SPARQL-to-SQL Query Translation: Bottom-Up or Top-Down?

Andrey Kashlev and Artem Chebotko *
Department of Computer Science
University of Texas – Pan American
1201 West University Dr., Edinburg, TX 78539-2999, USA
* Corresponding author. Email: artem@cs.panam.edu

Abstract—Emerging Semantic Web Services rely on the availability of metadata that describes various functional and non-functional characteristics of computational resources. A number of semantic vocabularies and datasets describing existing services and workflows are publicly available and their rapid growth brings forward a new challenge – efficient management of semantic data. Many existing semantic data repositories use conventional relational databases to store and query large RDF datasets. The most complex component of this approach is SPARQL-to-SQL query translation. Existing algorithms translate SPARQL queries to SQL using either bottom-up or top-down strategy and result in semantically equivalent but syntactically different relational queries. While it can be expected that relational query optimizers produce identical query execution plans for semantically equivalent bottom-up and top-down queries, is this usually the case in practice? To address this question, we study bottom-up and top-down translations of SPARQL queries with complex nested optional graph patterns that yield SQL queries with left outer joins whose reordering is not always possible. This paper reports our on-going research and performance study featuring SPARQL queries with nested optional graph patterns over semantic data repositories instantiated in Oracle, DB2, and PostgreSQL.

Keywords—SPARQL; SQL; translation; query; bottom-up; top-down; Semantic Web; RDF; query optimization; query performance

I. INTRODUCTION

Semantic Web technologies are finding more and more applications in solving challenging problems of intelligent data and computing resources search, discovery, sharing, and integration. In the services computing domain [1], the new area of Semantic Web Services [2], [3] has emerged to provide the benefits of automatic service discovery, invocation, composition, and interoperability. A number of semantic vocabularies for services and workflows have been designed, including Semantic Markup for Web Services (OWL-S), Semantic Annotations for WSDL (SAWSDL), Web Service Semantics (WSLD-S), Semantic Web Services Ontology (SWSO), Web Service Modeling Ontology (WSMO), Web Services Discovery Ontology (WSDO), and Open Provenance Model (OPM). The rapid growth of semantic datasets brings forward a new challenge - efficient management of RDF data that is crucial for supporting new semantics-enabled applications.

Many researchers have proposed using conventional relational databases to store and query large Semantic Web datasets [4]. Emerged systems, called relational RDF databases, share a common design pattern that uses a schema mapping algorithm to generate a relational database schema, a data mapping algorithm to insert RDF data into the database, and a query mapping algorithm to translate RDF queries into equivalent SQL queries. SPARQL-to-SQL translation is not only the most complex mapping in a relational RDF database, but also very critical to overall querying performance. Existing algorithms translate SPARQL queries to SQL using either bottom-up or top-down strategy and result in semantically equivalent but syntactically different relational queries.

II. BASIC GRAPH PATTERN TRANSLATION

A basic graph pattern, which is a set of triple patterns, is the main building block of SPARQL queries. While there exist both bottom-up and top-down strategies that generate equivalent SQL queries with nested sub-queries, we explore a simple strategy that generates fully flat SQL queries. Therefore, our research suggests that the order of translation becomes unimportant for basic graph patterns, since only a naive query optimizer does not consider inner join reordering. However, the same is not true for SPARQL nested optional graph patterns as we explain in the following section.

III. BOTTOM-UP AND TOP-DOWN NESTED OPTIONAL GRAPH PATTERN TRANSLATIONS

To illustrate the difference between bottom-up and top-down SPARQL-to-SQL translations in the context of nested optional graph patterns, we use a sample RDF graph G in Figure 1 that describes academic relations among professors and graduate students in a university. We design an RDF query that returns (1) every graduate student in the RDF graph; (2) the student’s advisor if this information is available; and (3) the student’s co-advisor if this information is available and if the student’s advisor has been successfully retrieved in the previous step. The SPARQL representation of this query is as follows:

```
SELECT ?s ?a ?c WHERE{?s type GradStudent./*R1(s)*/
  OPTIONAL {  
    ?s hasAdvisor ?a . /* R2(s,a) */
   OPTIONAL {  
?c hasCoadvisor ?c . /*R3(s, c) */
}}
```

SELECT ?s ?a ?c WHERE{?s type GradStudent./*R1(s)*/
  OPTIONAL {
    ?s hasAdvisor ?a . /* R2(s,a) */
  }
  OPTIONAL {
    ?s hasAdvisor ?a . /* R2(s,a) */
  }
  OPTIONAL {
    ?s hasCoadvisor ?c . /*R3(s, c) */
  }
```
The difference between bottom-up and top-down translation strategies is illustrated in Figure 1, which presents evaluation plans for the bottom-up and top-down SQL queries obtained from our sample SPARQL query. The respective SQL queries are generated using our modified bottom-up and top-down SPARQL-to-SQL query translation algorithms presented in [5] and [6] that are out of this paper scope. Both bottom-up and top-down SQL queries have two left outer joins, however the join order and conditions are different. The evaluation of these queries produces the same resulting relations as shown in the figure.

IV. PERFORMANCE STUDY

Our performance study was conducted using the WordNet dataset and 22 test SPARQL queries that were translated to SQL using the bottom-up and top-down query translation strategies and evaluated in three relational database management systems. The experiments were run on a server with two 2GHz Intel Xeon E5504 Nehalem CPUs, 32GB RAM and 6TB disk array running Ubuntu 9.02 Jaunty x64. Three different database management systems, namely Oracle 10.2 Express Edition, DB2 9.7 Express-C and PostgreSQL 8.3.12, were installed on the server. Generic schema and data mapping algorithms were used to generate identical database schemas in Oracle, DB2 and PostgreSQL, and to store the RDF dataset into the databases, respectively.

The results are reported in Figure 2. If the ratio >1, a top-down query was faster; if ratio <1, a bottom-up query was faster; and if ratio =1, both top-down and bottom-up queries showed the same execution times.

V. CONCLUSIONS

Our study suggested that the choice between bottom-up and top-down translation strategies can have dramatic performance implications on the resulting SQL queries. This choice depends on many factors, including selectivities of triple patterns, their order in a SPARQL query, and even a relational engine that evaluates translated queries.

REFERENCES