Translating XPath Queries into SPARQL Queries

M. Droop[1], M. Flarer[1], J. Groppe[2], S. Groppe[2], V. Linnemann[2], J. Pinggera[1], F. Santner[1], M. Schier[1], F. Schöpf[1], H. Staffler[1], and S. Zugal[1]

[1] University of Innsbruck, Technikerstrasse 21a, A-6020 Innsbruck, Austria
csae8559@uibk.ac.at, Markus.Flarer@student.uibk.ac.at, {Jakob.Pinggera,Florian.Santner}@student.uibk.ac.at, schiermike@gmx.net, felix.schoepf@gmx.at, {Hannes.Staffler, Stefan.Zugal}@student.uibk.ac.at

[2] IFIS, University of Lübeck, Ratzeburger Allee 160, D-23538 Lübeck, Germany
{jinghua.groppe,groppe,linnemann}@ifis.uni-luebeck.de

Extended Abstract

The W3C has developed XPath [3] as a query language for XML data. XPath is embedded in many other languages like XQuery and XSLT. The name of XPath derives from its basic concept, the path expression, with which the user can hierarchically address the nodes of the XML data. The user of XPath may not only use simple relationships like parent-child, but also more complex relationships like the descendant relationship, which is the transitive closure of the parent-child relationship. Furthermore, complex filter expressions are allowed in XPath queries.

RDF is a language for representing information about resources in the World Wide Web. SPARQL [2] supports querying RDF by triple and optional patterns, con- and disjunctions and extensible value testing.

```
<bookstore>
  <book category="CHILDREN">
    <title>Harry Potter</title>
    <author>J. K. Rowling</author></book>
  <book/></bookstore>
```

Fig. 1. An example XML document

The translation process from XPath [3] queries into SPARQL [2] queries consists of (i) the translation of the input data from XML into RDF, (ii) the translation from the XPath query into the translated SPARQL query, and (iii) the translation of the result into a format, which is equivalent to the result of the XPath query.

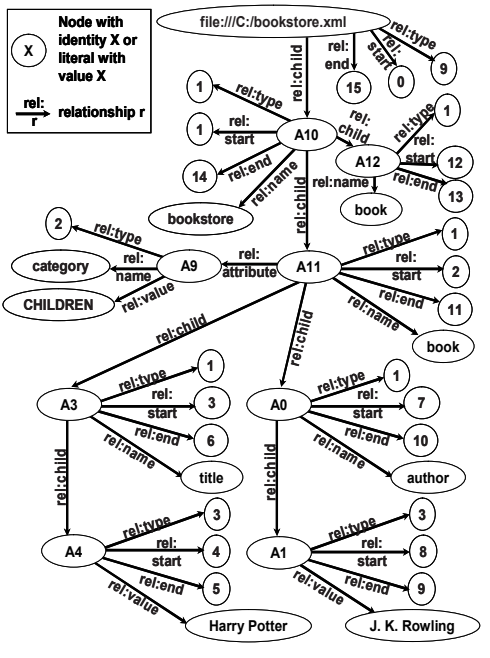

Fig. 2. Transformed RDF data of Fig. 1

R. Meersman, Z. Tari, P. Herrero et al. (Eds.): OTM 2007 Workshops, Part I, LNCS 4805, pp. 9–10, 2007.

We translate the XML data into RDF data by a depth-first traversal of the XML tree and annotate each translated node of the XML data with relationships for annotating the type of the XML node, attribute relationships, namespace relationships, parent-child relationships, its name, its value and two relationships for the usage of a numbering scheme [1] (see Fig. 1 and Fig. 2), as SPARQL queries do not allow the computation of the transitive closure, which is necessary for recursive XPath axes.

With these relationships, we can support all XPath axes in our translation scheme, as we can determine the nodes according to the basic relationships. Note that the XPath location step following::n is equivalent to ancestor-or-self::node()/following-sibling::node()/descendant-or-self::n, and the XPath location step preceding::n is equivalent to ancestor-or-self::node()/preceding-sibling::node()/descendant-or-self::n.

We use standard compiler techniques for the translation of the XPath query into the SPARQL query. See Fig. 3 for the translation of the XPath query /bookstore/parent::node()/descendant::title/text().

```
PREFIX rel: <http://uibk.ac.at/relations/>
PREFIX xsd:<http://www.w3.org/2001/XMLSchema#>
SELECT ?v9 WHERE { ?v0 rel:type "9".
                   ?v0 rel:child ?v1.
                   ?v1 rel:type "1".
                   ?v1 rel:name "bookstore".
                   ?v2 rel:child ?v1.
                   ?v7 rel:start ?v3.
                   ?v2 rel:start ?v5.
                   ?v7 rel:end ?v4.
                   ?v2 rel:end ?v6.
                   ?v7 rel:type "1".
                   ?v7 rel:name "title".
                   ?v7 rel:child ?v8.
                   ?v8 rel:type "3".
                   ?v8 rel:value ?v9.
                   FILTER(xsd:long(?v6)>xsd:long(?v4)).
                   FILTER(xsd:long(?v5)<xsd:long(?v3)).}
```

Fig. 3. Translated SPARQL query of /bookstore/parent::node()/descendant::title/text()

The result of a SPARQL query is a set of bindings of variables in the SELECT clause of a SPARQL query. We determine the translated SPARQL query in such a way that the retrieved bindings of the variables in the SELECT clause of the SPARQL query represent the resultant XML nodes of the original query. In the module of the translation of the result, we rebuild the subtrees of these resultant XML nodes by considering the information of the original XML tree in the RDF data. Furthermore, we sort the resultant XML trees according to the document order of the original XML tree, as the XPath language specifies the result of an XPath query to be in document order of the queried XML document.

We have developed a prototype to verify our translations and to show the practical usability. We have done a performance analysis to measure the execution times of the translations and the evaluations of the XPath query and the translated SPARQL query. The evaluation of translated SPARQL queries from XPath queries not containing a recursive axis is efficient and not significantly slower than processing the original XPath queries.

References

1. Grust, T., van Keulen, M., Teubner, J.: Accelerating XPath evaluation in any RDBMS. ACM Trans. Database Syst. 29, 91–131 (2004)
2. W3C, SPARQL Query Language for RDF, W3C Candidate Recommendation (2007)
3. W3C: XML Path Language (XPath) 2.0, W3C Recommendation (2007)