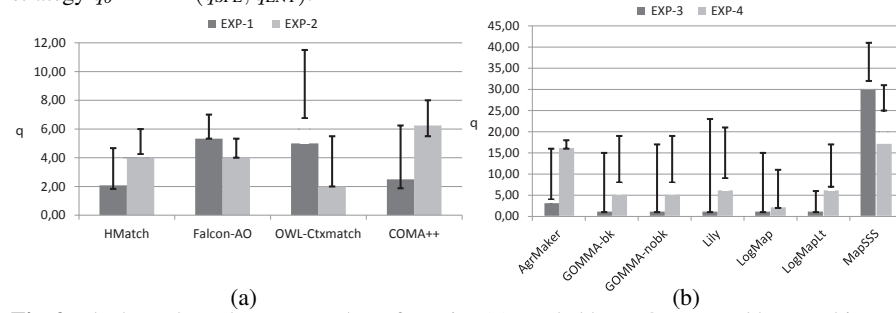


Fig. 2. The bars show the avg. number of queries (q) needed by RIO, grouped by matching tools. The distance from the bar to the lower (upper) end of the whisker indicates the avg. difference between q and the queries needed by the per-session better (worse) strategy q_b (q_w). Notation as in Figure 1.

Evaluation:

We performed four experiments EXP- i ($i = 1, \dots, 4$) where we applied RIO to a set of incoherent ontologies produced by automatic ontology matchers.² As data source for EXP-1 and EXP-2 we used a superset of the dataset³ used in [2] where it was shown

² For details and further results, see <http://code.google.com/p/rmbd/wiki/OntologyAlignmentAnatomy>

³ <http://code.google.com/p/rmbd/downloads>

that existing debugging approaches suffer from serious problems w.r.t. both scalability and correctness of results when tested on this dataset. As an interactive approach able to query and incorporate additional information into its computations, RIO can cope with cases unsolved in [2]. For the scalability tests in EXP-3 and EXP-4, we used the set of ontologies from the ANATOMY track in the Ontology Alignment Evaluation Initiative (OAEI) 2011.5, which comprises two input ontologies \mathcal{O}_1 (Human, 11545 axioms) and \mathcal{O}_2 (Mouse, 4838 axioms). The faulty aligned ontologies had up to 17844 axioms.

Available reference alignments enabled to fix a target diagnosis \mathcal{D}_t for each incoherent ontology. Throughout all experiments, unlike state-of-the-art alignment debuggers, we considered the most general problem where the search for the target diagnosis is not restricted to the alignment. In each test run we measured the number of required queries until \mathcal{D}_t was identified. All tests were executed on a Core-i7 (3930K) 3.2Ghz, 32GB RAM and with Ubuntu Server 11.04 and Java 6 installed. In EXP-1/EXP-3 fault probabilities were chosen reasonably (good case), whereas in EXP-2/EXP-4, they were specified in a way \mathcal{D}_t got very improbable (bad case). Queries were answered by an automatic oracle by means of the target ontology obtained through \mathcal{D}_t .

Results of $\langle \text{EXP-1}, \text{EXP-2} \rangle$ and $\langle \text{EXP-3}, \text{EXP-4} \rangle$, are summarized in Figure 2(a) and Figure 2(b), respectively. The results illustrate clearly that avg. performance achieved by RIO was always substantially closer to the better than to the worse strategy. In both EXP-1 and EXP-2, throughout 74% of 27 debugging sessions, RIO worked as efficiently as the best strategy (Figure 1(a)). In more than 25% of the cases in EXP-2, RIO even outperformed both other strategies; in these cases, RIO could save more than 20% of user interaction on average compared to the best other strategy. In one scenario in EXP-1, it took ENT 31 and SPL 13 queries to finish, whereas RIO required only 6 queries (improvement of $> 80\%$ and 53% , respectively). In $\langle \text{EXP-3}, \text{EXP-4} \rangle$, the savings achieved by RIO were even more substantial (superior behavior to both other strategies in 29% and 71% of cases, respectively). Not less remarkable, in 100% of the tests in EXP-3 and EXP-4, RIO was at least as efficient as the best other strategy. Concerning average number of queries per strategy, RIO is the best choice in all experiments. Consequently, RIO is suitable for both good meta information (EXP-1/EXP-3) and poor meta information (EXP-2/EXP-4). Moreover, assuming a user being capable of reading and answering a query in, e.g., half a minute on average, RIO shows best performance w.r.t. overall debugging time with savings of up to 50% compared to ENT/SPL. Reaction time of RIO, i.e. avg. time between two successive queries, was always $< 12.9\text{s}$. For SPL and ENT strategies, the difference w.r.t. the number of queries per test run between the better and the worse strategy was absolutely significant, with a maximum of 2300% in EXP-4 and averages of 190% to 1145% throughout all four experiments (Figure 1(b)). Moreover, results show that the different quality of probabilities in $\{\text{EXP-1}, \text{EXP-3}\}$ versus $\{\text{EXP-2}, \text{EXP-4}\}$ clearly affected performance of ENT and SPL strategies (Figure 1(a)). This perfectly motivates the application of RIO.

References

1. Shchekotykhin, K., Friedrich, G., Fleiss, P., Rodler, P.: Interactive ontology debugging : two query strategies for efficient fault localization. Web Semantics: Science, Services and Agents on the World Wide Web 12-13, 88–103 (2012)
2. Stuckenschmidt, H.: Debugging OWL Ontologies - A Reality Check. In: Proceedings of the 6th International Workshop on Evaluation of Ontology-based Tools and the Semantic Web Service Challenge (EON). pp. 1–12. Tenerife, Spain (2008)

On direct debugging of aligned ontologies^{*}

Kostyantyn Shchekotykhin, Philipp Fleiss, Patrick Rodler, and Gerhard Friedrich

Alpen-Adria Universität, Klagenfurt, 9020 Austria
firstname.lastname@aau.at

Abstract. Modern ontology debugging methods allow efficient identification and localization of faulty axioms defined by a user while developing an ontology. However, in many use cases such as ontology alignment the ontologies might include many conflict sets, i.e. sets of axioms preserving the faults, thus making the ontology diagnosis infeasible. In this paper we present a debugging approach based on a direct computation of diagnoses that omits calculation of conflict sets. The proposed algorithm is able to identify diagnoses for an ontology which includes a large number of faults and for which application of standard diagnosis methods fails. The evaluation results show that the approach is practicable and is able to identify a fault in adequate time.

