Proceedings of the 1st Workshop on Question Answering Over Linked Data (QALD-1)

May 30, 2011, Heraklion, Greece

Co-located with the 8th Extended Semantic Web Conference

Preface

While more and more semantic data is published on the Web, the question of how typical Web users can access this body of knowledge becomes of crucial importance. Keyword queries, as common with Web search nowadays, constitute a highly ambiguous and very impoverished representation of an information need. Thus, keyword-based search interfaces cannot fully exploit the expressive power of Semantic Web datamodels and query languages such as SPARQL. So far there are not many paradigms which would allow end users to profit from the expressive power of these standards while at the same time hiding the complexity behind an intuitive and easy-to-use interface. Important and promising research directed towards this goal is offered by search paradigms based on natural language question answering, which allow users to express arbitrarily complex information needs in an intuitive fashion. Studies like [5] have shown that when searching for information, users indeed prefer natural language interfaces to search interfaces based on keyword queries. The main challenge consists of translating these information needs into a form such that they can be evaluated using standard Semantic Web query processing and inferencing techniques.

In recent years, there have been important advances in semantic search and question answering over RDF data (for instance [8], [9] and [10], as well as [11] and [7]). In parallel to these developments in the Semantic Web community, there has been substantial progress on question answering from textual data [12] as well as in the area of natural language interfaces to databases [1]. An important challenge for the Semantic Web, but also Natural Language Processing communities, is scaling question answering approaches to Linked Data (for an overview on this topic see [13]). The main challenges involved herein are related to dealing with a heterogeneous, distributed and huge set of interlinked data. The availability of such an amount of open and structured data has no precedents in computer science. While successful natural language based search interfaces to structured knowledge exist, they are typically based on data that are by and large manually coded and homogeneous (e.g. Wolfram Alpha[1] and Powerset[2]) or propriertary (e.g. TrueKnowledge[3]). New approaches that can deal with the open, distributed, heterogeneous character of Linked Data are urgently needed. The 1st workshop on Question Answering over Linked Data (QALD-1) was aimed at exploring such approaches. To this end, we issued a call for papers on the following topics:

– Question analysis
– Natural language processing approaches applied to semantic search engines
– Disambiguation and inferencing across multiple sources and domains

[1] http://www.wolframalpha.com
[2] http://www.powerset.com
[3] http://www.trueknowledge.com

- Distributed evaluation of queries over Linked Data
- Lexical resources supporting question answering over Linked Data
- Discovery on the fly of relevant Linked Data sources
- Support of multiple languages
- Methods for scaling up question answering to Linked Data
- Efficiency and performance aspects
- Dealing with data and schema heterogeneity
- Answer merging and ranking
- Providing justifications of answers and conveying trust
- User feedback and interaction
- Habitability and usability aspects
- Evaluation of question answering approaches for Linked Data

We received six submissions. To each of them three reviewers were assigned. On the basis of their reviews, three papers were accepted for presentation, yielding an acceptance rate of 50%.

Additionally, the workshop was accompanied by an open challenge in order to facilitate the comparison between different approaches and systems, as the Semantic Web community has not yet adopted standard evaluation benchmarks for semantic question answering systems that focus on the ability to solve open-ended real life problems over real-world datasets. In order to develop the datasets needed to formally judge the quality of question answering approaches at a large scale, two independent datasets have been provided as part of this challenge: an RDF export of MusicBrainz and DBpedia 3.6 – together with 50 training questions and 50 test questions for each dataset, designed to represent questions of different complexity that real end users would ask, annotated with SPARQL queries and corresponding answers. The datasets as well as all training and test questions can be accessed from the workshop website:

`http://www.sc.cit-ec.uni-bielefeld.de/qald-1`

The main goal of the challenge was to get a picture of the strengths, capabilities and current shortcomings of question answering systems, as well as to gain insight into how question answering approaches can deal with the fact that the amount of RDF data available on the Web is huge and that this data is distributed as well as heterogeneous with respect to the vocabularies and schemas used. Three question answering systems participated in the test phase of the challenge: FREyA [2], covering both datasets, PowerAqua ([3],[4]), covering DBpedia, and SWIP [14], covering MusicBrainz. The results of our online evaluation are given in Figures 1 and 2. Additionally, there are question answering systems and query approaches that did not take part in the online evaluation but did use the datasets and questions, such as C-Phrase [1] (cf. pp. 38–43) and Treo (cf. pp. 24–37).

QALD-1 is the first in a series of workshops that aim at coupling current research on question answering over Linked Data with open challenges that benchmark question answering systems and thereby evaluate their success in fulfilling real-world tasks on the Semantic Web.

	total	processed	right	wrong	recall	precision	f-measure
FREyA	50	43	27	16	0.54	0.63	0.58
PowerAqua	50	46	24	22	0.48	0.52	0.5

Fig. 1: *Results for DBpedia*

	total	processed	right	wrong	recall	precision	f-measure
FREyA	50	41	33	8	0.66	0.8	0.71
SWIP	50	35	28	7	0.8	0.56	0.66

Fig. 2: *Results for MusicBrainz*

This volume contains the proceedings of the QALD-1 workshop. After an abstract of the invited talk *Inside the mind of Watson* given by Chris Welty from the IBM T.J. Watson Research Center in New York, it includes three contributions, presenting the question answering system *FREyA* by Damljanovic, Agatonovic & Cunningham, as well as its performance on the QALD-1 datasets, the query mechanism *Treo* by Freitas, Oliveira, Curry, O'Riain & Pereira da Silva, with an evaluation on the QALD-1 training data, and a natural language interface to the MusicBrainz dataset using the C-Phrase authoring tool by Granberg & Minock. They give a detailed impression of the challenges involved in scaling existing question answering approaches to Linked Data.

May 2011

Christina Unger
Philipp Cimiano
Vanessa Lopez
Enrico Motta
Organizing Committee QALD-1

References

1. Minock, M.: C-Phrase: A system for building robust natural language interfaces to databases. Data Knowl. Eng. 69:3, 290–302 (2010)
2. Damljanovic, D., Agatonovic, M., Cunningham, H.: Natural Language Interfaces to Ontologies: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction. In: Proceedings of the 7th Extended Semantic Web Conference, Springer Verlag (2010)
3. Lopez, V., Motta, E., Uren, V.: PowerAqua: Fishing the Semantic Web. In: Proceedings of the 3rd European Semantic Web Conference (2006)
4. Lopez, V., Sabou, M., Uren, V., Motta, E.: Cross-Ontology Question Answering on the Semantic Web: an initial evaluation. In: Proceedings of the Knowledge Capture Conference (2009)
5. Kaufmann, E., Bernstein, A.: How Useful Are Natural Language Interfaces to the Semantic Web for Casual End-Users? In: Procooedings of the Joint International and Asian Semantic Web Conference, pp. 281–294 (2007)
6. Kaljurand, K.: ACE View - An Ontology and Rule Editor based on Controlled English. In: Proceedings of the International Semantic Web Conference (2008)
7. Cimiano, P., Haase, P., Heizmann, J., Mantel, M., Studer, R.: Towards portable natural language interfaces to knowledge bases - The case of the ORAKEL system. Data Knowl. Eng. 65:2, 325–354 (2008)
8. Bernstein, A., Kaufmann, E., Göhring, A., Kiefer, C.: Querying Ontologies: A Controlled English Interface for End-Users. In: Proceedings of the International Semantic Web Conference, pp. 112–126 (2005)
9. Lopez, V., Nikolov, A., Sabou, M., Uren, V., Motta, E., D'Aquin, M.: Scaling up Question-Answering to Linked Data. In: Proceedings of the 17th International Conference on Knowledge Management and Knowledge Engineering (2010)
10. Lopez, V., Uren, V., Sabou, M., Motta, E.: Cross ontology query answering on the semantic web: an initial evaluation. In: Proceedings of the International Conference on Knowledge Capture, pp. 17–24 (2009)
11. Tran, T., Wang, H., Rudolph, S., Cimiano, P.: Top-k Exploration of Query Candidates for Efficient Keyword Search on Graph-Shaped (RDF) Data. In: Proceedings of the International Conference on Data Engineering, pp. 405–416 (2009)
12. Strzalkowsi, T., Harabagiu, S.: Advances in Open Domain Question Answering. Springer (2006)
13. Lopez, V., Uren, V., Sabou, M, Motta, E.: Is question answering fit for the semantic web?: a survey. Accepted for publication in: Semantic Web Journal.
14. Comparot, C., Haemmerlé, O., Hernandez, N.: An easy way of expressing conceptual graph queries from keywords and query patterns. In: Croitoru, M., Ferré, S., Lukose, D. (eds.) ICCS. LNCS, vol. 6208, pp. 84–96, Springer (2010)

Organization

QALD-1 is organized by the Semantic Computing Group, CITEC, Bielefeld University and the Knowledge Media Institute (KMi), Open University. QALD-1 is endorsed by FlareNet.

Organizing Committee

Christina Unger, CITEC, Bielefeld University, cunger@cit-ec.uni-bielefeld.de
Philipp Cimiano, CITEC, Bielefeld University, cimiano@cit-ec.uni-bielefeld.de
Vanessa Lopez, KMi, Open University, v.lopez@open.ac.uk
Enrico Motta, KMi, Open University, e.motta@open.ac.uk

Referees

Sören Auer
Avi Bernstein
Chris Bizer
Kalina Bontcheva
Johan Bos
Tim Finin
Jorge Gracia
Gregory Grefenstette
Iryna Gurevych
Andreas Harth
Tom Heath
Bernardo Magnini
Michael Minock
Dan Moldovan
Günter Neumann
David Palfrey
Maarten de Rijke
Sebastian Rudolph
Victoria Uren
Duc Thanh Tran
Haofen Wang
Gerhard Weikum

Table of Contents

Inside the mind of Watson (Invited talk)

Chris Welty

IBM T.J. Watson Research Center, New York

Abstract

Watson is a computer system capable of answering rich natural language questions and estimating its confidence in those answers at a level of the best humans at the task. On Feb 14–16, in an televised event, Watson triumphed over the best human players of all time on the American quiz show, Jeopardy!. In this talk I discuss how Watson works and dive into some of its answers (right and wrong), with a focus on the various uses of linked data as evidence.

About Chris Welty

Chris Welty is a Research Scientist at the IBM T.J. Watson Research Center in New York. Previously, he taught Computer Science at Vassar College,. taught at and received his Ph.D. from Rensselaer Polytechnice Institute, and accumulated over 14 years of teaching experience before moving to industrial research. Chris' principal area of research is Knowledge Representation, specifically ontologies and the semantic web, and he spends most of his time applying this technology to Natural Language Question Answering as a member of the DeepQA/Watson team and, in the past, Software Engineering. Dr. Welty is a co-chair of the W3C Rules Interchange Format Working Group (RIF), serves on the steering committee of the Formal Ontology in Information Systems Conferences, is president of KR.ORG, on the editorial boards of AI Magazine, The Journal of Applied Ontology, and The Journal of Web Semantics, and was an editor in the W3C Web Ontology Working Group. While on sabbatical in 2000, he co-developed the OntoClean methodology with Nicola Guarino. Chris Welty's work on ontologies and ontology methodology has appeared in CACM, and numerous other publications.

