An Empirical Study of Real-World SPARQL Queries

Mario Arias Gallego
Computer Science Dept.
Univ of Valladolid, Spain
mario.arias@gmail.com

Javier D. Fernández
Computer Science Dept.
Univ of Valladolid, Spain
jfjergar@infor.uva.es

Miguel A. Martínez-Prieto
Computer Science Dept.
Univ of Valladolid, Spain &
Univ. of Chile, Chile
migumar2@infor.uva.es

Pablo de la Fuente
Computer Science Dept.
Univ of Valladolid, Spain
pfuente@infor.uva.es

ABSTRACT
Understanding how users tailor their SPARQL queries is crucial when designing query evaluation engines or fine-tuning RDF stores with performance in mind. In this paper we analyze 3 million real-world SPARQL queries extracted from logs of the DBPedia and SWDF public endpoints. We aim at finding which are the most used language elements both from syntactical and structural perspectives, paying special attention to triple patterns and joins, since they are indeed some of the most expensive SPARQL operations at evaluation phase. We have determined that most of the queries are simple and include few triple patterns and joins, being Subject-Subject, Subject-Object and Object-Object the most common join types. The graph patterns are usually star-shaped and despite triple pattern chains exist, they are generally short.

Categories and Subject Descriptors
H.2.3 [Database Management]: Languages—Query languages

General Terms
Languages, Measurement

Keywords
SPARQL, usage analysis, RDF store, query evaluation

1. INTRODUCTION
RDF1 provides a simple declarative data model of triples (subject, predicate, object) to describe resources. The number of RDF data sets has increased in diverse areas of application such as bioinformatics, social networks, geographic locations, books or films. The Linked Data Project2 emerges as an initiative to promote the use of RDF to publish structured data on the Web in a distributed and interconnected manner [6]. Latest Linked Open Data (LOD) cloud estimations3 show that more than 25 billion RDF triples are being shared and increasingly interconnected (close to 1 billion links).

SPARQL4 is a declarative language recommended by the W3C for extracting information from RDF graphs. It proposes graph pattern matching facilities to perform searches and data extraction. For instance, it provides the possibility of extracting subgraphs using the CONSTRUCT keyword, or finding certain variable bindings using the SELECT clause. The semantics and complexity of the SPARQL query language have been fairly studied theoretically, showing that SPARQL algebra has the same expressive power as relational algebra [13], although their conversion is not trivial [3, 8].

Several works [12, 4] explore efficient SPARQL evaluation methods based on query evaluation optimization [9]. Some heuristics include triple pattern reordering based on selectivity estimation [16], dynamically restricting triple patterns [9], RISC-style query processing [12] and optimization based on “star shaped groups” [17], i.e., different triple patterns around one or few common variables. Some techniques focus on minimizing the processing time of joins. In [1], subject-subject joins are assumed to be very frequent operations, and they can be carried out in linear time (w.r.t. the size of the tables). Multi-way joins can also be performed instead of multiple individual joins [11]. RDF store benchmarking has also conjectured about SPARQL special features to provide sets of representative queries [7, 14].

To the best of our knowledge, there are no specific works analyzing SPARQL query logs, and few works tackle semantic searches [2]. However, there are several works regarding the Web [10, 15] that can provide some insights on how to face the three major steps of log analysis [10]: (i) data recollection, (ii) preparation, and (iii) analysis.

We argue that the look and feel of SPARQL queries has been only conjectured based on common sense or on the analysis of few synthetic SPARQL queries taken from benchmarks. However, there is no compelling scientific foundation to assess whether those assumptions are satisfied on real

---

1http://www.w3.org/TR/REC-rdf-syntax/

2http://linkeddata.org

3http://www4.wiwiss.fu-berlin.de/lodcloud/

4http://www.w3.org/TR/rdf-sparql-query/
data and real queries. The main objective of this work is analyzing real-world SPARQL queries to depict what kind of accesses the users perform, focusing on those clauses that are more expensive in terms of query evaluation, both from planification and index access points of view. This study is data-set-independent, in the sense that anyone could follow our methodology to analyze any SPARQL log, querying any RDF data set, supported by any RDF store or query engine. We consider that our study may assist the designers of indices, stores, optimizers and benchmarks in making reasonable assumptions and taking plausible decisions.

In Section 2, we first describe the properties of the logs and the preprocessing steps. Then, we provide a comprehensive analysis of the clauses and structure of the queries. Finally, Section 3 summarizes our conclusions and future work.

2. SPARQL LOG ANALYSIS

We use the logs from the USEWOD2011 Challenge, kindly provided by the organisation. They consist of several months of usage data (server logs) from DBPedia (about general knowledge) and Semantic Web Dog Food (about authors and publications). Since we have two sources of information, we analyze them aside and then compare their results. Both query sets are statistically relevant due to the amount and heterogeneity of the users originating them.