1 Motivation and algorithm details

Ontology development and maintenance relies on an ability of users to express their knowledge in form of logical axioms. However, the knowledge acquisition process might be problematic since a user can make a mistake in an axiom being modified or a correctly specified axiom can trigger a hidden bug in an ontology. These bugs might be of a different nature and result in violation of such *requirements* as consistency of an ontology or satisfiability of its classes. In such scenarios as ontology matching the complexity of faults might be very high because multiple disagreements between ontological definitions and/or modeling problems can be triggered by ailments at once.

Ontology debuggers simplify the development by allowing their users specification of requirements to the intended (target) ontology. In addition, a user can provide \mathcal{B} set of *background* (correct) axioms and sets of positive P and negative N test cases. A *positive* test case is a set of axioms that must be entailed by the intended ontology, whereas a *negative* test case must not. If any of the requirements or test cases are broken, i.e. an ontology \mathcal{O} is *faulty*, then the tuple $\langle \mathcal{O}, \mathcal{B}, P, N \rangle$ is a *diagnosis problem instance*. For a given problem instance a debugging tool computes a set of axioms $\mathcal{D} \subseteq \mathcal{O}$ called *diagnosis*. An expert should remove or modify at least *all* axioms of a diagnosis in order to be able to formulate the *target ontology* \mathcal{O}_t .

Nevertheless, in real-world scenarios debugging tools can return a set of alternative diagnoses \mathbf{D} . The reason is the practical impossibility for a user to specify such a set of requirements and test cases, prior to a debugging session, providing all information required for identification of the *target diagnosis* \mathcal{D}_t , i.e. the diagnosis which application allows formulation of the intended ontology. *Diagnosis discrimination* methods allow their users to reduce the number of diagnoses to be considered. An interactive algorithm suggested in [4] identifies the target diagnosis by asking an oracle a sequence of questions: whether some axiom is entailed by the target ontology or not. A general

^{*} This research is funded by Austrian Science Fund (Project V-Know, contract 19996).

interactive ontology diagnosis algorithm can be described as follows: (1) Generate a set of diagnoses \mathbf{D} including at most n diagnoses. (2) Compute a set of queries and select the best one using a predefined measure. (3) Ask the oracle and, depending on the answer, add the query either to P or to N . (4) Update the set of diagnoses \mathbf{D} and remove the ones that do not comply with the newly acquired test case. (5) Update the tree and repeat from Step 1 if the tree contains open nodes. (6) Return the set of diagnoses \mathbf{D} . The resulting set \mathbf{D} includes only diagnoses that are not differentiable in terms of their entailments, but have some syntactical differences. The preferred target diagnosis \mathcal{D}_t , in this case, should be selected by the user using some ontology editor such as Protégé.

Most of the debugging approaches, including [4], follow the standard model-based diagnosis approach [3] and compute diagnoses using *conflict sets* CS , i.e. irreducible sets of axioms $ax_i \in \mathcal{O}$ that preserve violation of at least one requirement. The computation of one conflict set can be done within a polynomial number of calls to the reasoner, e.g. by QUICKXPLAIN algorithm [2]. To identify a diagnosis of cardinality $|\mathcal{D}| = m$ the hitting set algorithm suggested in [3] requires computation of m conflict sets. In some practical scenarios, such ontology matching, the number of conflict sets m can be large, thus making the ontology debugging practically infeasible.

In this paper we present two algorithms INV-HS-TREE and INV-QUICKXPLAIN, which inverse the standard model-based approach to ontology debugging and compute diagnoses directly, rather than by means of conflict sets (see [5] for a detailed description of the algorithms). The main function of the latter algorithm splits recursively the initial diagnosis problem instance into two sub-problems by partitioning the axioms of a given faulty ontology into two subsets. In many cases SPLIT simply partitions the set of axioms into two sets of equal cardinality. The algorithm continues to divide diagnosis problems until it identifies a set \mathcal{D}' such that $\mathcal{O} \setminus \mathcal{D}'$ fulfills all the requirements, but $\mathcal{O} \setminus \mathcal{D}'_i$, where \mathcal{D}'_i are partitions of \mathcal{D}' , not. In further iterations the algorithm minimizes the \mathcal{D}' by splitting it into sub-problems of the form $\mathcal{D}' = \mathcal{D} \cup \mathcal{O}_\Delta$, where \mathcal{O}_Δ contains only one axiom. In the case when \mathcal{D}' is a diagnosis and \mathcal{D} is not, the algorithm decides that \mathcal{O}_Δ is a subset of the sought diagnosis. Just as the original algorithm, INV-QUICKXPLAIN always terminates and returns either a diagnosis \mathcal{D} or “no diagnosis”.

In order to enumerate all possible diagnoses we modified the HS-TREE algorithm [3] to accept diagnoses as node labels instead of conflict sets. The INV-HS-TREE algorithm constructs a directed tree from root to the leaves, where each node nd is labeled either with a diagnosis $\mathcal{D}(nd)$ or \checkmark (closed) or \times (pruned). The latter two labels indicate that the node cannot be extended. Each edge outgoing from the open node nd is labeled with an element $s \in \mathcal{D}(nd)$. $HS(nd)$ is a set of edge labels on the path from the root to the node nd . Initially the algorithm creates an empty root node and adds it to the *queue*, thus, implementing a breadth-first search strategy. Until the queue is empty, the algorithm retrieves the first node nd from the queue and labels it with either:

1. \times if there is a node nd' , labeled with either \checkmark or \times , such that $H(nd') \subseteq H(nd)$ (pruning non-minimal paths), or
2. $\mathcal{D}(nd')$ if a node nd' exists such that its label $\mathcal{D}(nd') \cap H(nd) = \emptyset$ (reuse), or
3. \mathcal{D} if \mathcal{D} is a diagnosis for the diagnosis problem instance $\langle \mathcal{O}, \mathcal{B} \cup H(nd), P, N \rangle$ computed by INV-QUICKXPLAIN (compute), or
4. \checkmark (closed).

The leaf nodes of a complete tree are either pruned (\times) or closed (\checkmark) nodes.