FREyA: an Interactive Way of Querying Linked Data using Natural Language

Danica Damljanovic[1], Milan Agatonovic[2], and Hamish Cunningham[1]

[1] Department of Computer Science, University of Sheffield
Regent Court, 211 Portobello Street, Sheffield, UK
{d.damljanovic|h.cunningham}@dcs.shef.ac.uk
[2] Fizzback, London, United Kingdom
magatonovic@fizzback.com

Abstract. Effective exploitation of the Linked Open Data is a big challenge largely because of the complexity and syntactic unfamiliarity of the underlying triple models and the query languages built on top of them. Natural Language Interfaces are increasingly relevant for information systems fronting rich structured data stores such as RDF and OWL repositories, mainly because of the conception of them being intuitive for human. In the previous work, we developed FREyA, an interactive Natural Language Interface for querying ontologies. It uses syntactic parsing in combination with the ontology-based lookup in order to interpret the question, and involves the user if necessary. The user's choices are used for training the system in order to improve its performance over time. In this paper, we discuss the suitability for using FREyA to query the Linked Open Data. We report on the performance in terms of precision and recall using the MusicBrainz and DBpedia datasets.

Keywords: natural language interfaces, ontologies, question-answering, learning, clarification dialogs

1 Introduction

With the rapid growth of the Linked Open Data (LOD) cloud[3] the effective exploitation becomes an issue largely because of the complexity and syntactic unfamiliarity of the underlying triple models and the query languages built on top of them. Natural Language Interface (NLI) systems become increasingly relevant for information systems fronting rich structured data stores such as RDF and OWL repositories, mainly because of the conception of them being intuitive for human.

According to [12], a major challenge when building NLIs is to provide the information the system needs to bridge the gap between the way *the user* thinks about the domain of discourse and the way *the domain knowledge is structured* for computer processing. This implies that in the context of NLIs to ontologies, it is very important to consider the ontology structure and content. Two

[3] http://linkeddata.org

ontologies describing identical domains (e.g., music) can use different modelling conventions. For example, while one ontology can use a datatype property *artist-Name* of class *Artist*, the other one might use instances of a special class to model the artist's name[4]. A *portable* NLI system would have to support both types of conventions without sacrificing performance. *Portable* or *transportable* NLIs are those that can be adapted easily to new domains (or new ontologies covering the same domains). Although they are considered as potentially much more useful than domain-specific systems, constructing transportable systems poses a number of technical and theoretical problems because many of the techniques developed for specialised systems preclude automatic adaptation to new domains [12]. Moreover, it is noted that portability affects retrieval performance: "the more a system is tailored to a domain, the better its retrieval performance is" [8, p.281].

Ontologies can be constructed to include sufficient lexical information to support a domain-independent query analysis engine. However, due to different processes used to generate ontologies, the lexicon might be of varying quality. In addition, some words might have different meanings in two different domains or context. For example, *How big* might refer to *height*, but also to *length*, *area*, or *population* – depending on the question context, but also on the ontology structure. This kind of adjustments – or mappings from words or phrases to ontology concepts/relations, is usually performed during the customisation of NLIs.

Many NLIs for querying ontologies have been developed in recent years. Challenges related to the Natural Language understanding such as *ambiguity* and *expressiveness* are balanced by constraining the supported language, e.g. by using a Controlled Natural Language, such as in AquaLog [2], or ORAKEL [6]. While NLI systems with a good performance require customisation (such as in the case of ORAKEL), several systems have been developed for which the customisation is not mandatory (e.g., AquaLog, PANTO [14], Querix [15], NLP-Reduce [15], QuestIO [17]). However, as reported in [2] the customisation usually improves recall. On the other hand, the complexity of supported questions differs from one system to another. While systems such as NLP-Reduce or QuestIO process queries without deep grammar analysis, the other systems (such as ORAKEL) support compositional semantic constructions such as quantification and negation.

With regards to *portability*, most of these systems are trialled with ontologies which cover different, but narrow domains, with the exception of PowerAqua [13], the system that evolved from AquaLog aiming to serve as a Question-Answering system for the Semantic Web. PowerAqua was evaluated in the open-domain scenario [11] (e.g. through querying the ontologies indexed by Watson [19]). Portability of the majority of other NLIs to ontologies is tested by demonstrating that all that is required to port the system is the ontology URI – the system

[4] See for example how class *Alias* is used in the Proton System Module ontology: `http://proton.semanticweb.org/`

automatically generates the *domain lexicon* by reading and processing ontology lexicalisations.

With the availability of Linked Open Data, *portability* gained a new dimension bringing up the open-domain scenario where the context is multiple domains/ontologies on the contrary to the previously considered closed-domain, where the context is usually one or a limited set of domains/ontologies. Having more than one ontology describing exactly the same domain, or hundreds of domains in one huge dataset requires support for *heterogeneity, redundancy* and *incompleteness* which comes with this multi-billion dataset. In other words, the system now needs not only to deal with how to map certain terms to the ontology concepts (such as mapping WH-phrases e.g. *Where* to an ontology concept e.g. *Location*) but it also needs to disambiguate and decide which ontology should provide the best answer (should *Where* be mapped to *http://purl.org/dc/terms/Location*, *http://dbpedia.org/ontology/locationCity* or any other). On the other hand, availability of such enormous knowledge base gives the possibility that the experience which has been collected for decades by researching open-domain Question-Answering systems, NLIs to databases, and dialog systems can now be merged in order to successfully accomplish what has been a great challenge for such a long time: answering questions automatically using the distributed sources available on the Web. This has not been possible with databases as they are distributed over the Web and not interoperable, while Question-Answering systems have to process large amounts of unstructured text and use methods from Information Retrieval to locate documents in which the answer may appear. Information Retrieval methods although scale well, do not often capture enough semantics - relevant documents could be easily disregarded if the answer was hidden in a form which is not in-line with the expected patterns.

In this paper, we discuss requirements and suitability for querying the Linked Open Data using the system called FREyA. FREyA is named after **F**eedback, **R**efinement and **E**xtended Vocabular**Y** **A**ggregation [5], as it aims to investigate whether *user interaction* coupled with deeper syntactic analysis and usability methods such as feedback and clarification dialogs can be used in combination to improve the performance of NLIs to ontologies. FREyA has previously shown a good performance (recall and precision reaching 94.4% [5]) on the Mooney GeoQuery dataset which has been extensively used for the evaluation of NLIs in recent years. We report on the performance of FREyA using the MusicBrainz and DBpedia datasets, which are provided within the QALD-1 challenge[5] and discuss how we begin to address the problem of querying the linked data in the open-domain scenario.

2 FREyA

FREyA is an interactive Natural Language Interface for querying ontologies which combines usability enhancement methods such as feedback and clarifi-

[5] `http://www.sc.cit-ec.uni-bielefeld.de/qald-1`

cation dialogs in an attempt to: 1) **improve recall** by generating the dialog and enriching the domain lexicon from the user's vocabulary, whenever an "unknown" term appears in a question 2) **improve precision** by resolving *ambiguities* more effectively through the dialog. The suggestions shown to the user are found through ontology reasoning and are initially ranked using the combination of string similarity and synonym detection (using WordNet[1]). The system then learns from the user's selections, and improves its performance over time. In what follows we give a brief overview of FREyA, followed by the requirements for using the system with different datasets and challenges raised by using it with the linked data. More details about FREyA and the underlying algorithms can be found in [5], [4] and [3].

2.1 FREyA workflow

Figure 1 shows the workflow starting with a Natural Language (NL) question (or its fragment), and ending when the answer is found. **Syntactic parsing**

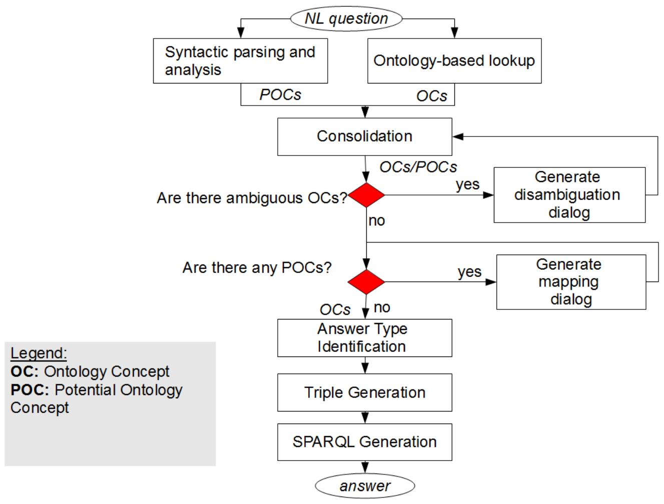

Fig. 1: FREyA Workflow

and analysis generates a parse tree using Stanford Parser [16] and then uses several heuristic rules in order to identify *Potential Ontology Concepts (POCs)*. POCs refer to question terms/phrases which can but not necessarily have to be linked to Ontology Concepts (OCs). POCs are chosen based on the analysis of the syntactic parse tree, however this analysis does not require strict adherence to syntax and works on ill-formed questions and question fragments as well as on the grammatically correct ones. For example, nouns, verbs, or WH-phrases such as *Where*, *Who*, *When*, *How many* are expected to be found by our **POC Identification algorithm**. This algorithm is based on the identification of prepreterminals and preterminals in the parsed tree, and also on their part-of-speech tags. It is possible to configure the system to ignore or consider specific part-of-speech tags.

Ontology-based Lookup links question terms to logical forms in the ontology which we call Ontology Concepts (OCs) without considering any context or grammar used in the question. Ontology Concepts refer to *instances/individuals*, *classes*, *properties*, or *datatype property values* such as string literals. By default, the system assumes that *rdfs:label* property is used to name the specific Ontology Concept. However, for ontologies which use different naming conventions (such as using *dc:title* inside the MusicBrainz dataset), it is possible to predefine which properties are used for names. This will enable the system to make the distinction between making a *datatype property value element* and an *instance element*. This distinction is important as different *elements* are used differently during the *Triple Generation* and *SPARQL generation* steps. To give an example using the MusicBrainz training dataset, for the query *When did Kurt Cobain die? Kurt Cobain* would be linked to four Ontology Concepts (all individuals), one referring to *mm:Artist*, one referring to *mm:Album*, and the other two referring to *mm:Track*[6], because it is matched with the string literal of the property *dc:title* of these four URIs which is *Kurt Cobain*. However, if the question were *Did Kurt Cobain die on April 8, 1994?*, the *April 8, 1994* would be annotated as a *datatype property value element* related to the individual *Kurt Cobain*, as it is matched with the value of the property *mm:endDate*.

