Graphical Representation of RDF Queries*

Andreas Harth Sebastian Ryszard Kruk Stefan Decker

Digital Enterprise Research Institute
National University of Ireland, Galway
Galway, Ireland

firstname.lastname@deri.org

ABSTRACT

In this poster we discuss a graphical notation for representing queries for semistructured data. We try to strike a balance between expressiveness of the query language and simplicity and understandability of the graphical notation. We present the primitives of the notation by means of examples.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: [Query formulation]

Keywords

semistructured data, metadata, RDF, query

1. INTRODUCTION

Although the Semantic Web is meant to help machines interpret data on the Web, end-users still need to view and query the data. Ideally, also non-experts should be empowered to formulate queries. State of the art retrieval systems for semistructured data such as [1] that are targeted towards end-users still use keyword-based search interfaces. We believe that there is a need for graphical interfaces that enable non-expert users to formulate complex queries over semistructured data.

Specifying a graphical notation for a query language involves a trade-off between user interface complexity and language expressiveness. At one end of the spectrum are systems such as Magnet [2] and multi-faceted browsing and navigation systems such as Flamenco [3]. In both system one uses so-called facets as restrictions to filter the data set. These systems are limited in that they do not allow object nesting; only filters pertaining to one item can be specified. In addition they need domain specific customizations tailored towards the schema definition of the data set.

On the other end of the spectrum is Query-by-Example (QbE) [4] known from databases. QbE is a complete query language for the relational calculus, can express a wide range of relational queries, and includes insert and update operations. There is no need to customize that language to a specific domain, since the queries are constructed based on the database schema. However, since semistructured data typically comes without a fixed schema, it is not possible to directly apply the QbE paradigm here.

The goal of this poster is to present a graphical notation for queries over semistructured data that identifies a middle ground between the two paradigms which we believe is useful and easy to use – and provides sufficient expressiveness to cover a wide range of queries.

We introduce our graphical notation by means of examples. The exemplary data set is expressed in RDF (Resource Description Framework) using Dublin Core[1] and FOAF[2] vocabularies.

2. PRELIMINARIES

In the following, we introduce the basic ideas for the representation. We assume that the reader has rudimentary knowledge of RDF, a language to represent semistructured data on the Web.

The atomic unit of RDF are RDF triples, which consist of subject, predicate, and object. We use the Notation3 (N3) syntax[3] that introduces variables to the RDF data model to be able to specify queries. The basic notion of an RDF query is an RDF triple pattern. A triple pattern is a triple where subject, predicate, or object can be a variable. An N3 query consists of a `ql:where` clause with one or more triple patterns specifying the selection criteria, and a `ql:select` clause which specifies the format of a query result.

In the following, we define the notion of an RDF facet which is the main element of a graphically constructed query.

Definition 1 (RDF Facet). *Given a set of URI references $\mathcal{U}$, a set of literals $\mathcal{L}$, and a set of variables $\mathcal{V}$, a triple $(s, p, o) \in \mathcal{V} \times \mathcal{U} \times (\mathcal{U} \cup \mathcal{L} \cup \mathcal{V})$ is called an RDF facet.*

A facet can be seen as a filter condition over an RDF graph. Multiple facets can be combined on subject and object positions by using the same variable, which amounts to a join.

3. WALKTHROUGH

In this section, we introduce the different building blocks of our graphical notation by means of examples. We begin each example with a textual description of the query, then

*This work has been partially supported by Science Foundation Ireland (SFI/02/CE1/I131).

WWW 2006, May 23–26, 2006, Edinburgh, Scotland.
ACM 1-59593-332-9/06/0005.

[1] http://dublincore.org/
[2] http://foaf-project.org/
[3] http://www.w3.org/DesignIssues/Notation3.html

show the corresponding graphical representation, and finally present the query expressed in N3 query syntax. In the queries we omit namespace declarations for brevity.

Single Facet. The simplest graphical query consists of just one facet. Please note that doublequotes (“”) mean exact matches on literals.