In the diagnosis discrimination settings the ontology debugger acquires new knowledge that can invalidate some of the diagnoses labeling the tree nodes. During the tree

update INV-HS-TREE searches for the nodes with invalid labels, removes its label and places it to the list of open nodes. Moreover, the algorithm removes all the nodes of a subtree originating from this node. After all nodes with invalid labels are cleaned-up, the algorithm attempts to reconstruct the tree by reusing the remaining valid minimal diagnoses (rule 2, INV-HS-TREE). Such aggressive pruning of the tree is feasible since a) the tree never contains more than n nodes that were computed with INV-QUICKXPLAIN and b) computation of a possible modification to the minimal diagnosis, that can restore its validness, requires invocation of INV-QUICKXPLAIN and, therefore, as hard as computation of a new diagnosis. Note also, that in a common diagnosis discrimination setting n is often set to a small number, e.g. 10, in order to achieve good responsiveness of the system. In the direct approach this setting limits the number of calls to INV-QUICKXPLAIN to n and results in a small size of the search tree. The latter is another advantage of the direct method as it requires much less memory in comparison to a debugger based on original HS-TREE.

2 Evaluation

We evaluated the direct ontology debugging technique using aligned ontologies generated in the framework of OAEI 2011 [1]. These ontologies represent a real-world scenario in which a user generated ontology alignments by means of (semi-)automatic tools. All ontologies used in the experiments are available online, together with detailed evaluation results¹.

In the first experiment we applied the debugging technique to the set of aligned ontologies resulting from “Conference” set of problems, which is characterized by lower precision and recall of the applied systems (the best F-measure 0.65) in comparison, for instance, to the “Anatomy” problem (average F-measure 0.8). The Conference test suite includes 286 ontology alignments generated by 14 ontology matching systems including: a) 140 ontologies are consistent and coherent; b) 122 ontologies are incoherent; c) 26 ontologies are inconsistent; and in 8 cases a reasoner was unable to classify in two hours². Note that only two systems CODI and MaasMatch out of 14 were able to generate consistent and coherent alignments. This observation confirms the importance of high-performance ontology debugging methods.

The 146 ontologies of the cases b) and c) were analyzed with both HS-TREE and INV-HS-TREE. The results of the experiment show that for 133 ontologies out of 146 both approaches were able to compute the required amount of diagnoses. HT-TREE required 49, 61 and 80 sec. on average to find 1, 9 and 30 leading minimal diagnoses. The direct algorithm required less time and returned the requested number of diagnoses on average in 27, 52 and 72 sec. respectively. In the 13 cases HS-TREE was unable to find all requested diagnoses in each experiment. Within 2 hours the algorithm calculated only 1 diagnosis for *csa-conference-ekaw* and for *ldoa-conference-confot* it was able to find 1 and 9 diagnoses, whereas INV-HS-TREE required 9 sec. for 1, 40 sec. for 9 and 107 sec. for 30 diagnoses on average.

Moreover, in the first experiment we evaluated the efficiency of the interactive direct debugging approach applied to the cases listed in Table 1. We selected the target diagnosis randomly among all diagnoses of the following diagnosis problem instance $\langle M_f, \mathcal{O}_i \cup \mathcal{O}_j \cup M_t, \emptyset, \emptyset \rangle$, where M_f and M_t are the sets of *false* and *true* positive

¹ <http://code.google.com/p/rmbd/wiki/DirectDiagnosis>

² Core-i7 (3930K) 3.2Ghz, 32GB RAM running Ubuntu 11.04, Java 6 and HetmiT 1.3.6.

Matcher	Ontology 1	Ontology 2	Time (sec)	#Query	React (sec)	#CC	CC (ms)
ldoa	conference	confof	11.6	6	1.4	430	3
ldoa	cmt	ekaw	48.5	21	2.2	603	16
mappso	confof	ekaw	9.9	5	1.8	341	7
optima	conference	ekaw	16.7	5	2.5	553	8
optima	confof	ekaw	23.9	20	1.1	313	14
ldoa	conference	ekaw	56.6	35	1.4	253	53
csa	conference	ekaw	6.7	2	2.7	499	3
mappso	conference	ekaw	27.4	13	1.8	274	28
ldoa	cmt	edas	24.7	22	1.1	303	8
csa	conference	edas	18.4	6	2.7	419	5
csa	edas	iasted	1744.6	3	349.2	1021	1333
ldoa	ekaw	iasted	23871.4	10	1885.9	287	72607
mappso	edas	iasted	18400.2	5	2028.2	723	17844

Table 1. Diagnosis discrimination using direct ontology debugging and Entropy scoring function. **React** time required to compute 9 diagnoses and a query, **#CC** number of consistency checks, **CC** gives average time needed for one consistency check.

alignments computed from the set of correct alignments M_c , provided by the organizers of OAEI 2011, and the set M_{ij} generated by a ontology matching system. In the experiment the used the Entropy scoring function [4] with prior fault probabilities of axioms corresponding to ailments set to $1 - v$, where v is the confidence value of the matcher for an axiom. All axioms of \mathcal{O}_i and \mathcal{O}_j were assumed to be correct and were assigned small probabilities. The results presented in Table 1 were computed for the diagnosis problem instance $\langle M_{ij}, \mathcal{O}_i \cup \mathcal{O}_j, \emptyset, \emptyset \rangle$. The experiment shows that the system was able to identify the target diagnosis efficiently with small reaction times. The system's performance decreased only in the cases when a reasoner required much time to verify the consistency of an ontology.

In the second scenario we applied the direct method to unsatisfiable and classifiable within 2 hours ontologies, generated for the Anatomy problem. The source ontologies \mathcal{O}_1 and \mathcal{O}_2 include 11545 and 4838 axioms correspondingly, whereas the size of the alignments varies between 1147 and 1461 axioms. The target diagnosis selection process was performed in the same way as in the first experiment. The results of the experiment show that the target diagnosis can be computed within 40 second in an average case. Moreover, INV-HS-TREE slightly outperformed HS-TREE.

References

1. Euzenat, J., Ferrara, A., van Hage, W.R., Hollink, L., Meilicke, C., Nikolov, A., Ritze, D., Scharffe, F., Shvaiko, P., Stuckenschmidt, H., Sváb-Zamazal, O., dos Santos, C.T.: Final results of the Ontology Alignment Evaluation Initiative 2011. In: Proceedings of the 6th International Workshop on Ontology Matching. pp. 1–29. CEUR-WS.org (2011)
2. Junker, U.: QUICKXPLAIN: Preferred Explanations and Relaxations for Over-Constrained Problems. In: Proc. 19th National Conference on Artificial Intelligence. pp. 167–172 (2004)
3. Reiter, R.: A Theory of Diagnosis from First Principles. Artificial Intelligence 32(1), 57–95 (1987)
4. Shchekotykhin, K., Friedrich, G., Fleiss, P., Rodler, P.: Interactive ontology debugging : two query strategies for efficient fault localization. Web Semantics: Science, Services and Agents on the World Wide Web 12–13, 88–103 (2012)
5. Shchekotykhin, K., Friedrich, G., Fleiss, P., Rodler, P.: Direct computation of diagnoses for ontology debugging. arXiv 1–16 (2012) <http://arxiv.org/abs/1209.0997>

