Deducing Federated SPARQL queries from RDF Mappings

Benjamin Moreau[1], Manoé Kieffer[2], and Patricia Serrano-Alvarado[2]

[1] OpenDataSoft {Name.Lastname}@opendatasoft.com
[2] Nantes University, LS2N, CNRS, UMR6004, 44000 Nantes, France
{Name.LastName@}univ-nantes.fr *

Abstract. Datasets can be virtually integrated into the Linked Data space through *RDF mappings*. An RDF mapping consists of rules that map data from an input dataset to RDF triples. Defining SPARQL queries is an arduous task. As RDF mappings define semantic schemas, it is possible to use them to deduce federated queries. In this demonstration, we propose MaRQ, a tool that deduces federated queries from a set of RDF mappings. The goal is to provide a clear idea of the conjunction possibilities of a dataset with other virtually integrated datasets.

1 Introduction and motivation

Having data as linked data enables vast amounts of datasets to be interconnected creating new and innovative applications. To avoid expensive investments in terms of storage and time, some data providers use RDF mappings to integrate virtually and on-demand, non-RDF data into the Linked Data [6, 7]. Making mappings is not easy, but some tools help to generate them [2, 8]. For data providers, it would be interesting to know how a particular dataset might benefit from existing semantic datasets. That is, *to which extent, their dataset can be combined with datasets of the Linked Data, i.e., which conjunctive queries could be processed by a federation that includes their dataset?*

In the state of the art, [1] allows to generate federated queries based on RDF datasets. This approach provides flexible parameterization of realistic conjunctive benchmark queries. Query parameterization includes structure, complexity, and cardinality constraints. Similarly, [9] can generate a variety of federated queries over a given set of RDF datasets to facilitate the process of benchmarking for federated query processing. Generated queries are conjunctive and use the OPTIONAL and FILTER keywords. [3] proposes a solution for generating federated queries from query logs executed in the past. Generated queries are conjunctive queries and queries using UNION, OPTIONAL, FILTER, etc.

This work provides a low-cost solution that uses RDF mappings instead of RDF datasets or query logs. We consider, that the particular dataset that a

provider wants to integrate into the Linked Data virtually is not materialized in the RDF format, and a log of executed SPARQL queries does not exist.

Our tool, named MaRQ, analyses RDF mappings to deduced conjunctive BGPs. An RDF mapping can be seen as an RDF summary of the dataset it describes. Thus, its analysis can provide helpful information with limited overhead. MaRQ identifies the conjunction capabilities between a particular dataset and a set of datasets of the Linked Data that are virtually integrated through RDF mappings. It provides a ranked list of RDF datasets according to their degree of conjunction with a particular dataset.

2 MaRQ: a tool to detect joins from RDF mappings

Consider Figure 1 that shows two mappings. Mapping 1 concerns a doctors' directory dataset. Mapping 2 concerns an Airbnb accommodations dataset. In bold, we highlight templates or references existing in the RDF mappings. We want to know how corresponding datasets could enrich one another. We recall that to deduce federated SPARQL queries, MaRQ uses nothing but RDF mappings. So deduced queries are not verified against instances of corresponding datasets.

MaRQ deduces queries of type *subject-subject*, *object-object*, and *subject-object*. The matching of terms is based on the pairwise Jaccard similarity of the types that describe them.

In MaRQ, the Jaccard similarity is the number of types in common divided by the total number of types of two subjects (templates) described in two mappings. Two terms from different mappings are *joinable* if their similarity is greater than a configurable threshold between 0 and 1. For instance, the Jaccard similarity between Address and Home is 0.20, i.e., 1/5. This is because these templates have only one type in common (schema:Place), and the total number of types of both templates is 5. If the threshold of MaRQ is equal to or greater than 0.2, then a join between these templates will be deduced.

Subject-subject queries. To identify two joinable subjects, MaRQ calculates the similarity of all subjects. Considering a similarity threshold ≥ 0.2, in our example, Doctor is similar to Host, and Address is similar to Home and Neighbourhood. Thus MaRQ deduces three queries. Each query contains one join BGP. BGPs will contain all types (rdf:type and corresponding classes) and all predicates with the same variable in subjects and, for not rdf:type predicates, different variables in objects. Listing 1.1 shows one of these queries. It asks for Doctors that are also BusinessEntities.

