Terp: Syntax for OWL-friendly SPARQL queries

Evren Sirin[1], Blazej Bulka[1], and Michael Smith[1]

Clark & Parsia, LLC, Washington, DC, USA
{evren,blazej,msmith}@clarkparsia.com

1 Introduction

Web Ontology Language (OWL) [5] can be seen as an extension of Resource Description Framework (RDF). The primary exchange syntax for OWL is RDF/XML, and every OWL ontology can be represented as an RDF graph. But there is no standard query language specifically for OWL ontologies. The most commonly used Semantic Web query language is SPARQL [7], which is intended to be used for RDF. Roughly speaking, SPARQL is specified as queries matching RDF graphs with simple RDF entailment. However, it allows this definition to be extended to OWL entailment. A semantics for SPARQL compatible with OWL DL has been defined in SPARQL-DL [8] and a similar formalization of OWL-compatible SPARQL semantics is being developed by the W3C's SPARQL Working Group as part of SPARQL 1.1.[1]

The semantics extension of SPARQL allows one to query OWL ontologies and get the expected results with respect to OWL entailments. However, writing SPARQL queries that involve complex OWL expressions ranges from challenging to unpleasant because SPARQL query syntax is based on Turtle [1], which isn't intended for OWL. SPARQL queries against OWL data have to encode the RDF serialization of OWL expressions: these queries are typically verbose, difficult to write, and difficult to understand.

In this paper we present Terp, a new syntax that combines Turtle and Manchester syntaxes to provide maximum legibility and conciseness when querying OWL with SPARQL. More precisely, Terp syntax allows class, property, and data range expressions, expressed in Manchester syntax, to be used inside SPARQL queries. In this paper, we provide examples to demonstrate how Terp reuses existing features from well-known syntaxes to make SPARQL queries of OWL data more concise and more legible.

2 Terp Syntax

SPARQL syntax provides many Turtle-derived shortcuts for writing concise queries. Such features are especially useful for instance queries. The following query against Wine ontology[2] shows several feature of the SPARQL syntax.[3]

[1] http://www.w3.org/TR/sparql11-entailment/
[2] http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine.rdf
[3] We omit prefix declarations in SPARQL queries due to space constraints.

```
SELECT ?wine ?flavor WHERE {
   ?wine a :FrenchWine , :RedWine ;
         :hasFlavor ?flavor
}
```

Ex. 1. [SPARQL] Find the flavors of red wines produced in France.

This query uses several Turtle shortcuts: 1. the Turtle keyword `a` instead of `rdf:type` qname, 2. object lists separated by comma when subject and predicate of triples is the same, and 3. predicate-object lists separated by semi-colon when only the subject of triples is the same.

But the shortcuts provided by Turtle are inadequate when OWL expressions are used inside SPARQL queries.[4] The following example shows the effect of using simple OWL constructs such as intersection and cardinality restrictions.

```
SELECT ?wine WHERE {
   ?wine a [ owl:intersectionOf (
         wine:Wine
         [  owl:onProperty wine:madeFromGrape ;
            owl:minCardinality 2
         ])]
}
```

Ex. 2. [SPARQL] Find wine instances that are made of at least two grapes.

Example 2 is very verbose even though it uses Turtle's bnode and list shortcuts. And the OWL keywords are mixed with the domain elements making it harder to read the query.

Manchester syntax [2] is commonly used for concisely writing OWL expressions. It was originally designed to express class expressions in OWL 1 and then later extended to write entire OWL 2 ontologies. It is used in RDF/OWL editors such as Protégé 4 and TopBraid Composer.

Terp extends SPARQL syntax by allowing class, property, and data range expressions expressed in Manchester syntax to be used in queries. The following example shows how this combination results in a much more concise query.

```
SELECT ?wine  WHERE {
  ?wine a ( wine:Wine and wine:madeFromGrape min 2 )
}
```

Ex. 3. [Terp] Example 2 written in Terp syntax.

Terp is designed to support everything in SPARQL syntax, including all of the abbreviations of Turtle, to which it adds Manchester syntax features to represent OWL expressions. The following example shows a valid Terp query that uses several features of SPARQL and Manchester syntaxes.

The grammar for Terp can be found online.[5] This grammar is nearly an exact merge of SPARQL grammar with Manchester syntax grammar. As a result,

[4] This isn't really a criticism of Turtle's design, since representing OWL constructs compactly was not a Turtle design desideratum.