Queries, the Missing Link in Automatic Data Integration

Aibo Tian, Juan F. Sequeda, and Daniel P. Miranker

Department of Computer Science, The University of Texas at Austin
Austin, Texas, USA

atian@utexas.edu, {jsequeda, miranker}@cs.utexas.edu

Abstract. This paper introduces the ontology mapping approach of a system that automatically integrates data sources into an ontology-based data integration system (OBDI). In addition to the target and source ontologies, the mapping algorithm requires a SPARQL query to determine the ontology mapping. Further, the mapping algorithm is dynamic: running each time a query is processed and producing only a partial mapping sufficient to reformulate the query. This approach enables the mapping algorithm to exploit query semantics to correctly choose among ontology mappings that are indistinguishable when only the ontologies are considered. Also, the mapping associates paths with paths, instead of entities with entities. This approach simplifies query reformulation. The system achieves favorable results when compared to the algorithms developed for Clio, the best automated relational data integration system.

We have developed an Ontology-based Data Integration (OBDI) system that departs from the conventional OBDI organization. The goal is to include automatic integration of new data sources, provided those data sources publish a self-describing ontology. A consequence of that goal is there is no longer the opportunity for an engineer to review and correct an ontology matching prior to its use by the query reformulation system. As ontology matching is understood to be an uncertain process, some other method of mapping refinement is needed. Our system uses queries for this purpose.

Ontology mapping in conventional OBDI systems is determined prior to, and without knowledge of the queries to be executed [3]. A static representation of a mapping between target and source ontologies serves as input to a query reformulation module (Fig. 1(a)). In the system described here, ontology mapping is a dynamically computed component whose result depends on the query that is being processed (Fig. 1(b)). In effect, the query becomes a third argument to the ontology mapping algorithm. The query provides context for selecting among competing mappings. Since a mapping is specific to a query, the results may be limited to the partial mapping required by the query reformulation system.

The organization was motivated by the following observations. A mapping method may determine that an entity in one ontology maps with equal likelihood to two or more entities in the other ontology. The mapping and reformulation of certain queries is correct only if one pairing is chosen. The correct choice may be different for different queries. The query itself may lend additional semantics that correctly resolve the ambiguity.

These observations are supported by the example in Fig. 2. Looking at the ontologies alone, there is insufficient information to determine if the class *T:People* should

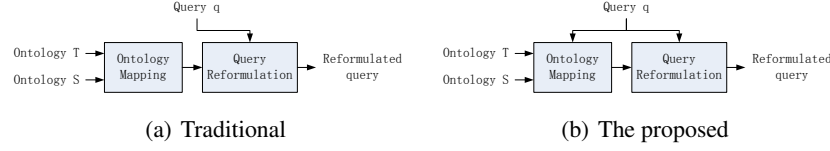


Fig. 1. OBDI systems with the traditional and the proposed ontology mapping component.

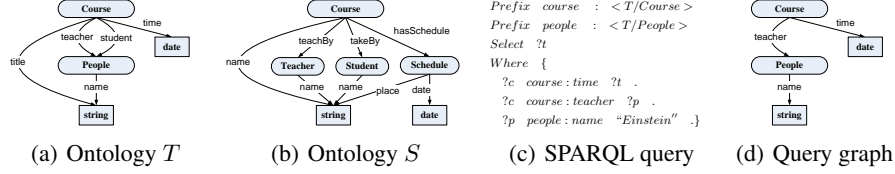


Fig. 2. Example ontologies and SPARQL query.

be mapped to $S:Teacher$ or to $S:Student$. A third possibility is a one-to-many mapping entailing both. However, given the SPARQL query in Fig. 2(c), which asks for the time of the course that is offered by “Einstein”, it is clear that $T:People$ should only be mapped to $S:teacher$. A complementary query addressing student enrollment requires $T:People$ to be mapped to $S:student$. Correct answers from a reformulated query can be achieved only through mutually exclusive query dependent mappings.

An overview of the matching algorithm is as follows. To exploit the context implicit in a query, the algorithm maps paths in the target ontology to paths in a source ontology. A path contains a datatype and multiple classes connected by properties. Each element can be considered as the context of other elements in the path. For example, if a path contains a class *People*, and a property *teaches*, then we can infer that *People* is a *Teacher* instead of a *Student*. It follows that not all entities in one path have a corresponding entity in the corresponding path. Path-based mapping must necessarily accommodate this. Since the number of unconstrained paths in a graph is much larger than the number of vertices, the properties suggest that path mapping is a combinatorially much harder problem. However, as the organization stipulates that ontology mapping is dynamic and specific to a query. Thus, the mapping may be limited to only those mappings required to reformulate the query. Relative to ontologies, queries are very small, and only the paths in the target ontology corresponding to the query need to be mapped. These constraints limit the search problem to a manageable size.

Consider the specific SPARQL query in Fig. 2(c). Fig. 2(d) illustrates the part of ontology that corresponds to q , which is called a query graph. The task is to generate mappings that can be used to reformulate q in terms of ontology S . The algorithm must address the following challenges. The class *People* in T can be mapped to *Teacher* or *Student* in S , and some entities, such as class *Schedule* in S , do not have any mapped entity in T . However, the mapping for *Schedule* is necessary to reformulate the query.

All of these challenges are met by mapping paths, represented as sequences of labels, where the sequence comprises alternating vertex and edge labels. In the example, query q has two paths in its query graph:

$\{T:Course, T:teacher, T:People, T:name, string\}$ and $\{T:Course, T:time, date\}$

We search for a subgraph with two paths in ontology S , which have the highest probability such that each of the paths in S are mapped to paths that correspond to the query graph of q . The probability of each path mapping is determined by scoring the similarity of all labels in the paths.

Mapping results should be:

$$\begin{aligned} &\{T:Course, T:teacher, T:People, T:name, string\} \\ &= \{S:Course, S:teachBy, S:Teacher, S:name, string\} \\ &\{T:Course, T:time, date\} \\ &= \{S:Course, S:hasSchedule, S:Schedule, S:date, date\} \end{aligned}$$

Note that this mapping is specific to the query. If another query asks for the time of the course that is taken by “Einstein”, the mapped path should contain $S:Student$ instead of $S:Teacher$. Thus, defining similarity to a sequence of labels identified by the query introduces context. Limiting the problem to the paths in the query graph not only limits the size of the computation, but also removes any consideration of potentially conflicting interpretation. Given that sequence matching is an endemic problem in genomic data processing, there are many avenues open to exploration.