Object-object queries. To identify two joinable objects, MaRQ uses objects that are described semantically in the mappings. In our example, three objects are described: Address, Host and Neighbourhood. If there is a pairwise similarity a query is deduced. Only Address is similar to Neighbourhood. BGP will contain different variables in all subjects, the same variable in the object, and the

predicates where the objects are used. Listing 1.2 shows this query. It asks for Doctors' locations that are neighbourhood of Airbnb homes.

It is possible also to deduce object-object queries only based on types. In our example, MaRQ deduces two queries of this type, one with the BGP {?s1 rdf:type schema:Person} and another with the BGP {?s1 rdf:type schema:Place}. These queries ask for all persons and all places of both datasets.

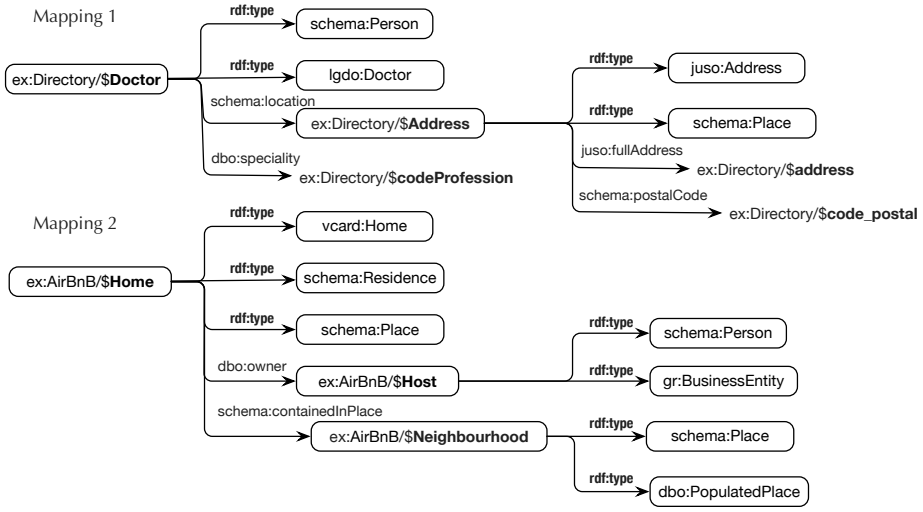

Fig. 1. Two RDF mappings.

```
SELECT * WHERE {
 ?s1 rdf:type schema:Person .
 ?s1 rdf:type lgdo:Doctor .
 ?s1 rdf:type gr:BusinessEntity .
 ?s1 schema:location ?o1 .
 ?s1 dbo:speciality ?o2}
```

Listing 1.1. A subject-subject query.

```
SELECT * WHERE {
 ?s1 schema:location ?o1 .
 ?s2 schema:containedInPlace ?o1}
```

Listing 1.2. An object-object query.

```
SELECT * WHERE {
 ?t1 rdf:type schema:Person .
 ?t1 rdf:type lgdo:Doctor .
 ?t1 schema:location ?f1 .
 ?t1 dbo:speciality ?f2 .
 ?f3 dbo:owner ?t1}
```

Listing 1.3. A subject-object query.

```
SELECT * WHERE {
 ?t2 rdf:type schema:Place .
 ?t2 rdf:type dbo:PopulatedPlace .
 ?f1 schema:location ?t2}
```

Listing 1.4. An object-subject query.

Subject-object queries. To identify subject-object joins, MaRQ identifies objects that are described semantically in the second mapping. In our example, objects Host and Neighbourhood are described in the second mapping. Then, it identifies subjects in the first mapping that are joinable with these objects. In our example, the subject Doctor is similar to the object Host, and the subject Address is similar to the object Neighbourhood. Thus, two subject-object queries are deduced. BGPs will contain all types (rdf:type and corresponding classes)

and properties of the joinable subject with the same variable in the subject, and the predicates where the joinable object is used with the same subject' variable. Listing 1.3 shows one of these queries. It asks for doctors that are also owners of Airbnb homes.