Consolidation algorithm aims at mapping existing POCs to OCs. While this algorithm attempts to perform this step automatically, it is possible that it requires attention from the user. This is the case when there are ambiguous OCs in the question which could not be resolved automatically, or when a POC could not be mapped to an OC automatically. If we look at the query *Give me all former members of the Berliner Philharmoniker.*, the **POC Identification Algorithm** will find that *the Berliner Philharmoniker* is a POC, while the **Ontology-based Lookup** will find that *Berliner Philharmoniker* is an OC, referring to an instance of *mm:Artist*. As the only difference in the POC and the OC text is a determiner (*the*), the consolidation algorithm will resolve this POC by removing it, and by verifying that this noun phrase refers to the OC with *dc:title Berliner Philharmoniker*.

In cases when the system fails to automatically generates the answer (or when it is configured to work in the *forceDialog* mode, see Section 2.2) it will prompt the user with a dialog. There are two kinds of dialogs in FREyA. The **disambiguation dialog** involves the user to resolve identified ambiguities in the question. The **Mapping dialog** involves the user to map the POC in question to the one of the suggested OCs. While the disambiguation dialogs always appear before the mapping dialogs, the sequence of disambiguation and mapping dialogs themselves is very important for correctly interpreting the question. This sequence is controlled differently for these two kinds of dialogs:

[6] For clarity of presentation, we use prefix *mm:* instead of `http://musicbrainz.org/mm/mm-2.1#` and *dc:* instead of `http://purl.org/dc/elements/1.1/title` in the examples.

- *The disambiguation dialogs* are driven by the question *focus* or the *answer type*, whichever is available first: the closer the OC to be disambiguated to the question focus/answer type, the higher the chance that it will be disambiguated before any other. The question focus is the term/phrase which identifies *what the question is about*, while the answer type identifies the type of the question such as *Person* in the query *Who owns the biggest department store in England?*. The focus of this question would be *the biggest department store* (details of the algorithm for identifying the focus and the answer type are described in [4]). After all ambiguities are resolved the FREyA workflow continues to resolve all POCs through the mapping dialogs.
- *The mapping dialogs* are driven by the availability of the OCs in the neighbourhood. We calculate the distance between each POC and the nearest OC inside the parsed tree, and the one with the minimum distance is the one to be used for the dialog before any other.

While the two types of dialogs are similar and indeed look identical from the user's point of view, there are differences which we will highlight here. Firstly, although we analyse the grammar in the parsed tree, we give a higher priority to the disambiguation dialog in comparison to the mapping dialog. This is because our assumption is that the question terms which exist in the graph (OCs) should be interpreted before those which do not (POCs). Note that FREyA does not attempt to interpret the whole question at once, but it does it for one pair of OCs at the time. In other words, one dialog can be seen as a pair of two OCs: an OC to which the question term is mapped, and the neighbouring OC. Secondly, the way the suggestions are generated for the two types of dialogs differ. The disambiguation dialog includes only the suggestions with Ontology Concepts that are the result of the ontology-based lookup (unless it is extended using the *forceDialog* mode, see Section 2.2). The mapping dialog, in contrast, shows the suggestions that are found through the ontology reasoning by looking at the closest Ontology Concepts to the POC (the distance is calculated by walking through the parsed tree). For the closest OC X, we identify its neighbouring concepts which are shown to the user as suggestions. Neighbouring concepts include the defined properties for X, and also its neighbouring classes. Neighbouring classes of class X are those that are defined to be 1) the domain of the property P where range(P)=X, and 2) the range of the property P where domain(P)=X.

After all OCs are disambiguated and no POCs remain to be resolved, the system proceeds to finding the answer. First, it identifies the *answer type*, and then combines OCs into triples, which are then used to generate the SPARQL query. Unlike other approaches which start by identifying the question type followed by the identification of the answer type, our approach tries to interpret the majority of the question before it identifies the answer type. The reason for this is that in FREyA there is no strict adherence to syntax, and the approach heavily relies on the ontology-based lookup and the definitions in the RDF structure. Hence, it can only identify the answer type after all relevant mappings and disambiguations are performed. Note however, that there are cases when the answer type is identified before the whole question is interpreted, and in this case it is used to drive the remaining mappings, if any (as described above).

An important part of FREyA is its **learning mechanism**. Our goal is to *learn the ranking* of the suggestions shown to the user so that after sufficient training the system can automatically generate the answer by selecting the best ranked options. In order to make the model as generic as possible, we do not update our learning model per question, but per combination of a POC/OC and the neighbouring OC. We also preserve a function over the selected suggestion such as minimum, maximum, or sum (applicable to datatype property values). This way we may extract several learning rules from one single question, so that if the same combination of POC/OC and the closest OC appears in another question, we can reuse it. The algorithm has previously shown a good performance on the Mooney GeoQuery dataset improving the initial suggestion rankings by 6% on a random sample of 103 questions (see [5]).

2.2 Querying Linked Data with FREyA

FREyA is a portable NLI in a sense that it can be easily ported to work with a different ontology, or a set of ontologies which are available either from the Web or on the local file system. It can either preload the ontologies into its own repository which is based on OWLIM[7], or connect to the already existing repository, which can be local or remote.

In order to perform the ontology-based lookup at the query processing time, FREyA requires extracting the ontology lexicalisations, processing them, and adding them to an index. The extraction of ontology lexicalisations requires reading the whole repository through the set of SPARQL queries. The number of SPARQL queries depends on the size of the schema which describes the dataset. Nowadays, the data are distributed over various types of servers, which often allow access through SPARQL endpoints. However, depending on the repository which is used underneath, some SPARQL queries can be highly unoptimised and slow. Alternative solution is to use services such as Watson [19] or Sindice ([7], [18]) which index ontologies on the Web, in order to remove the burden of the system initialisation which can be large for large datasets. However, the downside of this approach is the lack of control over these services. As pointed out in [11], the resources on the open Web that can be accessed through Watson seem to have quality issues: many ontologies are not populated, and there are many redundant, noisy and incomplete data (for example, the schema could be missing). These problems are partially addressed by approaches for assessing the quality through tracking the provenance (e.g.,[10]), however, much more needs to be done in the years to come in order to use and query the Web of Data effectively.

Most of the recently developed NLIs to ontologies (including our own QuestIO [17]) are built with the assumption that *ontologies are perfect*, and that:

[7] `http://ontotext.com/owlim`

– each concept/relation in the ontology has the human lexicalisation which describes it – not necessarily a definition, but rather a term which a human would use to refer to this concept/relation;
– each concept/relation is positioned carefully in the taxonomy: all super-concepts/relations are more generic, and all sub-concepts/relations are more specific.

Therefore, the performance of these systems is directly dependent on the quality of data available in the ontologies. At about the time of the development of these systems, the initiative has started to encourage people to publish their own data through the Linked Open Data Project, generate their own ontologies from structured data, and soon after, the large amount of RDF graphs have been made available and interlinked with each other. None of these were perfect – lexicalisations do exist, but not often they reflect "a term which a human would use to refer to this concept" – this is especially the case for properties. In addition, the flat structure is often dominant. One of the reasons for this is scalability: tractable reasoners do not scale well if the structure of the ontology is complex. The assumptions based on which NLIs to ontologies have been developed had to adapt to the new challenges which have appeared, the main one being that *ontologies are not perfect*, and that tools which work with them must take this into consideration. In addition, the *scale* becomes an issue, and also *incompleteness, heterogeneity, and noise* inherent in these data. A huge number of ontologies interlinked with each other means a high probability that there is a redundant information, which needs to be filtered out by the systems used for querying these data.

FREyA does not require a strict adherence to syntax, however, it relies on the ontology-based lookup. Trying a sample query *What is the capital of France?* with FREyA initialised with a superset of DBpedia (accessed through `http://www.factforge.net/sparql` repository) revealed that according to the extracted lexicon, **each word in the question refers to at least one Ontology Concept**. If there were no automatic disambiguation nor heavy grammar analysis, the system would model the first dialog asking: *What is 'what'? Is 'what' related to: LIST OF URIs.* A similar dialog would be modelled for 'is': the system would ask the user whether *is* is related to: *be*, *was*, or *were*. And so on, for each word in the question.

These situations must be resolved either by performing automatic disambiguation (which might be expensive for datasets with billions of triples) or by constraining the supported language and allowing the user to type in only a limited set of question types. In case of the system failing to automatically interpret the question, it can prompt the user with the dialog as is the case with FREyA. The fine balance is in the combination of these approaches: disambiguate as much as possible and use the ranking mechanisms (e.g., those that exist in FREyA, or any other methods for effective ranking such as [9]), and correct them if necessary using the interactive features of FREyA.

When trying any dataset with FREyA for the first time, it is advisable to use the dialog as much as possible in order to check the system interpretations and correct them if necessary. In that regard, there are several *modes* that can be used:

- **Automatic mode** The dialog in FREyA is designed in a way that it can be configured based on the level of confidence. The maximum confidence level allows using FREyA in the *automatic mode* meaning that the system will generate the answer by simulating selection of the best ranked options. This mode is used when the confidence is high that the ranking is effective, or the system has been trained enough and can make the decisions correctly.
- **ForceDialog mode** generates the dialog for each attempt to map a question term to an OC.

3 Evaluation

In this section we report the performance of FREyA using the MusicBrainz and DBpedia datasets provided within the QALD-1 challenge. We preloaded the data into our local repository (BigOWLIM 3.4, on the top of Sesame[8]) and then initialised the system using the SPARQL queries. Another option was to connect to the SPARQL endpoint provided by the QALD-1 challenge organisers[9], however, this was a difficult path due to the limited server timeout, which was not sufficient for executing some of the required queries.

Generating the index which is required for performing the ontology-based lookup is a mandatory step but is done once per dataset, although it might be time-consuming depending on the size of the data. Table 1 shows the number of statements loaded into OWLIM repository, the number of SPARQL queries executed and the time it took to finish the whole process per dataset.

	MusicBrainz	**DBpedia**
#explicit statements	14 926 841	328 318 709
#statements	19 202 664	372 110 845
#entities	5 490 237	96 515 478
#SPARQL queries executed	30	361623
initialisation time	1380s (0.38h)	182779s (50.77h)

Table 1: Initialisation of the system and the size of datasets

After the index is generated, it is used at the query execution time. We first ran the 50 training queries for both datasets and measured the overall precision, recall and f-measure. We then repeat the process with 50 test questions for each dataset. This experiment was conducted with FREyA in the *forceDialog* mode. Results are shown in Table 2.