Experimental Setup: The evaluation comprises real world ontologies from bibliography domain. DBLP ontology is generated by direct mapping the relational schema of the DBLP metadata database using Ultrawrap [2], and the UMBC ontology is from the OAEI benchmark track¹. These ontologies can be found on our website². We manually generated groundtruth mappings between paths. Subsequently, a computer program systematically generates two kinds of SPARQL queries for each ontology. (1) A *PathOnly* query has query graph consisting of only one path in the groundtruth mappings. (2) A *ClassAll* query has query graph consisting of the set of all paths that share a same source in the groundtruth mappings. For each query, the generated path mappings are evaluated by comparing to the groundtruth mappings. The mapping results are evaluated by three metrics. *valid_rate* measures whether the generated mappings contain all paths in a query, regardless of correctness. *path_precision* measures the correctness of individual path mapping. *query_precision* measures whether all path mappings are correct for a query.

Baseline: Clío is a semi-automatic relational schema mapping system, however, the resulting algorithms are applicable to ontologies [1]. We implemented multiple configurations of Clío as baselines. All baselines first generate mappings between datatype properties by picking the ones with highest similarities. Given a query, the baselines find the mapping candidates that contain all the mapped datatype properties. If there exists more than one candidates, Clío asks a user to make the decision, which is not allowed in our automatic setting. We implement three baselines to approximate this process: *clio-minimal*, *clio-maximal*, and *clio-similar*, which chooses the mapping candidate with minimal summation of path lengths, maximal summation of path lengths, and

¹ <http://oaei.ontologymatching.org>

² <http://www.cs.utexas.edu/~atian/page/dataset.html>

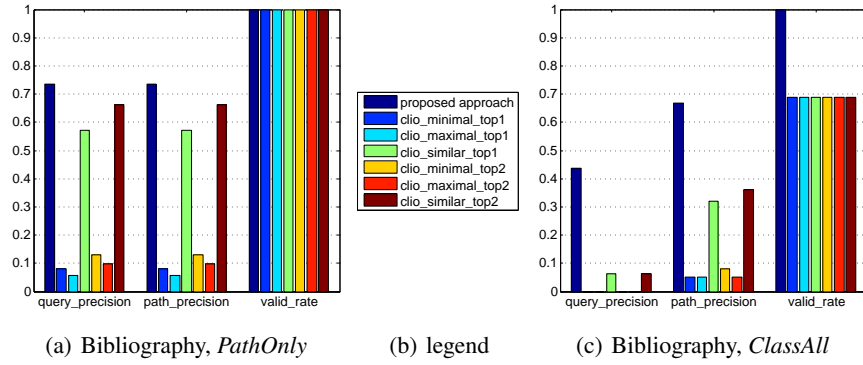


Fig. 3. The evaluation results for bibliography data sets.

the highest similarity between the sources of the paths. We also enhance the baselines, by keeping 2 mappings instead of 1 for each datatype property. We generate mapping for each of them, and consider the mapping as correct if any of them is correct.

Results: Overall, our approach dominates over all baselines as shown in Fig. 3. For *PathOnly* queries, *query_precision* and *path_precision* are the same, and all approaches have 100% *valid_rate*, because each query only consists of one path in the query graph. In terms of *query_precision* and *path_precision*, our approach is 0.15 higher than the best top1 baselines. The three top2 baselines are improved with respect to the top1 approaches. However, even though the top2 approaches choose the correct mapping from large number of candidate mappings, it would still not yield a mapping with higher precision than our approach. *clio_similar* has better performance comparing to *clio_minimal* and *clio_maximal*. This indicates that the similarity between sources is important to the path mapping. For *ClassAll* queries, none of the baselines are competitive with respect to our approach. This is because the baselines determine the datatype property and source mappings first, and only use query to find valid path mappings. In some cases, the valid mappings are not existed. For our approach, the path mappings are jointly determined by both entity mappings and the query, so we have higher chances to find valid correct mappings.

References

1. R. Fagin, L. Haas, M. Hernández, R. Miller, L. Popa, and Y. Velegrakis. Clio: Schema mapping creation and data exchange. *Conceptual Modeling: Foundations and Applications*, pages 198–236, 2009.
2. J. F. Sequeda and D. P. Miranker. Ultrawrap: Sparql execution on relational data. Technical Report TR-12-10, University of Texas at Austin, Department of Computer Sciences, 2012.
3. H. Wache, T. Voegelé, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner. Ontology-based integration of information—a survey of existing approaches. In *IJCAI-01 workshop: ontologies and information sharing*.

MeDetect: Domain Entity Annotation in Biomedical References Using Linked Open Data

Li Tian¹, Weinan Zhang¹, Haofen Wang¹, Chenyang Wu¹

Yuan Ni², Feng Cao², Yong Yu¹

¹Shanghai Jiao Tong University, Shanghai; ²IBM China Research Laboratory, Beijing
¹{tianli,wnzhang,whfcarter,wucy,yyu}@apex.sjtu.edu.cn; ²{niyuan,caofeng}@cn.ibm.com

Abstract. Recently, with the ever-growing use of textual medicine records, annotating domain entities has been regarded as an important task in the biomedical field. On the other hand, the process of interlinking open data sources is being actively pursued within the Linking Open Data (LOD) project. The number of entities and the number of properties describing semantic relationships between entities within the linked data cloud are very large. In this paper, we propose a knowledge-incentive approach based on LOD for entity annotation in the biomedical field. With this approach, we implement MeDetect, a prototype system to solve the problems mentioned above. The experimental results verify the effectiveness and efficiency of our approach.

Keywords: Domain Entity Annotation, Linked Open Data

1 Introduction

Entity annotation aims at discovering entities in references automatically. It is quite useful for many tasks including information extraction, classification, text summarization, question answering, and literature-based knowledge discovery. On the other hand, the Web as a global information space is developing from a Web of documents to a Web of data. Currently, there are billions of triples publicly available in Web data sources of different domains. These data sources are becoming more tightly interrelated as the number of links in the form of mappings grows. Based on the two points, what have we done in this paper can be summarized as follows.

1. We have proposed a novel knowledge-incentive approach based on LOD for entity annotation in the biomedical field. This approach has data flexibility, language independence, and semantic relationship enrichment, which makes it more convenient and informative for further applications.
2. We have proposed to make use of collective annotation leveraged by LOD information to conduct entity filtering and disambiguation.

