Querying Description Logics

Petr Křemen

1 SPARQL and Ontology Querying

SPARQL Language [SS13] is aimed at querying RDF(S) [GB04] documents. As OWL 2 [MPSP09] is an extension of RDF(S), we will use it here as a syntax for OWL 2 conjunctive queries.
We will use SPARQL queries of the form

```
PREFIX <prefix1>:<URI1>
PREFIX <prefix2>:<URI2>
SELECT <vars>
WHERE {
      <triple1> .
      <triple2> .
      ...
      <tripleN> .
}
```

where `<vars>` is a list of variables (identifier started with a sign "?") delimited with spaces and `<tripleI>` is a triple of the form `subj pred obj`, where each `subj`, `pred` and `obj` is either a variable, or individual URI (also in a shortened form using a PREFIX). Additionally, `obj` can be literal (string in double-quotes).

Example 1.1. Consider a conjunctive query (without full URIs)

$$Q_1(?X, ?Y) \leftarrow Professor(?X), worksFor(?X, CVUT), name(?X, ?Y). \quad (1)$$

Its SPARQL counterpart is :

```
PREFIX u: <http://krizik.felk.cvut.cz/university.owl#>

SELECT ?X ?Y
WHERE {
  ?X a u:Professor .
  ?X u:worksFor u:CVUT .
  ?X u:name ?Y .
}
```

If we are not interested in the binding of the particular name Y, we would pose a conjunctive query

$$Q_2(?X) \leftarrow Professor(?X), worksFor(?X, CVUT), name(?X, ?Y) \tag{2}$$

To reflect this change in SPARQL, we could simply remove the variable from the SELECT clause, obtaining:

```
PREFIX u: <http://krizik.felk.cvut.cz/university.owl#>

SELECT ?X
WHERE {
  ?X a u:Professor .
  ?X u:worksFor u:CVUT .
  ?X u:name ?Y .
}
```

However, this SPARQL query doesn't return individuals (bindings for $?X$) for which there is no name known, but their existence is inferred (e.g. i in axiom $(\exists name \cdot \top)(i)$). To capture all cases, we need to use a *bnode* instead of the (distinguished) variable:

```
PREFIX u: <http://krizik.felk.cvut.cz/university.owl#>

SELECT ?X
WHERE {
  ?X a u:Professor .
  ?X u:worksFor u:CVUT .
  ?X u:name _:Y .
}
```

2 Conjunctive Queries Practically

1. Download the Pellet 2.3.1 (e.g. from the seminar web pages, or through `http://clarkparsia.com/pellet/download/pellet-2.3.1`).
2. Pellet is a command-line tool. Use `./pellet.sh help` command to find out details about its usage.
3. Download the wine ontology from `http://www.w3.org/TR/owl-guide/wine.rdf` and save it into the Pellet home directory.
4. Open the wine ontology in Protege and insert a new instance of `Wine` into the ontology.
5. Download an example query from the seminar web pages. This query finds all regions (instancec of `Region`) :

```
pellet.bat query -q <file-with-query> <file-with-ontology>
```

6. In the file with a query, replace the distinguished variable `?Y` with an undistinguished variable `_:Y` and compare results (use the `--bnode` switch for Pellet)

7. Check that you got the same result as in the DL-query tab (How to formulate such query ?).

8. Formulate and evaluate a query that
   - finds all regions in USA together with dry wines produced in these regions.
   - finds all regions in USA that produce both dry and sweet wines.

9. Insert a new type `locatedIn some Region` to the individual `ItalianRegion`. Then, formulate a query that finds all wines that are produced in some (arbitrary) super-region of Italy (i.e. region in which `ItalianRegion` is located in (`locatedIn`)). Use the `--bnode` parameter in the Pellet command line to correctly evaluate the query.

Reference

[GB04] Ramanathan V. Guha and Dan Brickley. RDF Vocabulary Description Language 1.0: RDF Schema [online]. W3C Recommendation, W3C, 2004. Available at `http://www.w3.org/TR/2004/REC-rdf-schema-20040210`, cit. 11/1/2012.

[MPSP09] Boris Motik, Peter F. Patel-Schneider, and Bijan Parsia. OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax [online]. W3C Recommendation, W3C, 2009. Available at `http://www.w3.org/TR/2009/REC-owl2-syntax-20091027`, cit. 11/1/2012.

[SS13] Andy Seaborne and Harris Steve. SPARQL 1.1 query language [online]. W3C Recommendation, W3C, 2013. Available at `http://www.w3.org/TR/sparql11-query`, cit. 1/4/2013.