MusicBrainz was a challenging dataset due to the existence of properties such as *beginDate* and *endDate*, which do not have any domain defined, and moreover, which are used extensively throughout the ontology and especially in the

[8] `http://openrdf.org`
[9] `http://greententacle.techfak.uni-bielefeld.de:5171/sparql`

	MusicBrainz		**DBpedia**	
	Training	*Testing*	*Training*	*Testing*
Precision	0.75/0.77	0.66/0.8	0.74/0.85	0.49/0.63
Recall	0.66/0.68	0.54/0.66	0.58/0.66	0.42/0.54
F-measure	0.70/0.74	0.59/0.71	0.67/0.72	0.45/0.58
# questions not supported	6	9	11	7
# reformulated questions	1	6	4	6
avg.#dialogs per question	3.4	3.65	2.7	2.85
# partially correct questions	1	1	3	12

Table 2: Performance of FREyA using QALD-1 datasets: the left figures exclude while the right figures include the questions correctly answered after reformulation. The number of dialogs per question includes only the questions that could *correctly* be answered with or without reformulation. *Not supported* questions include those that could not be correctly mapped to the correct SPARQL query due to the limited language coverage. For example, questions requiring temporal reasoning such as *Which bands were founded in 2010?* or comparative such as in *Which locations have more than two caves?*. Partially correct questions are those that have returned a portion or a superset of the correct results.

combination with the blank nodes. Several failures were due to the misfunction of the *Triple Generator* when these two properties were mapped to the wrong entity. For example, *Since when is Tom Araya a member of Slayer?* resulted in generating the following triples:

```
?joker1 - beginDate - Tom Araya (Artist)
Tom Araya (Artist) - member of band - ?joker2
?joker2 - toArtist - Slayer (Artist)
```

and the corresponding SPARQL resulted in retrieving the birthday of Tom Araya, and not the date when he joined the group which would lead to the correct answer.

Other challenges related to the ontology design in MusicBrainz include existence of the property *trackList* which had a container of type *rdf:Seq* as a range. In addition, the statements with *releaseType* property use subclasses of class *Type* and not instances of that class which caused several failures. For example, the question *Who is the creator of the audiobook the Hobbit?* requires retrieving instances with lexicalisation *the Hobbit*, which are at the same type related to the class *TypeAudiobook* using the *releaseType* property, while FREyA expects that the Hobbit instances are related to the *TypeAudiobook* using relation rdf:type.

The main challenge with DBpedia was deciding which property to use to link a question term into an Ontology Concept, due to the large number of suggestions that have always been present. For example, *Who created English Wikipedia?* could be mapped to *?joker dbp:created dbpedia:English_Wikipedia* while the correct answer is returned only after using *dbo:author* relation, instead of *dbp:created*[10]. In addition, there are many quality issues such as in the question *Who designed the Brooklyn Bridge?* where *designed* was mapped

[10] We use *dbp for* `http://dbpedia.org/property` *and* dbo *for* `http://dbpedia.org/ontology` *namespaces.*

to *dbp:architect* instead of *dbp:designer* which resulted in retrieving `http://dbpedia.org/resource/John_Augustus_Roebling`, while using *dbp:designer* the result is `http://dbpedia.org/page/John_A._Roebling`. However, as no mapping exist between the two URIs, the former URI is not the same as the latter, and hence this is marked as an incorrect answer. Interestingly, the former URL is redirected to the latter, which indicates that the two URIs should also be connected using the property *sameAs* in the dataset.

Another challenge specific to DBpedia was the lack of the domain and range classes for properties. Therefore, some questions could not be correctly mapped to the underlying Ontology Concepts. In some cases, the reformulation of queries could help (such as using *spouse* instead of *married to*). However, reformulation was not always sufficient. For example, in *Which states border Utah?*, *border* needs to be mapped to the eight properties: *dbp:north*, *dbp:south*, *dbp:east*, *dbp:west*, *dbp:northwest*, *dbp:northeast*, *dbp:southwest*, and *dbp:southeast*. As none of these have any domain or range, they did not appear in the suggestions and hence the only way to answer the question using FREyA is to ask eight questions such as *Which states are north of Utah?*, *Which states are south of Utah*, and so on for each property. It is interesting to observe that 12 incorrectly answered questions using the DBpedia test questions were indeed partially correct. The correct mappings could only be placed if we were more familiar with the knowledge structure inherent in the dataset. This also explains the difference in the performance of FREyA using the training and the testing set of DBpedia.

Failures that were common for both datasets are related to the equal treatment of the datatype property values in FREyA. For example, the question *How many jazz compilations are there?* failed to be answered correctly due to FREyA finding all compilations that had the user defined tag 'jazz' which is case insensitive (using FILTER REGEX(str(?var), "`^jazz$`","i"). Therefore, it included also 'Jazz' which lead to the incorrect answer. On the other hand, it missed some entries when the fuzzy matching was needed such as in *Which companies are in the computer software industry?* that required finding not only companies with the property *industry* 'computer software' but also 'computer hardware, software', 'computer software and engineering', and the like. At the moment, the datatype property values in FREyA are supported by including the exact match (case insensitive) only. In future, we might extend our approach to support more sophisticated treatment of strings so that the treatment differs depending on the context.

Several reformulations for both datasets resulted in a significant increase of the precision and recall. Reformulations include adding quotes such as in *Which artists performed the song Over the Rainbow?*. As *Over* in *Over the Rainbow* was parsed as a preposition, the whole question failed to be answered. Adding quotes around this phrase resulted in the parser treating this phrase as a Noun Phrase which lead to the correctly answered question.

Learning To measure the effect of the learning mechanism in FREyA, we repeated the experiment in two iterations: we first answered 50 testing questions using an empty learning model and then using the system trained with 50 training questions. Results are shown in Table 3. The learning mechanism improved

	MusicBrainz		**DBpedia**	
	untrained system	*trained system*	*untrained system*	*trained system*
MRR	0.63	0.68	0.52	0.54

Table 3: Mean Reciprocal Rank for the testing set with and without learning

the overall ranking of suggestions for 0.05 for MusicBrainz, and only 0.02 for DBpedia. The reason is the size of the datasets and the relatively small number of the training questions. However, improvement of 0.02 is still an achievement considering that DBpedia has almost 100 million entities.

Execution time Considering queries that could be answered correctly, there are fluctuations which depend on the complexity of the question, but the biggest fluctuation is caused by the execution time of the SPARQL query. This affects our on fly mechanism for finding suggestions which requires executing a large number of SPARQL queries in order to generate the dialog. Long execution of the SPARQL queries is also visible when the final query is executed in order to retrieve the answer. For example, queries which include FILTER statements over literal strings such as FILTER (regex(?var, "`^jazz$`", "i")), which currently can take more than ten minutes to be executed[11]. The size of the dataset influences the execution time as well. For MusicBrainz, the average time per dialog was in the range from 0.073 to 11.4 seconds, or 8.5 seconds on average per question. For DBpedia, the execution time was much longer: from 5 to 232 seconds per dialog, and 36 seconds on average per question. This is quite slow, however, it can be optimized (e.g. by using the caching mechanisms for suggestions) and the optimisation work is in progress.

4 Conclusion and future work

In this paper we discussed the requirements and suitability of the Natural Language Interface – FREyA, to be used with different ontologies and for querying the Linked Open Data. The evaluation using the DBpedia and MusicBrainz testing datasets leads to the f-measure of 0.58 and 0.71 respectively which favourably compares to the other tested systems that participated in the QALD-1 challenge (PowerAqua 0.5 using DBpedia, SWIP 0.66 using MusicBrainz). More importantly, FREyA was the only system that is tested with both MusicBrainz and DBpedia datasets which demonstrates the portability. The learning mechanism improved the results for 5% and 2% for the MusicBrainz and DBpedia datasets respectively.

Acknowledgements Authors would like to thank Ivan Peikov from Ontotext who helped with the configuration of OWLIM which was necessary for performing the experiments reported in this paper. This research is partially supported by the EU-funded LarKC (FP7-215535) project.

[11] Experiments are run using the CentOS 5.2 Linux virtual machine running on a AMD Opteron 2431 2.40GHz CPU with 2 cores and 20G RAM.

References

1. Fellbaum, C.: WordNet - An Electronic Lexical Database. MIT Press (1998)
2. Lopez, V., Uren, V., Motta, E., Pasin, M.: AquaLog: An ontology-driven question answering system for organizational semantic intranets. Web Semantics 5(2), 72–105, Elsevier (2007)
3. Damljanovic, D.: Towards Portable Controlled Natural Languages for Querying Ontologies. In: Rosner, M., Fuchs, N. (eds) Pre-Proceedings of the Second Workshop on Controlled Natural Language. CEUR Workshop Proceedings, vol. 622 (2010)
4. Damljanovic, D., Agatonovic, M., Cunningham, H.: Identification of the Question Focus: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction. In: Proceedings of the 7th Language Resources and Evaluation Conference. ELRA (2010)
5. Damljanovic, D., Agatonovic, M., Cunningham, H.: Natural Language Interfaces to Ontologies: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction. In: ESWC 2010. LNCS, vol. 6088, pp. 106–120. Springer (2010)
6. Cimiano, P., Haase, P., Heizmann, J.: Porting natural language interfaces between domains: an experimental user study with the ORAKEL system. In: Proceedings of the 12th international conference on Intelligent user interfaces. ACM, New York (2007)
7. Tummarello, G., Delbru, R., Oren, E.: Sindice.com: Weaving the Open Linked Data. In: Proceedings of the 6th International Semantic Web Conference (2007)
8. Kaufmann, E., Bernstein, A.: How Useful are Natural Language Interfaces to the Semantic Web for Casual End-users? In: Proceedings of the Forth European Semantic Web Conference (2007)
9. Lopez, V., Nikolov, A., Fernandez, M., Sabou, M., Uren, V., Motta, E.: Merging and Ranking Answers in the Semantic Web: The Wisdom of Crowds. In: Gómez-Pérez, A., Yu, Y., Ding, Y. (eds) The Semantic Web. LNCS, vol. 5926, pp. 135–152. Springer (2009)
10. Hartig, O., Zhao, J.: Using Web Data Provenance for Quality Assessment. In: Proceedings of the First International Workshop on the Role of Semantic Web in Provenance Management (2009)
11. Lopez, V., Fernndez, M., Motta, E., Stieler, N.: PowerAqua: Supporting Users in Querying and Exploring the Semantic Web Content. Semantic Web Journal. IOS Press (2011)
12. Grosz, B.J., Appelt, D.E., Martin, P.A., Pereira, F.: TEAM: An experiment in the design of transportable natural-language interfaces. Artificial Intelligence 32(2), 173–243 (1987)
13. Lopez, V., Uren, V., Sabou, M., Motta, E.: Cross Ontology Query Answering on the semantic Web: an Initial Evaluation. In: Gil, Y., Fridman Noy, N. (eds) Proceedings of the 5th International Conference on Knowledge. ACM (2009)
14. Wang, C., Xiong, M., Zhou, Q., Yu, Y.: PANTO: A Portable Natural Language Interface to Ontologies. In: The Semantic Web: Research and Applications, pp. 473–487. Springer (2007)
15. Kaufmann, E., Bernstein, A., Fischer, L.: NLP-Reduce: A Naive but Domain-independent Natural Language Interface for Querying Ontologies. In: Proceedings of the European Semantic Web Conference. Springer (2007)
16. Klein, D., Manning, C.: Fast Exact Inference with a Factored Model for Natural Language Parsing. In: Becker, S., Thrun, S., Obermayer, K. (eds) Neural Information Processing Systems 2002. Advances in Neural Information Processing Systems 15, MIT Press (2003)