3. We have developed MeDetect to implement our proposal. The experimental results verify the effectiveness and efficiency of our approach.

2 Methods

The overall design of MeDetect is shown in Figure 1.

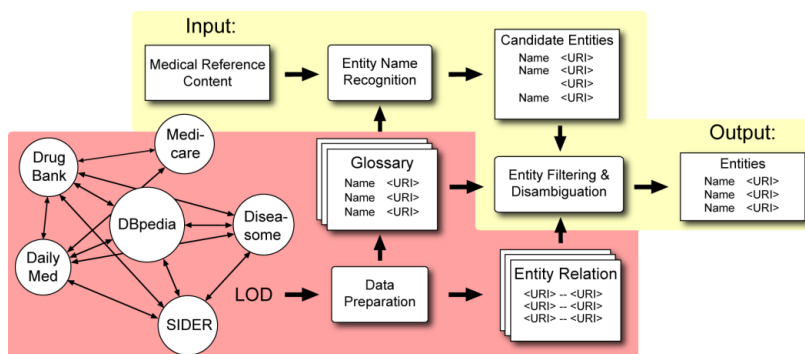


Fig.1. The architecture of MeDetect

Data Preparation. This step is conducted off-line. It generates two data structures for the on-line process of MeDetect. The first is an entity glossary, and the second is entity relation data. For Biomedical related ontology like DrugBank, we just use its entities and relations to build the entity glossary and relation data. For other general ontology like DBpedia which contains other non-biomedical entities, we propose a two-stage approach to handle it. In the first stage we use the link `owl:sameAs` from LODD to DBpedia to find the seed entities and expand the entity set from the original seed set using the sharing category information between entities. Secondly, we use a Support Vector Machine (SVM) [1] as the binary classification algorithm to pick biomedical entities from the candidate set.

Entity Name Recognition. With the biomedical entity glossary, the on-line entity name recognition step provides the syntactic match between the entity name in the glossary and the content of input biomedical references. This is based on syntactic matching, and the recognized entities with their URIs are passed to next step as candidate entities to be further filtered.

Entity Filtering and Disambiguation. The last but most important on-line step of MeDetect is entity filtering and disambiguation. We implement a system based on recent work on Web page annotation, Collective Annotation [2]. This approach not only detects the importance of each entity for the input text, but also, more importantly, filters out irrelevant entities based on the inter-entity relationship. There should be two functions in collective annotation: the single entity importance function and entity pair coherence function.

The single entity importance function estimates the relevance between an entity and the input text, based on their syntactic and semantic similarity, using logistic regression or category-based matching. Here, the entity description information in its URI can be utilized to match the input text.

The entity pair coherence function judges the topic similarity or consistency of pairs of entity URIs so as to filter out noise and cope with ambiguities. For example, if a candidate entity has no relationship or common topic with others, it is quite possible that this candidate is noise. Also if a candidate entity name has more than one URI, the entity pair coherence function will calculate the coherence of each of these URI with the ones of other entities and choose the most coherent one as the final URI of this entity name. Thus the problem of ambiguity is handled. In MeDetect, we use a LOD neighborhood overlap calculation [3] to implement the entity pair coherence function, as is shown in Figure 2.

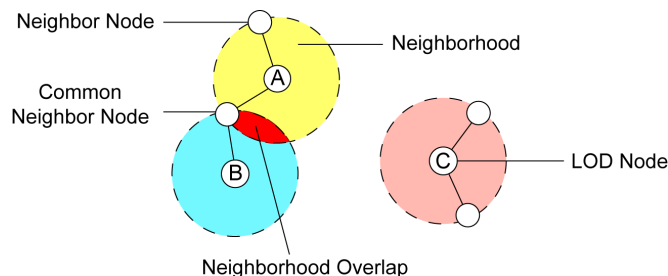


Fig.2. LOD neighborhood overlap calculation in collective annotation of MeDetect

Finally, we show a case of MeDetect entity annotation for a piece of biomedical reference in Figure 3. With the URI of each extracted entity, further information (such as the description of each entity and its links to related entities) can be directly provided in the annotation service.

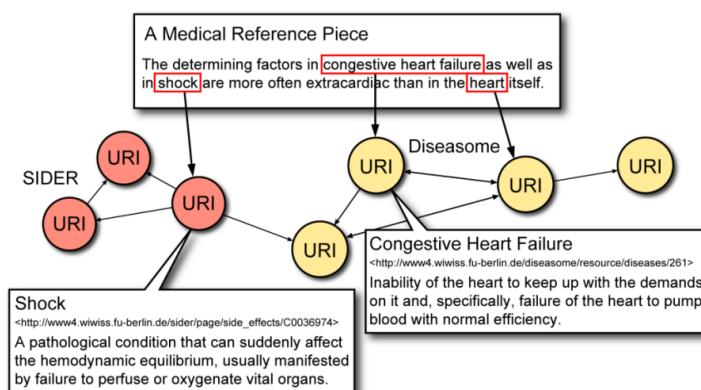


Fig.3. An Example of MeDetect Entity Annotation

3 Results

The effectiveness and efficiency of MeDetect is evaluated by an experimental study. In the experiment, 120 paper abstracts with different biomedical topics are randomly selected from PubMed and three human judges with biomedical or computer background score the output entities for each paper abstract. To have a comparative evaluation of the quality of MeDetect, we import MetaMap and LingPipe. Here MeDetect\FD means MeDetect without filtering and disambiguation.

Compared systems	#Test references	#Output entities	#Corrected entities	Average accuracy	Average running time
MeDetect	120	598	455	76.1%	20.2ms
MeDetect\FD	120	683	468	68.5%	11.9ms
MetaMap	120	1,062	412	35.4%	601.4ms
LingPipe	120	782	510	65.2%	69.3ms

Table 1. Performance comparison among MeDetect, MeDetect\FD, MetaMap, and LingPipe

In Table 1 the average accuracy of MeDetect is much higher than MetaMap and LingPipe. Without entity filtering and disambiguation, MeDetect\FD provides a lower accuracy, despite its higher efficiency. In sum, MeDetect provides the most satisfactory performance.

4 Conclusion

This paper describes a novel knowledge-incentive approach based on LOD for entity annotation in the biomedical field. This approach has data flexibility, language independence, and semantic relationship enrichment, which makes it more adaptive and informative for further applications. We implement a prototype system MeDetect to demonstrate our approach for domain entity annotation for biomedical references. Its three key components are data preparation, entity name recognition, and entity filtering and disambiguation. Our system demonstrates its high annotation accuracy and data flexibility for adding more LOD sources. In future work, we will enrich the entity glossary of MeDetect by adding more LOD sources. More importantly, MeDetect will be further utilized for triple extraction from biomedical references.

