Expressive Languages for Querying the Semantic Web

Marcelo Arenas[1] and Georg Gottlob[2] and Andreas Pieris[2]

[1]Department of Computer Science, PUC Chile, Chile
[2]Department of Computer Science, University of Oxford, UK
marenas@ing.puc.cl, {georg.gottlob, andreas.pieris}@cs.ox.ac.uk

The Resource Description Framework (RDF) is the W3C recommendation data model to represent information about World Wide Web resources. An atomic piece of data in RDF is a *Uniform Resource Identifier (URI)*. In the RDF data model, URIs are organized as RDF graphs, that is, labeled directed graphs where node labels and edge labels are URIs. The problem of querying RDF data is a central issue for the development of the Semantic Web. The query language SPARQL has become the standard language for querying RDF, since its standardization in 2008 [8]. However, the 2008 version of this language missed some important functionalities, and in particular, reasoning capabilities to deal with *RDF Schema (RDFS)* [2] and *Ontology Web Language (OWL)* [6] vocabularies, that is, vocabularies with predefined semantics which can be used to derive logical conclusions from RDF graphs. To overcome these limitations, a new version of SPARQL, called SPARQL 1.1 [5], was recently released, which includes entailment regimes for RDFS and OWL vocabularies.

Unfortunately, even if we focus on a lightweight profile of OWL 2 such as OWL 2 QL, the queries may become extremely complicated. This is mainly due to the fact that we are forced to encode the semantics of the RDFS and OWL vocabularies in the query. The aim of this work is to propose a language which separates the reasoning part from the actual query, and thus overcoming the above negative aspect of the existing query languages. Our language, called TriQ, is based on Datalog and some of its extensions; in particular, the members of the recently introduced Datalog$^{\pm}$ family of knowledge representation and query languages [3]. In the rest of this short paper, we illustrate, via a simple example, some of the difficulties encountered when querying RDF data with SPARQL, which motivated us to design TriQ.

Motivating Example

Assume that G_1 is an RDF graph containing the triples

$$\begin{array}{l}(\text{dbUllman},\ \text{is\_author\_of},\ \text{“The Complete Book”})\\ (\text{dbUllman},\ \text{name},\ \text{“Jeffrey Ullman”}).\end{array}$$

The first triple indicates that the object with URI dbUllman is one of the authors of the book “The Complete Book”, while the second triple indicates that the name of dbUllman is “Jeffrey Ullman”. To retrieve the list of authors mentioned in G_1 we can use the following SPARQL query:

$$\begin{array}{l}\text{SELECT } ?X\\ (?Y, \text{is\_author\_of}, ?Z)\ \text{AND}\ (?Y, \text{name}, ?X).\end{array} \qquad (1)$$

We use here the algebraic syntax for SPARQL introduced in [7]. Notice that variables start with the symbol ?. As mentioned above, one of the distinctive features of Semantic Web data is the use of the RDFS and OWL vocabularies. As an example of this, assume that G_2 is an RDF graph consisting of the following triples:

$$\begin{aligned}
&(\text{dbUllman},\ \text{is\_author\_of},\ \text{“The Complete Book”})\\
&(\text{dbUllman},\ \text{name},\ \text{“Jeffrey Ullman”})\\
&(\text{dbAho},\ \text{is\_coauthor\_of},\ \text{dbUllman})\\
&(\text{dbAho},\ \text{name},\ \text{“Alfred Aho”})\\
&(\text{r}_1,\ \text{rdf:type},\ \text{owl:Restriction})\\
&(\text{r}_1,\ \text{owl:onProperty},\ \text{is\_coauthor\_of})\\
&(\text{r}_1,\ \text{owl:someValuesFrom},\ \text{owl:Thing})\\
&(\text{r}_2,\ \text{rdf:type},\ \text{owl:Restriction})\\
&(\text{r}_2,\ \text{owl:onProperty},\ \text{is\_author\_of})\\
&(\text{r}_2,\ \text{owl:someValuesFrom},\ \text{owl:Thing})\\
&(\text{r}_1,\ \text{rdfs:subClassOf},\ \text{r}_2).
\end{aligned} \tag{2}$$

In G_2, the URIs with prefix rdfs: are part of the RDFS vocabulary, while the URIs with prefix owl: are part of the OWL vocabulary. More precisely, the third triple above indicates that the object with URI dbAho is a coauthor of the object with URI dbUllman. The fifth, sixth and seventh triples of G_2 define r_1 as the class of URIs a for which there exists a URI b such that $(a, \text{is\_coauthor\_of}, b)$ holds, while the following three triples of this graph define r_2 as the class of URIs a for which there exists a URI b such that the triple $(a, \text{is\_author\_of}, b)$ holds. Finally, the last triple of G_2 indicates that r_1 is a subclass of r_2.