17. Damljanovic, D., Tablan, V., Bontcheva, K.: A Text-based Query Interface to OWL Ontologies. In: Proceedings of the 6th Language Resources and Evaluation Conference. ELRA (2008)
18. Oren, E., Delbru, R., Catasta, M., Cyganiak, R., Stenzhorn, H., Tummarello, G.: Sindice.com: A document-oriented lookup index for open linked data. International Journal of Metadata, Semantics and Ontologies 3:1 (2008)
19. D'Aquin, M., Baldassarre, C., Gridinoc, L., Angeletou, S., Sabou, M., Motta, E.: Watson: A Gateway for Next Generation Semantic Web Applications. Poster at ISWC 2007, `http://iswc2007.semanticweb.org/papers/Paper366-Watson-poster-ISWC07.pdf`

Treo: Combining Entity-Search, Spreading Activation and Semantic Relatedness for Querying Linked Data

André Freitas[1], João Gabriel Oliveira[1,2], Edward Curry[1], Seán O'Riain[1], and João Carlos Pereira da Silva[2]

[1]Digital Enterprise Research Institute (DERI)
National University of Ireland, Galway
[2]Computer Science Department
Universidade Federal do Rio de Janeiro

Abstract. This paper describes *Treo*, a natural language query mechanism for Linked Data which focuses on the provision of a precise and scalable semantic matching approach between natural language queries and distributed heterogeneous Linked Datasets. Treo's semantic matching approach combines three key elements: *entity search*, a *Wikipedia-based semantic relatedness measure* and *spreading activation search*. While entity search allows Treo to cope with queries over high volume and distributed data, the combination of entity search and spreading activation search using a Wikipedia-based semantic relatedness measure provides a flexible approach for handling the semantic match between natural language queries and Linked Data. Experimental results using the DBPedia QALD training query set showed that this combination represents a promising line of investigation, achieving a *mean reciprocal rank* of 0.489, *precision* of 0.395 and *recall* of 0.451.

Keywords: Natural Language Queries, Linked Data

1 Introduction

The problem of providing query and search capabilities for casual Linked Data consumers [1] poses new challenges for the areas of information retrieval and databases. Users querying Linked Data on the Web expect to query relationships represented in the data model behind datasets. Structured query languages such as SPARQL provide the capability of explicitly and unambiguously specifying these relationship constraints, at the cost of demanding from users an a priori understanding of the data representation. The scale and decentralized nature of the Web, however, represents a concrete barrier for this paradigm: it is not feasible for end users to become aware of all the vocabularies and possible representations of the data in order to query Linked Data. While this constraint can be manageable for developers building applications on the top of a limited number of datasets or vocabularies, it strongly limits the visibility and consequent utility of the data for casual users. At the heart of this problem lies the lack of flexibility of query mechanisms to cope with lexical and structural differences between user queries and the data representation, which ultimately creates a semantic gap between users and datasets. In addition, querying Linked Data can usually imply coping with distributed, high-volume and dynamic data, bringing additional challenges for the construction of effective query mechanisms for Linked Data.

This work focuses on the description of *Treo*, a natural language query mechanism for Linked Data which focuses on the provision of a best-effort semantic matching approach, balancing precision and flexibility, while satisfying the ability to perform queries over distributed, large and dynamic data. With these objectives in mind a query mechanism based on a combination of *entity search*, *spreading activation* and a *Wikipedia-based semantic relatedness measure* is proposed. The final query processing approach provides an opportunity to revisit cognitive inspired spreading activation models over semantic networks [16] under contemporary lenses. The recent availability of Linked Data, large Web corpora, hardware resources and a better understanding of the fundamental principles behind information retrieval can provide the necessary resources to enable practical applications over cognitive inspired architectures.

This paper is structured as follows: section 2 outlines the motivation and the dynamics of the proposed approach. Section 3 describes the *entity recognition and entity search approach*, section 4 describes the *query parsing* strategy followed by the description of the *Wikipedia-based semantic relatedness measures* (section 5) and by the description of the *semantic relatedness spreading activation* approach. Each section details how each component part is used to build the final query approach. Section 7 provides a discussion and preliminary results using the QALD DBPedia training dataset, followed by related work on section 8. Finally, section 9 provides a conclusion and future work.

2 Outline of the Query Processing Approach

The query approach described in this work focuses on the construction of a query mechanism targeted towards bridging the semantic gap between user queries and Linked Data vocabularies and datasets, with the objective of enabling natural language queries over Linked Data for casual users.

The strategy employed in the construction of the *Treo* query mechanism is to provide a best-effort approach for natural language queries over Linked Data. Having this objective in focus, an answer format and an user interaction approach that could better match a best-effort natural language scenario was developed, where a set of triple paths (i.e. a set of ranked connected triples which are subgraphs in the Linked Data Web) supporting a query answer is returned as a result set. The exposure of a partial set of the data where users could cognitively assess the suitability of the results and search iteratively is a strategy widely used in the construction of document search engines and a variation of this strategy is used in this work in the context of natural language queries over Linked Data.

The query processing starts with the determination of the *key entities* present in the natural language query. Key entities are entities which can be potentially mapped to instances or classes in the Linked Data Web. After detection, key entities are sent to the *entity search engine* which resolves *pivot entities* in the Linked Data Web. A pivot entity is a URI which represents an entry point for the spreading activation search in the Linked Data Web. After the entities present in the user natural language query are determined, the query is analyzed in the *query parsing module*. The output of this module is a *partial ordered dependency structure* (PODS), which is a reduced representation of the query targeted towards maximizing the matching probability between the structure of the terms present in the query and the *subject, predicate, object* structure of RDF.

Taking as an input the list of URIs of the pivots and the partial ordered dependency structure, the algorithm follows a *spreading activation search* where nodes in the Linked Data Web are explored using a *semantic relatedness measure* as an activation function to match the query terms present in the PODS (user query representation) to dataset terms (classes, properties and instances). Starting from the pivot node, the algorithm navigates through neighboring nodes in the Linked Data Web computing the semantic relatedness between query terms and vocabulary terms in the node exploration process. The query answer is built through the node navigation process. The algorithm returns a set of ranked triple paths determined by the navigation from the pivot entity to the final resource representing the answer, ranked by the average of the relatedness scores over each triple path. Answers are displayed to users using a list of triple paths merged in a graph after a simple post-processing phase. Figure 1 depicts the spreading activation process for the example query *'From which university did the wife of Barack Obama graduate?'* over DBPedia.

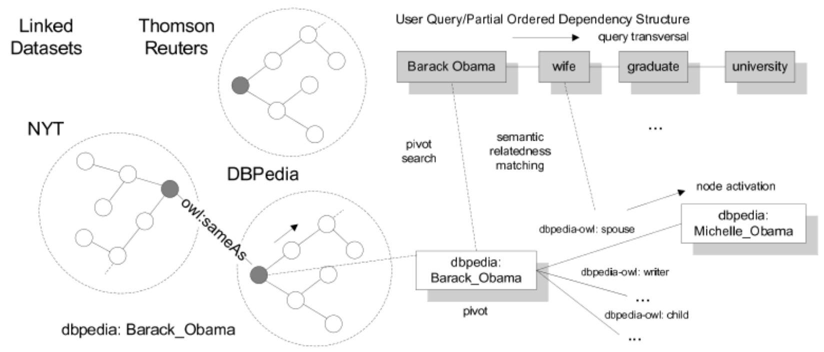

Fig. 1: The relatedness spreading activation for the question *'From which university did the wife of Barack Obama graduate?'*.

The squared gray nodes depict the PODS, a (graph pattern-like) representation of the natural language query while the squared white nodes represent nodes in the Linked Data Web. In the example, the query entity Barack Obama is mapped to a pivot entity in the Linked Data Web, which is the starting point of the spreading activation process. The following sections detail each step in the query processing approach. The query above is used as a motivational scenario to introduce each step.

3 Entity Recognition and Entity Search

The query processing starts with the determination of the set of key entities that will be used in the generation of the partial ordered dependency structure and in the determination of the final pivot entity. The process of generating a pivot candidate starts by detecting named entities in the query. The named entity recognition (NER) approach used is based on Conditional Random Fields sequence models [3] trained in the CoNLL 2003 English training dataset [17], covering people, organizations and locations. After the named entities are identified, the query is tagged by a part-of-speech (POS) tagger, which assigns grammatical tags to query terms. Rules based on the POS tags are used to determine pivot candidates which are not named entities (classes or individuals representing categories). The POS tagger used is a log-linear POS tagger based on [4]. In the case of the example query, the named entity *Barack Obama* is recognized as the main entity.

The terms corresponding to the pivot candidates are sent to an entity search engine which will resolve the terms into the final pivot URIs in the Linked Data Web. Entity-centric search engines for Linked Data are search engines where the search is targeted towards the retrieval of instances, classes and properties in the Linked Data Web, instead of approaching the problem of treating an entire RDF document as a single resource or addressing complex structured queries over Linked Data. This work uses the entity search approach proposed by Delbru et al. [5], implemented in Siren, the Semantic Information Retrieval Engine. The approach proposed by Delbru, combines query-dependent and query-independent ranking techniques to compute the final entity scores. The query-dependent score function uses a variation of the TF-IDF weighting scheme (term frequency - inverse subject frequency: TF-ISF) to evaluate entities by aggregating the values of partial scores for predicates and objects. The TF-ISF scheme gives a low weight to predicates or objects which occur in a large number of entities. This work only uses the query dependent approach.

The detected entities are sent as keywords to the entity search engine. In this process, if named entities are available they are prioritized as pivots candidates. In case the query has more than one named entity, both candidate terms are sent to the entity search engine and the entities with a larger number of properties are prioritized in the determination of the final pivot entity. In the example query, the named entity *Barack Obama* is mapped to a list of URIs representing the entity Barack Obama in different datasets (e.g. *http://dbpedia.org/resource/Barack_Obama*).

4 Query Parsing

After the key entities and pivots are determined, they are sent together with the natural language query to the query parsing module. The center of the query parsing strategy is to maximize the structural similarity between the query and the RDF-based representation of the data. The query parsing is based on the use of Stanford dependencies [2]. The fundamental notion behind dependency parsing is the idea that the syntactic structure of a sentence is determined by a set of directional bilexical relations and by the lack of phrasal nodes. Since the triple structure (subject, predicate, object) of the RDF representation minimizes the use of certain gramatical elements which are present in the free text query, the Stanford dependencies are reduced into a *partial ordered dependency structure* (PODS) by the application of a set of query transformation operations. The PODS also includes the introduction of an ordering which is defined by the relative position of the pivot and detected entities in the query. The final PODS is a directed acyclic graph connecting a subset of the original terms present in the natural language query.