We first extract the queries from the HTTP log to their textual representation, obtaining about 5 million queries for DBPedia and 2 million for SWDF. Then we parse each query using Jena and extract all relevant features using a tool we specifically designed for this task. Users tend to repeat some queries, so lots of them are mere duplicates. Since this fact might bias our results, we exclude from our study all the identical queries generated from the same host, and all those that do not comply with the SPARQL grammar specification and result in parsing errors. Table 1 summarizes some statistics for the original and resultant data sets.

We first investigate which are the most common types of SPARQL queries. The most frequent one is SELECT, comprising the 96.9% and 99.7% of DBPedia and SWDF queries respectively. It is more surprising that ASK (1.6%/0.2%), CONSTRUCT (1.5%/0.01%) and DESCRIBE (0.002%/0.002%) are scarcely used.

Table 1: DBPedia and SWDF query log statistics.

<table>
<thead>
<tr>
<th></th>
<th>DBPedia</th>
<th>SWDF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Queries</td>
<td>5,166,272</td>
<td>2,092,508</td>
</tr>
<tr>
<td>Duplicates from same host</td>
<td>51.7%</td>
<td>69.8%</td>
</tr>
<tr>
<td>Parse error</td>
<td>4.37%</td>
<td>1.03%</td>
</tr>
<tr>
<td>Analyzed</td>
<td>44.9%</td>
<td>29.1%</td>
</tr>
</tbody>
</table>

Then, we estimate the frequency of appearance of the different SPARQL features (figure 1). FILTER is the most used one by almost 49% in both sources. This is a relevant result considering that some query planification algorithms [17] push filters to be executed first, thus, the space of RDF triples to be explored is considerably reduced. LANG is the most used function, it occurs in the 28% of the total DBPedia filters. However, it was not mentioned in SWDF, which is quite obvious considering that DBPedia publishes contents in many languages and SWDF only contains English literals. The following most-used function in DBPedia is the equal comparator (23%) which holds the first position in SWDF (93%). A further analysis of the filter expressions reveals that 99.4% of the filters only affect one variable. These facts envisage that adequate indices would enhance access operations on the many queries including filters.

We also analyze the usage of DISTINCT, which ensures that no duplicates are returned. We observe that it is more popular on DBPedia than SWDF, perhaps because of the complexity of its schema. We also study the appearance of SELECT REDUCED that lets the SPARQL engine remove duplicates if possible, but not mandatory. We discover that only an insignificant amount of two queries did use this method, therefore RDF store designers should not rely on users using this modifier. The FROM feature is widely used on DBPedia, which is composed by several data sources, but it is not used on SWDF since it only has one big graph and this would be redundant. We also note that there is a lack of usage of the features ORDER BY, GRAPH, FROM NAMED and OFFSET which occurred less than 0.5% in our tests.

Table 2: Triple Patterns (C=Constant, V=Variable).

<table>
<thead>
<tr>
<th>Pattern</th>
<th>DBPedia</th>
<th>SWDF</th>
</tr>
</thead>
<tbody>
<tr>
<td>C V V</td>
<td>66.35%</td>
<td>47.79%</td>
</tr>
<tr>
<td>C V C</td>
<td>21.56%</td>
<td>9.32%</td>
</tr>
<tr>
<td>V C C</td>
<td>7.06%</td>
<td>46.08%</td>
</tr>
<tr>
<td>C C C</td>
<td>1.46%</td>
<td>4.21%</td>
</tr>
<tr>
<td>C C C</td>
<td>1.01%</td>
<td>0.001%</td>
</tr>
<tr>
<td>V C V</td>
<td>0.34%</td>
<td>0.19%</td>
</tr>
<tr>
<td>V C C</td>
<td>0.20%</td>
<td>0.006%</td>
</tr>
<tr>
<td>V V V</td>
<td>0.04%</td>
<td>1.18%</td>
</tr>
</tbody>
</table>

Figure 1: Percentage of queries using the different SPARQL features at least once.

Some authors [12] state that the most used feature in SPARQL is conjunction. While this statement holds true, the amount of disjunctions (UNION) is not small enough to be overlooked whatsoever, since it appears in 11.84% of DBPedia queries. We also find that there is a significant amount of OPTIONAL blocks. This result is critical, because some studies proved that the optional operator from the SPARQL algebra is the major culprit of the query evaluation being PSPACE-complete [13].