EXAMPLE 1. *Get resources with the predicate* `foaf:name` *and object* `‘‘Andreas Harth’’`.


```
<> ql:select {
  ?subject1 ?p ?o .
}; ql:where {
  ?subject1 foaf:name ‘‘Andreas Harth’’ .
  ?subject1 ?p ?o .
} .
```

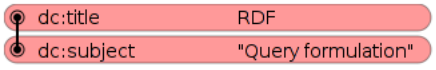
Keyword Search. Exact matches require the user to specify exactly the value of the RDF object, which is not practical. Therefore, we introduce a notion to specify keyword search on an RDF object (if it is a literal). Graphically, we just omit the quotes around the object.

EXAMPLE 2. *Get resources which contain in its* `dc:title` *the keyword* `computer`.


```
<> ql:select {
  ?subject1 ?p ?o .
}; ql:where {
  ?subject1 dc:title ?keyword .
  ?keyword yars:keyword ‘‘computer’’ .
  ?subject1 ?p ?o .
} .
```

Multiple Facets. Specifying only single facets to restrict the result set is not sufficient. Users need facilities to restrict the result set on multiple dimensions. In terms of a query, specifying multiple facets amounts to a join on the subject position.

EXAMPLE 3. *Get resources with keyword* `RDF` *in* `dc:title` *and* `dc:subject` *equals* `‘‘Query formulation’’`.




```
<> ql:select {
  ?subject1 ?p ?o .
}; ql:where {
  ?subject1 dc:title ?keyword .
  ?keyword yars:keyword ‘‘RDF’’ .
  ?subject1 dc:subject ‘‘Query formulation’’ .
  ?subject1 ?p ?o .
} .
```

Specifying Results. So far we have implicitely assumed that the query returns all triples for the subject of the facet. In the following, we show an example that specifies that resources at other positions should be returned, which is denoted using the double circle.

EXAMPLE 4. *Get resources that* `‘‘Andreas Harth’’` *knows.*


```
<> ql:select {
  ?object1 ?p ?o .
}; ql:where {
  ?subject1 foaf:name ‘‘Andreas Harth’’ .
  ?subject1 foaf:knows ?object1 .
  ?object1 ?p ?o .
} .
```

Object Nesting. A powerful feature of RDF is the ability to nest objects, that is, to reference one object from another one. The corresponding notation to specify queries with nested objects is introduced in the following example. In terms of an N3 query, we use this notation to express joins on object and subject of different facets.

EXAMPLE 5. *Get resources that* `‘‘Andreas Harth’’` *knows and that are interested in* `DERI`.


```
<> ql:select {
  ?object1 ?p ?o .
}; ql:where {
  ?subject1 foaf:name ‘‘Andreas Harth’’ .
  ?subject1 foaf:knows ?object1 .
  ?object1 foaf:interest ?keyword .
  ?keyword yars:keyword ‘‘DERI’’ .
  ?object1 ?p ?o .
} .
```

Please observe that the queries return all RDF triples, that is, the language has closure in a sense that results of one query can serve as input of another query. In addition, only dealing with RDF graphs facilitates displaying the results.

4. CONCLUSION AND FUTURE WORK

We have identified a subset of RDF queries that can be expressed in graphical notation. We believe that the notation is simple enough to enable end-users to graphically construct sufficiently complex queries.

We intend to verify our notation by implementing a user interface that allows users to formulate the queries presented in the poster. The graphical representations shown here are encoded in Scalable Vector Graphics (SVG) format. We plan to include in the system client-side scripts to be able to interactively compose queries within a Web browser.

5. REFERENCES

[1] L. Ding, T. Finin, A. Joshi, R. Pan, R. S. Cost, Y. Peng, P. Reddivari, V. C. Doshi, and J. Sachs. Swoogle: A search and metadata engine for the semantic web. In *Proceedings of the Thirteenth ACM Conference on Inf. and Knowledge Mgmt*, 2004.

[2] V. Sinha and D. R. Karger. Magnet: Supporting navigation in semistructured environments. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, June 2005.

[3] K.-P. Yee, K. Swearingen, K. Li, and M. Hearst. Faceted metadata for image search and browsing. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, April 2003.

[4] M. M. Zloof. Query by example – a data base language. *IBM Systems Journal*, 16(4):324 – 343, 1977.