After the Stanford dependencies are determined, four operations are applied to build the PODS: *merge*, *eliminate*, *join condition* and *ordering*. The *merge* operation consists in merging dependencies which are likely to form multi-words or compound expressions (e.g. Barack Obama). The *eliminate* operation remove dependencies which are less semantically significant, unlikely to be expressed in the RDF representation such as determiners, prepositions, etc. The *join condition* joins dependencies which represents conjunctive or disjunctive statements so that star-shaped graph patterns can be introduced in the final query, and finally, the *ordering* operation introduces the ordering based on the position of pivot and entities that is used in the final spreading activation algorithm. The final ordering is defined by taking the pivot entity as the root of the partial ordered dependency structure and following the dependencies until the end of the structure is reached.

For the example query the partial ordered dependency structure returned by the query parser is: Barack Obama → wife → graduate → university.

5 Semantic Relatedness

After the natural language query is parsed into the PODS and having the list of pivots determined, the spreading activation search process in the Linked Data Web starts. The center of the spreading activation search proposed in this work is the use of a semantic relatedness measure as the activation function, matching query terms to vocabulary terms. The proposed approach is highly dependent on the quality of the semantic relatedness measure. This section briefly describes the basic concepts behind semantic relatedness and the measure used in the algorithm.

Generally speaking the problem of measuring the semantic *relatedness* and *similarity* of two concepts is associated with the determination of a measure f(A,B) which expresses the semantic proximity between these concepts. While the idea of semantic *similarity* is associated with taxonomic relations between concepts, semantic *relatedness* represents more general classes of relations. Since the problem of matching natural language terms to concepts present in Linked Data vocabularies can cross both taxonomic and part-of-speech boundaries, the generic concept of semantic relatedness is more suitable to the task of semantic matching for queries over the Linked Data Web. In the example query, the relation between 'graduate' and 'University' is non-taxonomic and a purely similarity analysis would not detect appropriately the semantic proximity between these two terms. In the context of query-dataset semantic matching by spreading activation, it is necessary to use a relatedness measure that: (i) can cope with terms from different part-of-speech (e.g. verbs and nouns); (ii) measure relatedness among multi-word expressions; (iii) are based on comprehensive knowledge bases.

Existing approaches for querying Semantic Web/Linked Data knowledge bases are mostly based on WordNet similarity measures (see Related Work section). WordNet-based similarity measures [26] are highly dependent on the structure and scope of the WordNet model, not addressing the requirements above. Distributional relatedness measures [8][9][26] are able meet the previous requirements, providing approaches to build semantic relatedness measures based on large Web corpora. Recent approaches propose a better balance between the cost associated in the construction of the relatedness measure and the accuracy provided, by using the link structure present in the corpora. One example of this class of measures is the Wikipedia Link-based Measure (WLM), proposed by Milne & Witten [10], which achieved high correlation measurements with human assessments. This work will use WLM as the relatedness measure for the spreading activation process.

The WLM measure is built based on the links between Wikipedia articles. The process of creation of the WLM relatedness measure starts by computing weights for each link, where the significance of each link receives a score. This procedure is equivalent to the computation of the TF-IDF weighting scheme for the links in the place of terms: links pointing to popular target articles (receiving links from many other articles) are considered less significant from the perspective of relatedness computation. The weighting expression is defined below:

$$w(s \rightarrow t) = log\left(\frac{\mid W \mid}{\mid T \mid}\right), \text{if s} \in T, 0 \text{ otherwise} \tag{1}$$

where s and t represent the source and target articles, W is the total number of articles in Wikipedia and T is the number of articles that link to t. The relatedness measure is defined by an adaptation over the Normalized Google Distance (NGD)[10][25].

$$r(a,b) = \frac{log(max(\mid A \mid, \mid B \mid)) - log(\mid A \cap B \mid)}{log(\mid W \mid) - log(min(\mid A \mid, \mid B \mid))} \tag{2}$$

where a and b are the two terms that the relatedness is being measured, A and B are the respective articles that are linked to a and b and W is the set of all Wikipedia articles. The final relatedness measure uses a combination of the two measures. The reader is directed to [10] for additional details on the construction of the relatedness measure.

6 The Semantic Relatedness Spreading Activation Approach

Spreading activation is a search technique used in graphs based on the idea of using an activation function as a threshold for the node exploration process. Spreading activation has its origins associated with modeling the human semantic memory in cognitive psychology and have a history of applications in cognitive psychology, artificial intelligence and, more recently, on information retrieval [18]. The spreading activation theory of human semantic processing was first proposed by Quillian [20] and it was later extended by Collins & Loftus [19]. The spreading activation model introduced by Quillian [20] was proposed in the context of semantic networks, a graph of interlinked concepts which contained the basic elements formalized today under the scope RDF and RDFS. The recent emergence of the Linked Data Web is likely to motivate new investigations in spreading activation techniques.

The processing technique behind spreading activation is simple, consisting of one or more propagated pulses and a termination check. In addition, the model can implement propagation decays and constraints on the spreading activation process. The semantic relatedness spreading activation algorithm proposed in this work takes as an input a partial ordered dependency structure D(V, E) and searches for paths in the Linked Data Web graph W(V, E) maximizing the semantic relatedness between D and W taking into account the ordering of both structures, where both structures are defined by the set of vertices V and edges E. In the approach used in this work, the propagation is defined by the computation of the relatedness measure between the terms present in the query (PODS) and the dataset terms, while the termination check is given by the PODS size.

The spreading activation works as follows. After the URI of the pivot element is dereferenced (the associated RDF descriptor for the URI is fetched), the algorithm computes the semantic relatedness measure between the next term in the PODS and the *properties*, *type terms* and *instance terms* in the Linked Data Web. Type terms represent the types associated to an instance through the *rdfs:type* relation. While properties and ranges are defined in the terminological level, type terms require an instance dereferenciation to collect the associated types. Nodes above a relatedness score threshold (activation function) determine the node URIs which are explored in the search process. The activation function is given by an *adaptive discriminative relatedness threshold* which is defined based on the set of relatedness scores associated with a specific node (the adaptive threshold is defined for each explored node). The threshold selects the relatedness scores with higher discrimination and it is defined as a linear function of the standard deviation σ of the relatedness scores. The linear constant α associated with σ is determined empirically and it represents the average discrimination in terms of σ for the relatedness measure employed ($\alpha = 2.185$). The original value of α decays by an exponential factor of 0.9 until it finds a candidate node which is above the activation threshold. The final semantic relatedness spreading activation algorithm is defined below:

```
$D(V_G, E_G)$ : partial ordered dependency graph
$W(V_W, E_W)$ : LD graph
$A(V_A, E_A)$ : answer graph
$pivots$ : set of pivots URI's
for all $p$ in $pivots$ do
  $initialize(A, p)$
  while $hasUnvisitedNodes(V_A)$ do
    $v \leftarrow nextNode(V_A)$
    $dereference(v, W)$
    for all $nextT$ in $getNextDependencyNodes(D)$ do
      $best \leftarrow bestNodes(v, nextT, W)$
      $update(V_A, best)$
      $update(E_A, best)$
    end for
  end while
end for
```

For the example query *'From which university did the wife of Barack Obama graduate?'*, starting from the pivot node (*dbpedia: Barack_Obama*), the algorithm follows computing the semantic relatedness between the next query term (*'wife'*) and all the properties, associated types and instance labels linked to the node dbpedia:Barack_Obama (*dbpedia-owl:spouse*, *dbpedia-owl:writer*, *dbpedia-owl:child*, ...). Nodes above the adaptive relatedness threshold are further explored. After the matching between *wife* and *dbpedia-owl: spouse* is defined, the object pointed by the matched property (*dbpedia: Michelle_Obama*) is dereferenced, and the RDF of the resource is retrieved. The next node in the PODS is *graduate*, which is mapped to both *dbpedia-owl:University* an *dbpedia-owl:Educational_Institution* specified in the types. The algorithm then navigates to the last node of the PODS, *university*, dereferencing *dbpedia:Princeton_University* and *dbpedia: Harvard_Law_School*, matching for the second time with their type. Since the relatedness between the terms is high, the terms are matched and the algorithm stops, returning the subgraph containing the triples which maximize the relatedness between the query terms and the vocabulary terms. The proposed algorithm works as a best-effort query approach, where the semantic relatedness measure provides a semantic ranking of returned triples. The final algorithm returns a ranked list of triple paths (figure 2) which are (when possible) merged into a graph.

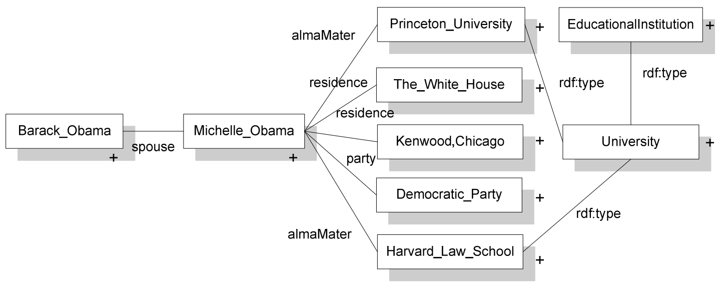

Fig. 2: Merged set of returned triple paths for the example query. The correct answer is given by the nodes Princeton University and Harvard Law School

7 Discussion & Preliminary Evaluation

This section has the objective of providing a brief analysis of the strenghts and weakenesses of the proposed approach and to describe the preliminary results achieved over the 1st QALD DBPedia training query set [7].

The proposed query approach consists of a two step process where the core entity in the query is resolved into entities in Linked datasets in the first step, followed by the structural query matching process using spreading activation combined with semantic relatedness. The process of first resolving the core entity plays an strategic role in the scalability and parallelism of the approach. The process of finding pivot entities associated with instances in the datasets, tends to be less ambiguous and it is related to the most informative part of the query (informativeness in this context is defined as the ability to constrain the search space in the Linked Data Web). Entity search demands the creation of an entity index for the datasets. As a strength, entity indexes, differently from structure indexes [22] can benefit from less challenging constraints in terms of index space, time and indexing frequency (related with dataset dynamics). In the proposed approach, entity search is used as the main mechanism for ranking pivots. One of the current limitations of using a pure entity search approach for pivot selection is the fact that it ignores the context provided by the remaining query terms. This ultimately can lead to an increase in query execution time or a decrease in precision. One possible solution for this problem is to extend the existing mechanism with dataset summaries [24], where additional query terms can be used in a summary-based search, composing the information provided in summaries with the entity search process.

The semantic relatedness spreading activation search algorithm traverses the Linked Data Web graph using the PODS query representation. Two important factors for the performance of the approach is the term ordering in the PODS and the overall performance of the semantic relatedness measure. The WLM semantic relatedness measure used in this work showed high discrimination in the node exploration process (average 2.81 σ measured in a 50% sample of the query set) and robustness in relation to the computation of the relatedness among terms from different grammatical classes for the DBPedia query set. Another strength of the proposed approach is the fact that it is highly and easily parallelizable both for spreading activation search and in the computation of semantic relatedness measure. The query mechanism currently uses sequences of URI dereferenciations as the primary way to navigate through Linked Data. From a practical perspective, the HTTP requests can bring high latencies in the node exploration process. In order to be effective, the algorithm should rely on mechanisms to reduce the number unnecessary HTTP requests associated with the dereferenciation process, unecessary URI parsing or label checking and unnecessary relatedness computations. The prototype, *Treo*, have implemented three local caches: one for RDF, one for relatedness values and the third for URI/label-term mapping.