[5] `http://clark-parsia.svn.cvsdude.com/pellet-devel/tags/release-2.1.0/query/antlr/SparqlOwl.g`

```
SELECT ?mealCourse ?label WHERE {
   ?mealCourse rdfs:subClassOf
         food:MealCourse ,
         food:hasDrink some (wine:Wine and
                             wine:hasBody value wine:Full)
   OPTIONAL {
      ?mealCourse rdfs:label ?label
   }
   FILTER ( ?mealCourse != owl:Nothing )
}
ORDER BY ?label
```

Ex. 4. [Terp] Find meal courses that go with full-bodied wines, optionally find the associated label. Filter owl:Nothing from results and order by labels.

Manchester syntax expressions can appear in the subject and object position of SPARQL triple patterns. Ultimately, any Terp query can be translated to pure SPARQL queries in a straight-forward way by using the translation rules for Manchester syntax [2].[6]

There are a few cases where ambiguity arises due to the combination of grammars. For example, the use of parentheses to indicates lists in SPARQL and nesting in Manchester syntax. If the ambiguity cannot be resolved by the context, Terp grammar assumes the SPARQL intention, primarily because SPARQL is known to more people (or so we reasonably assume) than Manchester.

3 Implementation

Experimental Terp support is available in Pellet version 2.1.[7] The Pellet distribution contains many more examples of Terp queries. Terp can be used from the Pellet command-line to execute queries written in Terp syntax. Terp can also be accessed programmatically: it reads Terp queries and generates standard SPARQL queries that can be executed by *any OWL-aware SPARQL endpoint.*

4 Related Work

There have been several efforts to create a query language for OWL. SPARQLAS[8] is one example where OWL functional syntax with variables and Manchester syntax like abbreviations is used to encode queries. Similar to Terp, there is a well-defined translation from SPARQLAS queries to pure SPARQL. SPARQLAS certainly makes writing OWL expressions easier than in standard SPARQL; however, it supports only conjunctive queries without any SPARQL operators (i.e., UNION, OPTIONAL, or FILTER).

OWLLink [4] protocol provides a query language that is intended to be used for OWL. The OWLLink core provides a set of general requests, called *basic*

[6] Manchester syntax defines a mapping to OWL 2 functional syntax for which an RDF mapping is defined.

[7] `http://clarkparsia.com/pellet`

[8] `http://code.google.com/p/twouse/wiki/SPARQLAS`

asks, for retrieving entailments and a query extension for more complex union of conjunctive queries. There is both a functional syntax and XML syntax binding for these queries. However, OWLLink is not designed to improve the concision or legibility of SPARQL queries for OWL.

SQWRL (Semantic Query-enhanced Web Rule Language) [6] is another query language for OWL; it's based on human-readable syntax of SWRL [3]. In principle, the SWRL specification allows class expressions in human-readable syntax, but there is no formal grammar about how this can be done. SWRL syntax is only suitable for instance queries and schema queries need to be handled by SQWRL built-in function extensions.

5 Conclusions and Future Work

In this paper, we provide a brief description of Terp syntax which allows using Manchester syntax expressions in SPARQL for concise and readable OWL queries. Terp is mainly intended to be easy to read and write for humans. Since Terp queries can be translated to SPARQL queries, this syntax can be used in conjunction with standard SPARQL Protocol to communicate with SPARQL endpoints, since the translation can be done on either client or server side. Terp can be useful even if SPARQL queries are auto-generated through a UI, especially when those queries need to be inspected or debugged by people.

Terp syntax is still evolving and there are several OWL 2 features such as negative property assertions and axiom annotations that it does not provide syntactic shortcuts. Furthermore, there are several extensions possible for Terp such as allowing Manchester syntax keywords in predicate position (e.g. use `equivalentTo` to replace `owl:equivalentClass` and `owl:equivalentProperty`) or additional syntactic sugar for some frequently used OWL 2 constructs.

References

1. D. Beckett and T. Berners-Lee. Turtle - Terse RDF Triple Language. W3C Team Submission, 2008.
2. M. Horridge and P. F. Patel-Schneider. OWL 2 web ontology language manchester syntax. W3C Working Group Note, 2009.
3. I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A semantic web rule language combining OWL and RuleML. W3C, 2004.
4. T. Liebig, M. Luther, and O. Noppens. The OWLlink Protocol. In *Proc. of 6th OWL: Experiences and Directions Workshop (OWLED2009)*, 2009.
5. B. Motik, P. F. Patel-Schneider, and B. Parsia. OWL 2 web ontology language structural specification and functional-style syntax. W3C Recommendation, 2009.
6. M. J. O'Connor and A. K. Das. SQWRL: A Query Language for OWL. In *Proc. of 6th OWL: Experiences and Directions Workshop (OWLED2009)*, 2009.
7. E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. W3C Recommendation, 2008.
8. E. Sirin and B. Parsia. SPARQL-DL: SPARQL Query for OWL-DL. In *Proc. of 3rd OWL: Experiences and Directions Workshop (OWLED2007)*, 2007.