Pattern-based analysis of SPARQL queries from the LSQ dataset*

Timo Stegemann and Jürgen Ziegler

University of Duisburg-Essen, Duisburg, Germany,
timo.stegemann@uni-due.de,
http://interactivesystems.info

Abstract. This paper presents a pattern-based analysis of the Linked SPARQL Queries dataset (LSQ). The analysis showed that from more than 630,000 unique SELECT queries stored in the dataset, 99% of them are represented by only 120 different query patterns.

1 Introduction

In this paper, we present an analysis of the Linked SPARQL Queries dataset (LSQ), collected by Saleem et al. [2]. The LSQ dataset has already been evaluated in statistical ways by the authors themselves as well as others (eg. [1]). They investigated, among other things, the usage of specific SPARQL features (UNION, DISTINCT, FILTER, etc.) or different forms of joins (star, path, sink, etc.). In contrast to these evaluations, we analysed the patterns that were used when constructing the queries. The results of our analysis might be relevant in the field of teaching, for developers of Linked Data applications that support users in the query writing process, or for providers of public SPARQL endpoints.

2 Analysis of the Linked SPARQL Query Dataset

The Linked SPARQL Queries dataset (LSQ) contains nearly 1.75 million unique queries (date: July 2017) with a total of approximately 5.68 million query executions from four different public SPARQL endpoints[1]. From this dataset we extracted 636,876 unique SELECT queries with 1,526,804 executions[2] that did not produce any parse errors and returned a valid result at the time of their logging[3]. These SELECT queries represent 91.0% of all valid queries (ASK 4.5%,

* The poster is accompanying our ISWC'17 research track paper [3].

[1] DBpedia, LinkedGeoData (LGD), Semantic Web Dog Food (SWDF), British Museum

[2] Our number of queries differ slightly from the ones that Saleem et al. received during their analysis. We additionally parsed all queries with functions from the Apache Jena framework. During this procedure some queries that were marked in the dataset as correct were sorted out because of parsing errors.

[3] Queries from the British Museum have been completely filtered out, since it was not recorded if they returned a result. Furthermore, all requests from the dataset were made by a single agent and match the same simple query pattern.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT DISTINCT *
  WHERE { ?value rdfs:label 'Semantic Web'@en . }
LIMIT 10
```

⇓