Once we have shown a basic insight on the usage of single elements, we proceed to perform a higher level analysis on the structure of the query expressions. SPARQL provides a means to match graph patterns by specifying several triple...
The cornerstone of efficient SPARQL evaluation is optimizing the planification of join order execution. Thus, we need to count the number of joins appearing on each query and their types. We can define the join operation as a conjunction of two triple patterns, where both have one variable in common. This leads to six types of joins depending on which position the common variable appears in each pattern: Subject-Subject, Predicate-Predicate, Object-Object, Subject-Predicate, Subject-Object and Predicate-Object. The SPARQL specification does not determine in which order the joins shall be performed since the result is equivalent due to the commutative property. It is the task of the query evaluation engine to decide the final order of the processing. Given a single query, there is not a unique way of taking the groups, hence, the count of join types varies. In figure 3 we can see an example of the different join possibilities among 5 graph patterns and one variable ?V, being C non-relevant constants. In this case one of the joins is redundant, and depending on which one we leave out, the join count for each type will differ.

We propose counting joins by first grouping the same type ones (SS, PP, OO) and then those of different type (SP, SO, PO). This is a simple and consistent method of evaluating join types regardless of the query evaluation engine. Figure 4 shows the results of our join counting method applied to the log data set. We noticed that 2.66% of the total queries in DBPedia have a single join, 0.75% have two joins, and this percentage gradually decreases with a maximum of 10 joins in a query. In all, 4.25% of the queries have at least one join (see figure 1). It is also remarkable that the most common type of join is SS, followed by SO and OO.

Another interesting hypothesis posed on previous works (and assumed as true) is that SPARQL graph patterns are typically star-shaped or include long-chains [12]. We propose an analysis to provide empirical evidence to check whether
Table 3: Pattern graph out degree serialization.

<table>
<thead>
<tr>
<th>Pattern</th>
<th>DBPedia</th>
<th>SWDF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 0</td>
<td>56.51%</td>
<td>97.46%</td>
</tr>
<tr>
<td>2 0 0</td>
<td>3.77%</td>
<td>1.02%</td>
</tr>
<tr>
<td>3 0 0</td>
<td>1.37%</td>
<td>0.48%</td>
</tr>
<tr>
<td>4 0 0 0</td>
<td>0.10%</td>
<td>0.01%</td>
</tr>
<tr>
<td>5 0 0 0 0</td>
<td>0.03%</td>
<td>0.00%</td>
</tr>
<tr>
<td>6 0 0 0 0 0</td>
<td>0.01%</td>
<td>0.00%</td>
</tr>
<tr>
<td>&lt;Others&gt;</td>
<td>0.01%</td>
<td>0.14%</td>
</tr>
</tbody>
</table>

these two statements are valid. To do so we construct a directed graph for each query (similar to the one used in figure 3) and calculate its longest path. We discovered that 98% of both DBPedia and SWDF queries had a length of just 1, 1.8% had 2, and very few queries had up to 5 jumps. Thus, there conclude that graph pattern do include chains, but at least in our data sets they are very scarce.

In order to characterize whether pattern graphs have star shapes, we propose using a serialization of the out-degree of each node of the graph in decreasing order. Star-shaped graphs will have a central node with a high out-degree, and several leaf nodes with null out-degree (For instance: 3 0 0 0). Then we count the frequency of each degree pattern as shown in table 3. We see that the most common is the most simple one with one single triple, which might be considered a trivial star and chain. If we keep browsing the rest degree patterns, we observe that there is a big proportion of appearances (more than the 99.93%) of almost-star-shaped graph structures between 3 and 9 nodes.

3. CONCLUSIONS AND FUTURE WORK

We noticed that previous works made assumptions about SPARQL queries that were not successfully supported scientifically. Hence we carried out a study of real-world SPARQL queries in order to check out these hypothesis. We expect our results to be valuable to RDF store designers, specially in the tasks of query evaluation planification and index construction.

We conclude that most queries are simple, i.e., 66.41% of DBPedia queries and 97.25% of SWDF just contain a single triple pattern. However, there are many examples of queries including expensive SPARQL operations, like UNION, OPTIONAL and joins. We also detected that most queries (99.97%) have a star-shaped graph pattern, and the chains in 98% of the queries have length one, with the longest path having a length of five. The percent of queries using join ranges from 2.19% to 4.25%, and they are typically of the types SS(~ 60%), SO(~ 35%) and OO(~ 4.5%).

We consider that our study is fairly representative, since we have analyzed a complex and large RDF data set such as DBPedia, and then we have contrasted those results with SWDF. In future works, we plan to apply the tool we have developed to other query logs from different domains of application such as biological data.

4. ACKNOWLEDGMENTS

Partially funded by the MICINN (TIN2009-14009-C02-02) and the Millennium Institute for Cell Dynamics and Biotechnology (ICDB) (Grant ICM P05-001-F). The second author is granted by a fellowship from Erasmus Mundus, the Regional Government of Castilla y Leon (Spain) and the European Social Fund.

5. REFERENCES