The last seven triples of G_2 indicate that for every pair a, b of elements such that $(a, \text{is\_coauthor\_of}, b)$ holds, it must be the case that a is an author of some publication. Thus, if we want to retrieve the list of authors mentioned in G_2, then we expect to find dbAho in this list. However, the answer to the SPARQL query (1) over G_2 does not include this URI, and we are forced to encode the semantics of the RDFS and OWL vocabularies in the query. In fact, even if we try to obtain the right answer by using SPARQL 1.1 under the entailment regimes for these vocabularies, we are forced by the restrictions of the language [4] to replace the triple $(?Y, \text{is\_author\_of}, ?Z)$ in (1) by:

$$\begin{aligned}
&(?Y, \text{rdf:type}, ?Z)\ \text{AND}\\
&\quad (?Z, \text{rdf:type}, \text{owl:Restriction})\ \text{AND}\\
&\quad (?Z, \text{owl:onProperty}, \text{is\_author\_of})\ \text{AND}\\
&\quad (?Z, \text{owl:someValuesFrom}, \text{owl:Thing}),
\end{aligned}$$

which indicates that we are looking for the objects that are authors of some publication (that is, the objects of type r_2).

As the reader may have noticed, the resulting query is very complicated. In the query language proposed in this paper, the user can use separate modules to encode reasoning capabilities and actual queries. In particular, the user first needs to utilise a module for the RDFS and OWL vocabularies (or for some fragment of them), that could consist of

Datalog$^{\exists,\neg,\bot}$ rules such as the following:

$$\begin{aligned}&\text{triple}(?X, \text{rdf:type}, ?Y),\\&\text{triple}(?Y, \text{rdf:type}, \text{owl:Restriction}),\\&\text{triple}(?Y, \text{owl:onProperty}, ?Z),\\&\text{triple}(?Y, \text{owl:someValuesFrom}, ?U) \ \rightarrow\ \exists ?W\, \text{triple}(?X, ?Z, ?W).\end{aligned}$$

In this module, the predicate triple is used to store the triples of the RDF graphs. Notice that the rules of the module are used to encode the semantics of the respective vocabulary. Besides, these rules are fixed, they do not depend on the query that the user is trying to answer. Thus, to pose the desired query, the user just need to write on top of this module a simple query similar to (1):

$$\text{triple}(?Y, \text{is\_author\_of}, ?Z), \text{triple}(?Y, \text{name}, ?X) \ \rightarrow\ \text{query}(?X). \qquad (3)$$

In particular, (s)he does not need any prior knowledge about the semantics and inference rules for the respective vocabulary. In fact, the module for encoding this vocabulary can be publicly available, thus greatly simplifying the process of writing queries.

The modular structure of TriQ queries is very convenient to deal with SPARQL queries over the OWL vocabulary. In fact, if we focus on the OWL 2 QL profile of OWL 2, that is designed to be used in applications where query answering is the most important reasoning task, then it can be shown that every SPARQL query under the entailment regime for OWL 2 QL can be naturally translated into a TriQ query. Moreover, we can show that the use of TriQ allows us to formulate SPARQL queries in a simpler way, as a more natural entailment regime can be easily defined by using this query language. For more details about the TriQ language, we refer the interested reader to [1].

Acknowledgements. M. Arenas was supported by the Millennium Nucleus Center for Semantic Web Research under Grant NC120004 and Fondecyt grant 1131049, G. Gottlob was supported by ERC grant 246858 (DIADEM), and A. Pieris was supported by EPSRC grant EP/J008346/1 (PrOQAW).

References

1. M. Arenas, G. Gottlob, and A. Pieris. Expressive languages for querying the semantic web. In *Proc. of PODS*, 2014. To appear.
2. D. Brickley and R. Guha. RDF vocabulary description language 1.0: RDF schema. W3C Recommendation 10 February 2004, http://www.w3.org/TR/rdf-schema.
3. A. Calì, G. Gottlob, T. Lukasiewicz, B. Marnette, and A. Pieris. Datalog+/-: A family of logical knowledge representation and query languages for new applications. In *Proc. of LICS*, pages 228–242, 2010.
4. B. Glimm and C. Ogbuji. SPARQL 1.1 entailment regimes. W3C Recommendation 21 March 2013, http://www.w3.org/TR/sparql11-entailment/.
5. S. Harris and A. Seaborne. SPARQL 1.1 query language. W3C Recommendation 21 March 2013, http://www.w3.org/TR/sparql11-query/.
6. D. L. McGuinness and F. van Harmelen. OWL web ontology language overview. W3C Recommendation 10 February 2004, http://www.w3.org/TR/owl-features/.
7. J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of sparql. *ACM Trans. Database Syst.*, 34(3), 2009.
8. E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF. W3C Recommendation 15 January 2008, http://www.w3.org/TR/rdf-sparql-query.