References

1. Suykens J.A.K. and Vandewalle J. Least Squares Support Vector Machine Classifiers. Neural Processing Letters 1999.
2. Kulkarni S., Singh A., Ramakrishnan G. and Chakrabarti S. Collective Annotation of Wikipedia Entities in Web Text. SigKDD Proc. 2010.
3. Zhou W., Wang H., Chao J., Zhang W. and Yu Y. LODDO: Using Linked Open Data Description Overlap to Measure Semantic Relatedness Between Named Entities. JIST Proc. 2011.

Towards Licenses Compatibility and Composition in the Web of Data

Serena Villata* and Fabien Gandon

INRIA Sophia Antipolis, France
{firstname.lastname}@inria.fr

Abstract. We propose a general framework to attach the licensing terms to the data where the compatibility of the licensing terms concerning the data affected by a query is verified, and, if compatible, the licenses are combined into a composite license. The framework returns the composite license as licensing term about the data resulting from the query.

1 Introduction

The absence of clarity concerning the licensing terms does not encourage the reuse of the data in the Web of Data [3]. When consumers query the Web of Data, results from different datasets, and thus released under different licensing terms, are provided. In this paper, we propose first to verify the compatibility among the licensing terms associated to a query result, and second, to compose, if compatible, the distinct licensing terms for creating a composite license. The composite license is returned together with the query result using the standard SPARQL query results XML format by means of the `<link>`¹ element. We adopt Semantic Web languages only, and reuse the Creative Commons (CC) licenses schema [1] to define the anatomy of our licenses. Licenses are composed by *models*: `cc:Permission`, `cc:Requirement`, and `cc:Prohibition`. Models are composed by *elements* el_i like `ShareAlike`, `Attribution`, and many others. We choose CC because it provides a general schema for licenses specification, even if there are works which should not be released under the CC licenses [5, 3]. For addressing this issue and covering a wider range of machine-readable license specifications, we align the CC vocabulary with the other schemas including licensing terms (Figure 1). We extend and adapt existing proposals for licenses compatibility and composition in the area of service license analysis [2] to the Web of Data scenario. However, the different application scenarios (service composition vs Web of Data) open different problems. The compatibility rules we define are different, and the definition of the composite license mirrors the same differences. Truong et al. [6] address the issue of analyzing data contracts using RDF for the contracts representation. This work concentrates on data contracts and not on data licenses. Krotzsch and Speiser [4] present a semantic framework for evaluating *ShareAlike* recursive statements while we address the problem of licenses composition.

* The author acknowledges the support of the DataLift Project ANR-10-CORD-09.

¹ <http://www.w3.org/TR/rdf-sparql-XMLres/#head>

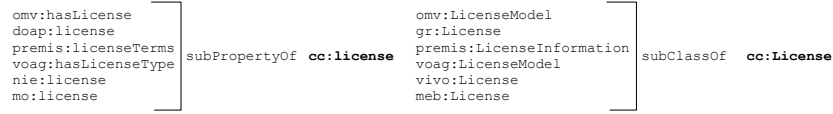


Fig. 1: Alignment between the Creative Commons vocabulary and other vocabularies.

2 Our proposal

Licenses Compatibility. We define a set of compatibility rules assessing the possible compatibility among the elements composing the licenses \mathcal{L} , following [2] for service licenses. First, there are certain elements which are broader in scope of permission than other elements, e.g., *Sharing* is more permissive than *Reproduction* [1]. The subsumption rules² for **cc:Permission** elements, and the way they can be combined are shown in Table 1. Table 2.a shows whether an element el_1 is compatible with another element el_2 , i.e., $el_1 \bigcirc el_2$, concerning **cc:Permission** elements³. The rationale is that these elements are compatible if there is a subsumption relation between them.

Subsumption	More permissive	Less permissive
<i>DerivativeWorks</i> \succ <i>Sharing</i>	<i>DerivativeWorks</i>	<i>Sharing</i>
<i>Distribution</i> \succ <i>Reproduction</i>	<i>Distribution</i>	<i>Reproduction</i>
<i>DerivativeWorks</i> \succ <i>Reproduction</i>	<i>DerivativeWorks</i>	<i>Reproduction</i>
<i>Sharing</i> \succ <i>Reproduction</i>	<i>Sharing</i>	<i>Reproduction</i>
<i>DerivativeWorks</i> \succ <i>Distribution</i>	<i>DerivativeWorks</i>	<i>Distribution</i>

Table 1: Compatibility rules for subsumption relation among **cc:Permission** elements.

Second, a possible situation in analyzing license compatibility is that one license \mathcal{L}_i specifies clauses which are not specified by the other license \mathcal{L}_j , e.g., \mathcal{L}_i specifies **cc:Prohibition** and \mathcal{L}_j does not specify this clause. Table 2.b shows the compatibility rules for specified elements against *Unspecified* elements⁴. The requirement for specification of *Attribution* does not affect the compatibility with *Unspecified* (the same holds for *Notice*, *SourceCode*, and *CopyLeft*). Concerning prohibitions, these elements are not compatible with *Unspecified*, e.g., commercial use is the default setting of the licenses, thus we cannot assume compatibility if commercial use is denied by *NonCommercial*. For permissions, we follow the “conservative” approach where unspecification means a denial of compatibility⁵.

² Subsumption \succ means that there is a compatibility if a certain license element el_i is more permissive, i.e., it accepts more, than the other license element el_j .

³ Rules are expressed under the form of truth tables where elements are evaluated as compatible T , or incompatible F .

⁴ We interpret unspecified elements as “do not care”, as in [2].

