Querying Non-Materialized Ontology Views

Landon T Detwiler, MS, James F Brinkley, MD, PhD
Structural Informatics Group, Departments of Biological Structure and Medical Education and Biomedical Informatics, University of Washington, Seattle, WA 98195

One approach to simplifying ontologies, for inclusion in a more tractable semantic web, is through the use of non-materialized view queries. View queries define how a simplified "view" or "application" ontology is derived from larger more complex ontologies. In this work we look at a language for specifying view queries over OWL/RDFS sources, and we illustrate some initial ideas for how to execute user queries over our view ontology, without materializing it first.

The vision of the Semantic Web is to create a decentralized network of machine processable OWL or RDF(S) ontologies, much like the WWW is a decentralized network of human readable sources. Reference ontologies, like the University of Washington's Foundational Model of Anatomy (FMA), provide detailed representations of general knowledge domains (anatomy in the case of the FMA). If we wish to include such ontologies in a computational framework like the Semantic Web we would benefit significantly from first reducing them, both in terms of size and complexity, to just what is needed for a given application. Views are one approach to accomplishing this goal.

Like views in SQL, we will define our RDFS/OWL views using a declarative query language. Queries in this discourse are expressed using SparQLeR [1], an extension of SparQL (the W3C recommended RDF(S) query language), which includes support for regular paths. We regard regular paths, including recursive predicates, as necessary constructs of a view language. Queries against the view will be combined with the view query to produce queries over the underlying source ontology(s), insuring up-to-date query results.

The SparQLeR view query V1, center of Figure 1, identifies all classes reachable from *Heart* via paths matching the Kleene closure :regionalPart* (paths containing only *regionalPart* relationships). It then constructs a new graph containing all triples from the source graph whose subject is a regional part of *Heart*, and whose predicate is either :regionalPart or rdfs:subClassOf, but in the former case replaces the predicate with has*_part* from the OBO relation ontology.

Q1 returns all triples whose subject *isa Organ*. If this query were issued against the entire FMA, the results would include triples like:

(esophagus,memberOf, set_of_viscera)
(heart, regionalPart, right_side_of_heart)

Q1 issued against the view V1 would return one of the previous triples, with its predicate replaced:

(heart, obo:has_part, right_side_of_heart)

Q1:

```
CONSTRUCT { $subject $relation $object }
WHERE{
    $subject rdfs:subClassOf  :Organ .
    $subject $relation $object .}
```

V1:

```
PREFIX obo: < http://purl.org/obo/owl/>
CONSTRUCT{ $sub obo:has_part $part .
     $sub rdfs:subClassOf  $superClass .}
WHERE{
    :Heart %p $sub .
    FILTER (regex(%p,":regionalPart*","ds")) .
    $sub :regionalPart $part .
    $sub rdfs:subClassOf $superClass .}
```

Q2 (Q1 + V1) :

```
CONSTRUCT {$sub obo:has_part $part .
     $sub rdfs:subClassOf  $superClass .}
WHERE{
    :Heart %p $sub .
    FILTER (regex(%p,":regional_part*","ds")) .
    $sub :regionalPart $part .
    $sub rdfs:subClassOf $superClass .
    $sub rdfs:subClassOf  :Organ .
    $sub $relation $obj .}
```

Figure 1: Example query (Q1), simple FMA view (V1), and query composition (Q2 = Q1+V1).

To execute Q1 against the view query, V1, without materializing V1's RDF result graph, we compose Q1 and V1 to form a query over the underlying ontology (FMA). Q2 illustrates a composition of Q1 with V1 (note the substitution of $sub and $obj for the $subject and $object variables in Q1). Q2's WHERE clause imposes the combined graph matching constraints of Q1 and V1. The CONSTRUCT clause retains the triple modifications of the view query, unless overridden by Q1.

The advantages of these types of views are 1) they provide a formal mapping to application ontologies from well-structured ontologies, and 2) they are always up-to-date. Remaining challenges include further study of query composition strategies both to enable more complex view queries and to develop efficient methods of query reformulation.

Supported by NIH grant HL087706

[1] Kochut KJ, Maciej J. SPARQLeR: Extended Sparql for Semantic Association Discovery. UGA CS Tech. Report 2006.