The quality of the approach was evaluated against the 50 queries of the QALD DBPedia training query set in terms of *precision*, *recall* and *mean reciprocal rank*. Online DBPedia (version 3.6) [6] was used as the dataset and a local version of Siren, indexing a local copy of DBPedia, was used as the entity search mechanism. In the scope of the system described on this paper, an answer is a set of ranked triple paths. Different from a SPARQL query result set or from typical QA systems, the proposed algorithm is a best-effort approach where the relatedness activation function works both as a ranking and a cut-off function and the final result is a merged and collapsed set of subgraphs containing the answer triple paths. For the determination of *precision* we considered as correct answers triple paths containing the URI of the answer. For the example query used through this article, the graph in figure 2 containing the answer *Barack Obama* $\rightarrow$ *spouse* $\rightarrow$ *Michelle Obama* $\rightarrow$ *alma mater* $\rightarrow$ *Princeton University* and *Harvard Law School* is the answer provided by the algorithm, instead of just the names of the two universities. To determine both precision and recall, triple paths strongly supporting answers are also considered. For the query *'Is Natalie Portman an actress?'*, the expected result is the set of nodes which highly supports the answer for this query, including the triples stating that she is an actress and that she starred in different movies (criteria which is used for both precision and recall). The QALD dataset contains queries with aggregate and conditional operators which were included in the evaluation. However, since Treo does not have a more sophisticated post-processing phase, triples supporting an answer for queries with operators were considered as correct answers. Table 1 contains the quality of results in terms of precision, recall and mrr. Three variations of query sets were taken into account. The first query set (Full DBPedia) evaluates the query mechanism against the 50 queries present in the query set. The second query set had queries with non-dereferenceable pivots (literals and classes) removed (total 42 queries evaluated), while the third category just considered queries which were answered by the approach (27 queries).

The *average query execution time* was 728s with no caching and 353s with active caches using an Intel Centrino 2 machine with 4 GB RAM.

Query Set Type	MRR	Avg. Precision	Avg. Recall
Full DBPedia Training	0.489	0.395	0.451
DBPedia Training (no non-deref. pivots)	0.661	0.534	0.609
DBPedia Training (answered queries)	0.906	0.706	0.805

Table 1: Quality of results for the Treo query mechanism

The experiments revealed two main directions for improvements. The first improvement direction is related to addresssing limitations in the pivot determination process related to the detection of complex-type classes (e.g. pivot classes with more than 2 terms typically from YAGO) and coping with non-dereferenceable pivots (i.e. queries having object literals as pivots). The second improvement direction aims towards coping with multiple candidate PODS representations, taking into account metrics of term informativeness (e.g. TF-IDF). 98% of the final set of PODSs contained the correct ordering in relation to the RDF structure. However, the occurrency of less informative terms before the target terms and the associated process of term-by-term PODS traversal created a semantic discontinuity that the relatedness measure was not able to handle. The use of PODS terms informativeness metrics in the generation of alternative query traversal sequences is a future direction for investigation. One important dimension which was not evaluated in the approach was the generality of the Wikipedia-based relatedness measure beyond the scope of generic datasets such as DBPedia. For these cases the construction of domain-specific distributional relatedness measures which are less dependent on the link structure and organization of Wikipedia (e.g. LSA) are likely be more general solutions across different domains.

8 Related Work

There is an extensive literature in natural language query systems over unstructured and structured data. The analysis of the related work focuses on natural language query approaches for Semantic Web/Linked Data datasets. PowerAqua [11] is a question answering system focused on natural language questions over Semantic Web/Linked Data datasets using PowerMap, a *hybrid matching algorithm comprising terminological and structural schema matching techniques with the assistance of large scale ontological or lexical resources.* PowerMap [15] uses WordNet based similarity approaches as a semantic approximation strategy. NLP-Reduce [13] approaches the query-data matching problem from the perspective of a lightweight natural language approach, where the natural language input query is not analyzed at the syntax level. The matching process between the query terms and the ontology terms present in NLP-Reduce is based on a WordNet expansion of synonymic terms in the ontology and on matching at the morphological level. The matching process of another approach, Querix [14], is also based on the synonyms expansion based on WordNet. Querix, however, uses syntax level analysis over the input natural language query, using this additional structure information to build the corresponding query skeleton of the query. In case ambiguities are detected, a disambiguation dialog is returned for user feedback. Ginseng [12] follows a controlled vocabulary approach: the terms and the structure of the ontologies generate the lexicon and the grammar for the allowed queries in the system. Ginseng ontologies can be manually enriched with synonyms. ORAKEL [21] is a natural language interface focusing on the portability problem across different domains. For this purpose, ORAKEL implements a lexicon engineering functionality, which allows the creation of explicit frame mappings. Instead of allowing automatic approximations, ORAKEL focuses on a precise manually engineered model. Comparatively, Treo provides a query mechanism exploring Wikipedia-based semantic relatedness measures as a semantic approximation technique, where the use of semantic relatedness combined with spreading activation can cope with the heterogeneity of the query-vocabulary problem on Linked datasets on the Web. In addition, Treo's design supports the query of dynamic and distributed Linked Data by relying on entity search and sequential dereferenciations. Treo also differentiates itself from existing approaches in the query strategy: (i) instead of building a SPARQL query, Treo navigates the Linked Data nodes from a pivot node and (ii) Treo focuses on a best-effort ranked approach.

9 Conclusion & Future Work

This work describes *Treo*, a natural language query mechanism for Linked Data. The cognitively inspired architecture of Treo, combining *entity search*, *spreading activation* and *semantic relatedness* is designed to cope with critical features for natural language queries over Linked Data, including the ability to query high volume, heterogeneous, distributed and dynamic data. The approach was evaluated using the QALD query datasets containing 50 natural language queries over DBPedia, achieving an overall *mean reciprocal rank* of 0.489, *precision* of 0.395 and *recall* of 0.451, answering 56% of the queries. The proposed query mechanism approaches natural language queries over Linked Data using a best-effort approach where results are displayed in the form of triple paths, ranked paths containing the desired entities in the Linked Data Web. Future work will concentrate in two main directions: the improvement of the query execution time of the approach, exploring optimizations through parallelization, indexing and pipelining [23] strategies and the introduction of answer post-processing techniques for the generation of answers from triple paths.

Acknowledgments. The work presented in this paper has been funded by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2).

References

1. Kaufmann, E., Bernstein, A.: Evaluating the usability of natural language query languages and interfaces to Semantic Web knowledge bases. Web Semantics: Science, Services and Agents on the World Wide Web 8 393-377 (2010).
2. Marneffe, M., MacCartney, B. and Manning, C. D., Generating Typed Dependency Parses from Phrase Structure Parses. In LREC 2006 (2006).
3. Finkel, J.R., Grenager, T. , and Manning, C. D., Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005), pp. 363-370 (2005).
4. Toutanova, K., Klein, D., Manning, C.D. and Singer, Y., Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In Proceedings of HLT-NAACL 2003, pp. 252-259 (2003).
5. Delbru, R., Toupikov, N., Catasta, M., Tummarello, G., Decker, S.: A Node Indexing Scheme for Web Entity Retrieval. In Proceedings of the 7th Extended Semantic Web Conference (ESWC) (2010).
6. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S.r., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. Web Semantics: Science, Services and Agents on the World Wide Web 7 (2009).
7. 1st Workshop on Question Answering over Linked Data (QALD-1), `http://www.sc.cit-ec.uni-bielefeld.de/qald-1` (2011).
8. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using Wikipedia-based explicit semantic analysis. International Joint Conference On Artificial Intelligence (2007).
9. Deerwester, S., Dumais, S.T., Furnas, G.W.: Indexing by Latent Semantic Analysis. Journal of the American Society for Information Science 41 (6) 391407 (1990).
10. Milne, D. and Witten, I.H. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In Proceedings of the first AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI'08), Chicago, I.L. (2008).
11. Lopez, V., Motta, E., Uren, V.: PowerAqua: Fishing the Semantic Web. Proc 3rd European Semantic Web Conference ESWC, Vol. 4011. Springer 393-410 (2004).
12. Bernstein, A., Kaufmann, E., Kaiser, C., Kiefer, C.: Ginseng A Guided Input Natural Language Search Engine for Querying Ontologies. Jena User Conference 2006 (2006).
13. Kaufmann, E., Bernstein, A., Fischer, L.: NLP-Reduce: A naive but Domain-independent Natural Language Interface for Querying Ontologies. 4th European Semantic Web Conference ESWC 2007 1-2 (2007).
14. Kaufmann, E., Bernstein, A., Zumstein, R.: Querix: A Natural Language Interface to Query Ontologies Based on Clarification Dialogs. 5th International Semantic Web Conference (ISWC). Springer 980-981 (2006).
15. Lopez, V., Sabou, M., Motta, E.: PowerMap: Mapping the Real Semantic Web on the Fly. International Semantic Web Conference, Vol. 4273. Springer 5-9 (2006).
16. Cohen, P.: Information retrieval by constrained spreading activation in semantic networks, Information Processing & Management, Vol. 23, no. 4, pp. 255-268 (1987).
17. Conference on Computational Natural Language Learning (CoNLL-2003), http://www.clips.ua.ac.be/conll2003/ (2003).
18. Crestani, F. Application of Spreading Activation Techniques in Information Retrieval, Artificial Intelligence Review, vol. 11, no. 6, pp. 453-453 (1997).

19. Collins, A.M., Loftus, E.F.: A spreading activation theory of semantic processing, Psychological Review, vol. 82, no. 6, pp. 407-428 (1975).
20. Quillian, M.R.: Semantic Memory, Semantic Information Processing, MIT Press, pp. 227-270 (1968).
21. Cimiano, P., Haase, P., Heizmann, J., Mantel, M. and Studer,R.: Towards portable natural language interfaces to knowledge bases: The Case of the ORAKEL system, Data Knowledge Engineering (DKE), 65(2), pp. 325-354 (2008).
22. Dong, X., Halevy, A.: Indexing Dataspaces, In Proceedings of the ACM SIGMOD (2007).
23. Hartig, O., Bizer, C., and Freytag, J.C.: Executing SPARQL Queries over the Web of Linked Data, Proceedings of the 8th International Semantic Web Conference (2009).
24. Harth, A., Hose, K., Karnstedt, M., Polleres, A., Sattler, K.-U. and Umbrich, J.: Data summaries for on-demand queries over linked data, in Proceedings of the 19th international conference on World Wide Web, USA (2010).
25. Cilibrasi, R.L. and Vitanyi, P.M.B.: The Google Similarity Distance, IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 3, 370-383 (2007).
26. Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Pasca, M. and Soroa, A., A study on similarity and relatedness using distributional and WordNet-based approaches, In Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, 19-27 (2009).