⁵ Table 2.c shows three exceptions of compatible elements from distinct models.

el_1	el_2	$el_1 \odot el_2$	el_1	el_2	$el_1 \odot el_2$
<i>Sharing</i>	<i>DerivativeWorks</i>	<i>T</i>	<i>Notice</i>	<i>Unspecified</i>	<i>T</i>
<i>Reproduction</i>	<i>Distribution</i>	<i>T</i>	<i>Attribution</i>	<i>Unspecified</i>	<i>T</i>
<i>Reproduction</i>	<i>DerivativeWorks</i>	<i>T</i>	<i>ShareAlike</i>	<i>Unspecified</i>	<i>T</i>
<i>Reproduction</i>	<i>Sharing</i>	<i>T</i>	<i>SourceCode</i>	<i>Unspecified</i>	<i>T</i>
<i>Distribution</i>	<i>DerivativeWorks</i>	<i>T</i>	<i>CopyLeft</i>	<i>Unspecified</i>	<i>T</i>
<i>Sharing</i>	<i>Distribution</i>	<i>F</i>	<i>NonCommercial</i>	<i>Unspecified</i>	<i>F</i>
(a)			<i>HighIncomeNationUse</i>	<i>Unspecified</i>	<i>F</i>
el_1	el_2	$\odot \mathcal{L}_1 \odot \mathcal{L}_2$	<i>Reproduction</i>	<i>Unspecified</i>	<i>F</i>
<i>Attribution</i>	<i>ShareAlike</i>	<i>T</i> $el_1 \wedge el_2$	<i>Distribution</i>	<i>Unspecified</i>	<i>F</i>
<i>Attribution</i>	<i>NonCommercial</i>	<i>T</i> $el_1 \wedge el_2$	<i>DerivativeWorks</i>	<i>Unspecified</i>	<i>F</i>
<i>ShareAlike</i>	<i>NonCommercial</i>	<i>T</i> $el_1 \wedge el_2$	<i>Sharing</i>	<i>Unspecified</i>	<i>F</i>
(c)			(b)		

Table 2: (a) Compatibility rules among **cc:Permission** elements, (b) Compatibility rules among **cc:Requirement**, **cc:Prohibition**, **cc:Permission** elements against *Unspecified*, (c) Composition rules among **cc:Requirement** and **cc:Prohibition** elements.

Let $\mathcal{L}(C)$ be the set of licenses associated to the named graphs affected by the consumer's query, we say that two licenses are compatible if the models in both the licenses are compatible [7]. The models are compatible if (i) the models are the same, (ii) the models are composed by elements which satisfy the compatibility rules (Table 2.c), and (iii) their elements are compatible. The elements are compatible if (i) the elements are the same, (ii) the elements satisfy the subsumption rules (Table 1), (iii) the elements satisfy the compatibility rules against *Unspecified* (Table 2.b), and (iv) the elements satisfy the compatibility rules (Table 2.a-c). If the licenses are not compatible, then we leave to the data provider to decide the strategy to deal with this situation, e.g., the data is returned together with the more constraining license among $\mathcal{L}(C)$.

Licenses Composition. If the licenses are compatible then we compose them such that the resulting composite license \mathcal{L}_c (\odot is the composition relation) satisfies the following properties: \mathcal{L}_c can be generated only if all the licenses composing it are compatible, and \mathcal{L}_c is consistent with the set of licenses used to compose it. The definition of \mathcal{L}_c is achieved through the definition of (i) redefinition rules to be applied in case a subsumption relation holds (Table 1), (ii) composition rules necessary to maintain the consistency of \mathcal{L}_c w.r.t. $\mathcal{L}(C)$ (Table 2.c), and (iii) heuristics to compose the elements of each license into \mathcal{L}_c . We consider three basic heuristics: *OR-composition*: $\forall l \in \mathcal{L}_i$ then $l \in \mathcal{L}_c$; *AND-composition*: if $\exists l \in \mathcal{L}_1 \wedge \dots \wedge \mathcal{L}_n$ then $l \in \mathcal{L}_c$; *Constraining-value*: most constraining $l \in \mathcal{L}(C)$ is included in \mathcal{L}_c . We leave to the data provider the choice of her best strategy for composing the licenses, e.g., AND-composition typically leads to a shorter and simpler license, while OR-decomposition leads to a more complex license where all the clauses in $\mathcal{L}(C)$ are listed. For example, assume we want to analyze two licenses \mathcal{L}_1 and \mathcal{L}_2 (Figure 2.a-b). We first compare the two licenses at the model level. Both licenses contains the model **cc:Permission**. Even if the two elements are not the same, there is a subsumption relation between them (Table 1). The first license does not contain the model **cc:Prohibition** and the

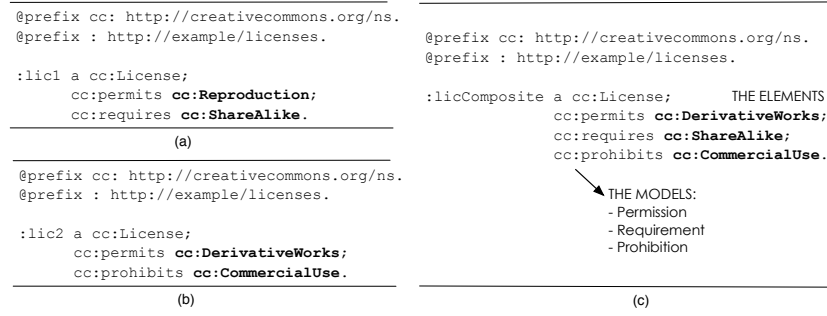


Fig. 2: Example of compatibility evaluation and composition of two licenses.

second license does not contain the model `cc:Requirement`. However, the two elements `cc:ShareAlike` and `cc:CommercialUse` are compatible (Table 2.c). Thus, the two licenses are compatible. \mathcal{L}_c is obtained using the more permissive permission, and the OR-composition heuristic.

3 Future challenges

The first point to be addressed is a legal and social validation of the proposed framework. This means to evaluate, not only quantitatively the performance of the algorithms to retrieve remote licensing statements referenced in the data, but also the legal value of the composite license. We will address an evaluation of the disparate named graphs that might be used in a “typical” query, and their relative proportions that have compatible or incompatible licenses. Finally, we are defining more complex heuristics like the one looking for the minimal composite license.

References

1. H. Abelson, B. Adida, M. Linksvayer, and N. Yergler. ccREL: The creative commons rights expression language. Technical report, 2008.
2. G. R. Gangadharan, M. Weiss, V. D’Andrea, and R. Iannella. Service license composition and compatibility analysis. In *ICSOC, LNCS 4749*, pages 257–269, 2007.
3. T. Heath and C. Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Morgan & Claypool, 2011.
4. M. Krötzsch and S. Speiser. Sharealike your data: Self-referential usage policies for the semantic web. In *ISWC, LNCS 7031*, pages 354–369, 2011.
5. P. Miller, R. Styles, and T. Heath. Open data commons, a license for open data. In *LDOW*, 2008.
6. H. L. Truong, G. R. Gangadharan, M. Comerio, S. Dustdar, and F. De Paoli. On analyzing and developing data contracts in cloud-based data marketplaces. In *AP-SCC, IEEE*, pages 174–181, 2011.
7. S. Villata and F. Gandon. Licenses compatibility and composition in the Web of Data. In *COLD*, 2012.