```
SELECT ?v0 WHERE { ?v0 <i0> 'l0'@lang }
```

Fig. 1: Exemplary transformation of a SPARQL query into a parameterized form.

	Overall	DBpedia	LGD	SWDF
0.50	3	2	1	5
0.75	6	3	3	13
0.85	12	6	5	21
0.90	21	8	9	30
0.95	42	14	17	44
0.99	120	52	36	62

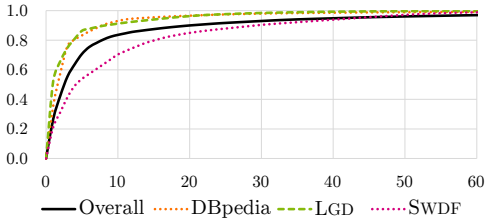

Fig. 2: Fitted cumulative Pareto distribution of the most frequently used query patterns overall and for each endpoint, showing how many query patterns are at least required to represent a specific fraction of the executed queries.

DESCRIBE 3.8%, CONSTRUCT 0.7%). While Saleem et al. analyzed the dataset as a whole, we set our focus on the used query patterns.

To extract query patterns from the individual queries, we transformed the queries in the test set into a parameterized form. In the first step, we removed all parts from the query string that have no impact on the pattern itself, such as PREFIX, FROM, LIMIT, OFFSET, and ORDER BY, as well as the DISTINCT keyword and corresponding parenthesis. In the next step, we mapped all IRIs, variables, literals and language tags of each query to a generic format (IRIs: `<i#>`, variables: `?v#`, literals: `'l#'`, and language tags: `@lang`, where `#` is the index of its first occurrence in the query). We replaced all wildcards in SELECT statements with the corresponding list of variables from the WHERE statement and harmonized language filter expressions. Fig. 1 shows an example transformation of a SPARQL query into parameterized form. In the last step we merged all identically parameterized queries into a single query pattern.

Through this, we obtained 1619 unique query patterns where the first 120 patterns of the most frequently executed queries represent 99% of the queries executed overall. The first 42 represent 95%, the first 21 represent 90%, and the first 3 already represent more than 50% of the queries executed overall (see Fig. 2). Complete results are available online[4].

A separate analysis for each endpoint in the data set resulted in 1240 unique query pattern for the DBpedia endpoint, 289 for LGD, and 151 for SWDF. Only 17 patterns appear in all three endpoints (but represent 24.8% off all executions), 27

[4] `https://semwidg.org/page/download-research#LSQ_Analysis`

	Overall	DBpedia	LGD	SWDF
1	0.267	0.357	0.525	0.212
2	0.159	0.221	0.134	0.102
3	0.129	0.183	0.091	0.102
4	0.081	0.042	0.060	0.076
5	0.069	0.028	0.054	0.046
6	0.046	0.027	0.013	0.033
⋮	⋮	⋮	⋮	⋮
12	0.001	0.005	0.006	0.021

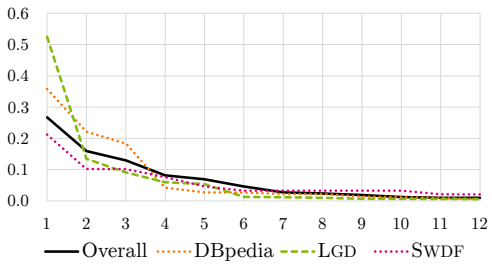

Fig. 3: Frequency distribution of the most frequently used query patterns.

appear in two different endpoints (9.5%), and 1575 appear in only one endpoint (65.6%). The Pareto distribution of the query patterns for the DBpedia and LGD endpoint are similar, although it is notable that the most frequently used pattern in the LGD endpoint already represents more than every second executed query. For the SWDF endpoint, the patterns are more evenly distributed (see Fig. 3), but it should be noted that 16.5% of the executed queries in SWDF context are probably made by a PHP library[5] that tests the feature support of an endpoint by using a set of very characteristic queries.

The 12 most frequently used query patterns of the overall LSQ dataset, representing more than 85% of all executed queries, are listed in Table 1 together with their ranking of the separate analysis for the different SPARQL endpoints. Each additional pattern represents less than 0.1% of the queries executed overall (see Fig. 3).

Most of these patterns are simple subject-predicate-object relations with one or two variables, that are also part of the SELECT statement, on varying positions. Four of them (patterns (a), (b), (f) and (i)) are using OPTIONAL pattern matching. FILTER expressions are only used in very simple ways. In pattern (g) the subject variable is restricted to a set of specific IRIs. In pattern (d) and (i) a FILTER expression is used to restrict the result set to a specified language. In pattern (k) this is done by a language tag. Pattern (k) is also the only one of these patterns that makes use of a literal value. Patterns (b) and (k) have more than one triple pattern sharing the same subject. Pattern (i) is the only complex one of these patterns. Additionally to the FILTER expression and the OPTIONAL pattern matching it contains a path spreading over two triple patterns and a triple pattern that takes a predicate from a previous triple pattern as subject.

In the overall data set, language filters (in form of FILTER expressions or language tags) are used in 14.8% of all executed queries. FILTER expressions that do not restrict the language are used in 5.7%, UNIONS in 7.5%, and GROUP BY expressions in 0.5%. Aggregate functions are used in 1.2%, whereas most of them are COUNT expressions (0.9%). Other SPARQL features like subqueries, BIND, or HAVING are merely used in negligibly small numbers.

[5] `http://graphite.ecs.soton.ac.uk/sparqllib/`

Table 1: Ranking of the 12 most frequently used query patterns of the LSQ dataset and their positions for the single endpoints. These queries cover more than 85% of all requests.

	Overall	DBpedia	LGD	SWDF	Query Pattern
(a)	1	-	1	-	`SELECT ?v0 WHERE { OPTIONAL { <i0> <i1> ?v0 } }`
(b)	2	1	-	-	`SELECT ?v0 ?v1 ?v2 WHERE { <i0> <i1> ?v0 ↪ OPTIONAL { <i0> <i2> ?v1 ; <i3> ?v2 } }`
(c)	3	2	4	54	`SELECT ?v0 WHERE { <i0> <i1> ?v0 }`
(d)	4	3	128	-	`SELECT ?v0 WHERE { <i0> <i1> ?v0 FILTER langMatches( lang(?v0), 'l0' ) }`
(e)	5	31	2	20	`SELECT ?v0 ?v1 WHERE { ?v0 <i0> ?v1 }`
(f)	6	-	3	-	`SELECT ?v0 WHERE { OPTIONAL { ?v0 <i0> <i1> } }`
(g)	7	-	5	-	`SELECT ?v0 ?v1 WHERE { ?v0 <i0> ?v1 FILTER ↪ (?v0 = <i1> || ?v0 = <i2> || ?v0 = <i3> || ?v0 = <i4> || ?v0 = <i5>) }`
(h)	8	5	19	1	`SELECT ?v0 ?v1 WHERE { <i0> ?v0 ?v1 }`
(i)	9	4	-	-	`SELECT ?v3 ?v0 ?v1 ?v2 WHERE { <i0> ?v0 ?v1 OPTIONAL { ?v1 <i1> ?v2 } ↪ OPTIONAL { ?v0 <i1> ?v3 } FILTER ((langMatches(lang(?v3), 'l0') ↪ || !langMatches(lang(?v3), '*')) && (langMatches(lang(?v1), 'l0') ↪ || !langMatches(lang(?v1), '*')) && (langMatches(lang(?v2), 'l0') ↪ || !langMatches(lang(?v2), '*'))) }`
(j)	10	6	50	54	`SELECT ?v0 WHERE { ?v0 <i0> <i1> }`
(k)	11	7	80	-	`SELECT ?v0 ?v1 WHERE { ?v0 <i0> 'l0'@lang ; <i1> ?v1 }`
(l)	12	8	-	-	`SELECT ?v0 ?v1 WHERE { ?v0 ?v1 <i0> }`

3 Conclusion

In summary our analysis showed that the most frequently executed queries in the LSQ dataset are rather simple and represented by only few different query patterns. It is common to request several properties of a specific resource in a single query and optionally filter them by their language. Other filter expressions are rare. It should be noted that the analyzed queries are not necessarily completely representative, since the query logs are provided by only three public SPARQL endpoints.

References

1. Han, X., Feng, Z., Zhang, X., Wang, X., Rao, G., Jiang, S.: On the statistical analysis of practical sparql queries. In: Proceedings of the 19th International Workshop on Web and Databases. p. 2. ACM (2016)
2. Saleem, M., Ali, M.I., Hogan, A., Mehmood, Q., Ngomo, A.C.N.: The Semantic Web - ISWC 2015: 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part II, chap. LSQ: The Linked SPARQL Queries Dataset, pp. 261–269. Springer International Publishing, Cham (2015)
3. Stegemann, T., Ziegler, J.: Investigating learnability, user performance, and preferences of the path query language SemwidgQL compared to SPARQL. In: The Semantic Web - ISWC 2017: 16th International Semantic Web Conference (to appear)