A Natural Language Interface over the MusicBrainz Database

Johan Granberg and Michael Minock

Department of Computing Science: Umeå University

Abstract. This paper demonstrates a way to build a natural language interface (NLI) over semantically rich data. Specifically we show this over the MusicBrainz domain, inspired by the second shared task of the QALD-1 workshop. Our approach uses the tool C-Phrase [1] to build an NLI over a set of views defined over the original MusicBrainz relational database. C-Phrase uses a limited variant of X-Bar theory [2] for syntax and tuple calculus for semantics. The C-Phrase authoring tool works over any domain and only the end configuration has to be redone for each new database covered – a task that does not require deep knowledge about linguistics and system internals. Working over the MusicBrainz domain was a challenge due to the size of the database – quite a lot of effort went into optimizing computation times and memory usage to manageable levels. This paper reports on this work and anticipates a live demonstration[1] for querying by the public.

Keywords: Natural Language Interfaces, Relational Databases, MusicBrainz, C-Phrase

1 Introduction

It has often been noted that natural language interfaces to databases suffer when the manner in which data is stored does not correspond to the user's conceptual view of such data [3–5]. This mismatch between the way data is structured and the user's conceptual model can be for a variety of reasons, but here we speculate that the three most common reasons are:

1. The database is highly normalized (e.g. to BCNF) for the sake of eliminating update anomalies.
2. The database is highly abstracted, including many attributes per relation so as to avoid cost associated with joins.
3. The database is stored in a semi-structured form corresponding to RDF triples.

It is the thesis of this paper that the user would prefer to query the data in a conceptual form similar to what is represented in the entity-relationship diagram

[1] `http://www.cs.umu.se/~johang/research/brainz/public/`

of the database domain (see figure 1). Naturally if we are given a database in one of the three forms above, we assume that it is possible, via standard view definitions, to transform the data so that it may be accessed (and perhaps even updated) via the conceptual model.

This paper explores these ideas and shows some preliminary results over the MusicBrainz database paired with the 50 natural language queries in the shared task for MusicBrainz in the QALD-1 workshop. In section 2 we present our approach to building a natural language interface supporting queries over MusicBrainz. Section 3 discusses our initial results and some anecdotes from our development efforts. Section 4 summarizes our findings and points toward near term and longer term plans, including the fielding of a live interface to MusicBrainz for querying by the public.

2 Approach

2.1 The conceptual model

After looking at the set of 50 natural language queries to be supported over MusicBrainz, we defined the conceptual model appearing in figure 1. This diagram is traditional except that it shows a form of conceptual aggregation. For example a group or a person can release an album, soundtrack or single. A traditional ER diagram would require six relationship diamonds instead of one to depict this.

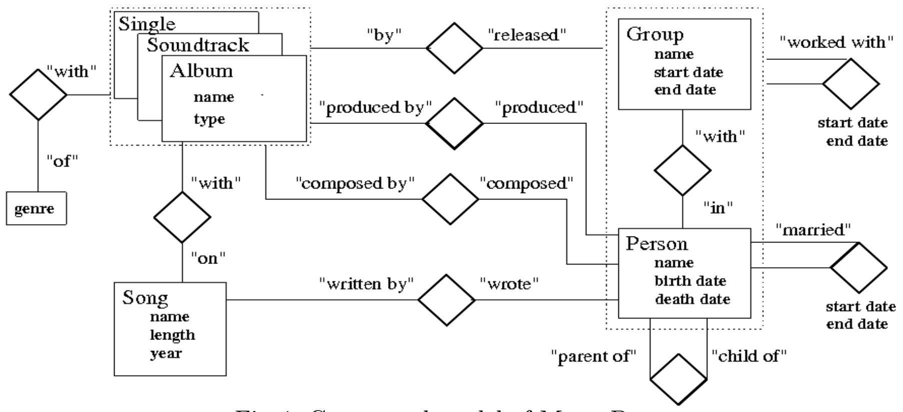

Fig. 1: Conceptual model of MusicBrainz.

2.2 The data source

We had two choices for what we used as the data source for the actual MusicBrainz data. The first was the original data stored in a PostgreSQL database and the second was the RDF dump of the data built for the QALD-1 shared task. Because of our rooting in relational databases, we chose to simply draw on the data in the original MusicBrainz PostgreSQL database.

The schema of the underlying MusicBrainz database is primarily designed according to option 2 from section 1 above. That is, several highly abstracted relations with many attributes represent abstract entities such as `L_ARTIST_ARTIST` and `LT_ARTIST_ARTIST` which has tuples that represent both individual artists (e.g. Bob Dylan) as well as bands (e.g. REM). The database itself is rather large. There are approximately 10.6 million songs, 0.8 million albums and 0.6 million individual and bands. In our experiment we remove the tuples containing non-ASCII characters and arrive at 9.7 million songs, 0.7 million albums and 0.4 million artists.

2.3 The view definitions

Views were defined in the standard way over the base relations of the MusicBrainz database. One consideration was whether to materialize these views for quicker access to the data. This essentially doubles the database size. Our findings are that this leads to a speed up factor of approximately 3. For example using regular views the test query, "which singles did the Dead Kennedys release?" took 2.0 ms on average. Using materialized views it took an average of 0.7 ms. Considering other performance issues in the system (e.g. natural language parsing times), we concluded that querying through the views is not currently a bottleneck. Thus we did not elect to materialize views.

2.4 Authoring with the C-Phrase administration interface

Once several errors were dealt with (see section 3.1) the authoring process proceeded well. It followed the name, tailor and define method laid out in [1]. Well over half the training set can be authored for within 90 minutes. With our new enhancements to the parser to better handle WH-movement, this may be further reduced. We intend to produce series of YouTube videos that demonstrate this process.

3 Preliminary Results

3.1 Difficulties

There were several difficulties we encountered that, while perhaps anecdotal, are still worth mentioning. To date the C-Phrase system has been applied only over small databases. MusicBrainz is a sizable database, so this brought up some scalability issues that we had not earlier experienced. We assume other NLIs attempting to scale to this size of a database might face similar problems.

Large main-memory hash tables The first issue related to memory management issues in CLISP, the version of LISP that C-Phrase is implemented over. There are some unfortunate memory bugs that corrupt CLISP memory when utilization climbs over a certain threshold. It is difficult to track, because the corruption generally causes a `Segmentation Fault` at later steps when memory is accessed. Under normal circumstances this problem does not surface. However to allow for named entity recognition, C-Phrase materializes string values from the database into hash tables to scan for matching values in the user's typed request. In the case of MusicBrainz this means building main memory hash tables containing millions of constants. This was too much for CLISP to handle.

After several false starts, the solution to this problem was to implement a remote hash facility that maintained the main memory hash tables remotely outside of CLISP. The overhead access time for these hash tables is negligible and the implementation is stable and scalable, bounded ultimately by the size of virtual memory.

Limitations of the PostgreSQL **query optimizer** C-Phrase maps English to logical expressions in Codd's tuple calculus. From such logical expressions, SQL is in turn generated. Before our experiment we would generate SQL such as the following to answer the query, "Which singles did the Dead Kennedys release?"

```
SELECT DISTINCT NAME
FROM SINGLE AS x
WHERE
 EXISTS(
   SELECT *
   FROM BAND as y1
   WHERE
     x.artist = y1.id  AND
     y1.name = 'Dead Kennedys').
```

Unfortunately such queries are not taken up by PostgreSQL's optimizer. This query in fact takes 23 minutes to answer on an older Solaris server where we run our database. In contrast the equivalent query

```
SELECT DISTINCT x.NAME
FROM SINGLE AS x,BAND as y1
WHERE x.artist = y1.id  AND y1.name = 'Dead Kennedys'"
```

takes 0.7 seconds on the same server. Here the query time is entirely dominated by the time to establish the connection, the actual query execution is reported as 2 ms. We assume that the speed-up is due to the fact that the optimizer has access to both relations on the second line of the query and can plan accordingly. Instead of pestering PostgreSQL about 'improving their optimizer', we altered our translator to produce SQL of the later variety.

3.2 Performance

We are still in the process of collecting performance data. The running example query of "Which singles did the Dead Kennedys release?" shows that the performance is adequate in the case of single join queries. We have run additional tests on queries that exercise more joins and are convinced that the current approach is feasible. The time it takes to parse natural language queries is typically in the range of one to three seconds.

3.3 Current coverage

Unfortunately recent extensions to C-Phrase have introduced system instability and have blocked a systemic precision, recall and f-measure study on the 50 unseen queries released as part of the QALD-1 shared task. The problems are mostly due to unanticipated parser bugs and performance problems when extending our parser to handle gap threading. Work continues to resolve these problems. In the meantime we have elected not to read the 50 new unseen queries in anticipation of doing a clean coverage test in the future.

As for the original 50 queries for the MusicBrainz example, in our prior working version of C-Phrase, we covered all but 5 of these queries. The 5 problematic queries (in order of estimated level of difficulty) are those involving implied time intervals ("How many bands broke up in 2010"), types (e.g. "Is Liz Story a person or a group?"), complex time calculations ("Which artists have their 50th birthday on May 30, 2011?"), computed comparison values with ellipsis (e.g. "Which artists died on the same day as Michael Jackson"), and queries involving non-quoted, non-domain string values (e.g. "Are the members of the Ramones that are not called *Ramone*"). Time permitting, we will extend C-Phrase to handle these types of queries. It will be interesting to learn about how other groups at QALD-1 approached these queries.

4 Conclusions

We were very pleased when we heard of the QALD-1 shared task. We decided to focus our efforts on the more closed-domain task of queries over MusicBrainz. We based our data on the original MusicBrainz relational database and, after confronting several technical difficulties in scaling C-Phrase, we managed to build a natural language interface that covered 45 of the 50 training examples in the QALD-1 shared task for MusicBrainz.

Unfortunately technical problems have delayed a systematic evaluation. We still intend to perform and report an evaluation of how well we cover the 50 unseen queries, but perhaps more tellingly we intend to field a natural language interface[2] for real-time querying of MusicBrainz by the public.

[2] `http://www.cs.umu.se/~johang/research/brainz/public/`

References

1. Minock, M.: C-Phrase: A System for Building Robust Natural Language Interfaces to Databases. Journal of Data and Knowledge Engineering 3:69, 290–302 (2010)
2. Jackendoff, R.: X-bar-Syntax: A Study of Phrase Structure. Linguistic Inquiry Monograph 2. MIT Press (1977)
3. Copestake, A., Sparck Jones, K.: Natural Language Interfaces to Databases. The Natural Language Review 5(4), 225–249 (1990)
4. Androutsopoulos, I., Ritchie, G.D.: Database Interfaces. In: Dale, R., Moisl, H., Somers, H. (eds) Handbook of Natural Language Processing, pp. 209–240. Marcel Dekker Inc. (2000)
5. Ogden, W., Bernick, P.: Using Natural Language Interfaces. Technical report MCCS-96-299, Computing Research Laboratory, New Mexicon State University, Las Cruces (1996)