Deduction of *object-subject queries* follows the algorithm of subject-object ones, but mappings are taken in inverse order. In our example, the object Address is similar to the subjects Home and Neighbourhood. Thus, two more queries are deduced, and they are considered subject-object queries. Listing 1.4 shows one of these queries. It asks for Airbnb places that are also doctors' locations.

3 Demonstration

The MaRQ implementation uses YARRRML mappings [4]. You can test it in command line `https://github.com/Manoe-K/MaRQ`. It is also available as a Web application available at `https://marq-priloo.univ-nantes.fr/`.

During the demonstration, attendees will be able to choose some RDF mappings. They will then be able to choose one mapping in order to compare it to all the others. A graph representing the number of triple patterns that MaRQ generates by query type will be shown for each mapping. The similarity threshold used by the Web application is 0.2 (this threshold is configurable in the command line version). Deduced queries will be shown, and they are also downloadable.

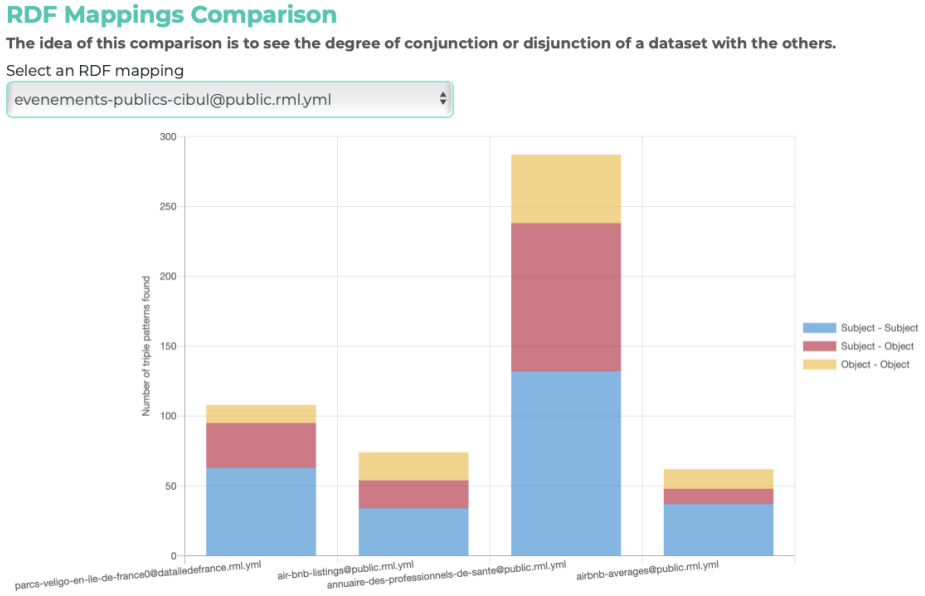

Fig. 2. MaRQ screenshot.

Figure 2 shows a screenshot of the MaRQ web application. In this example, the dataset evenements-publics-cibul is compared other three datasets. The

graph shows the number of potential join queries and their types (subject-subject, subject-object, object-subject).

Final remarks. MaRQ identifies possible conjunctive queries from RDF mappings. The pertinence of deduced queries depends on the quality of mappings and the similarity threshold. In addition, as each BGP contains all the possible triple patterns of joinable terms, it is unlikely that these queries, with such a big number of constraints, give results. However, they provide a valuable set of BGPs whose triple patterns can be analyzed to identify joins that may return instances.

In addition, join queries to integrate different datasets face the problem of entity-matching. For instance, URIs can be of different domains but referring to the same entity. This problem is out of the scope of this work, but several solutions exist, see [5] for a survey.

Acknowledgment. Authors thank Fatim Touré (Master student of the University of Nantes) for her participation in the early stages of this work.

References

1. Görlitz, O., Thimm, M., Staab, S.: SPLODGE: Systematic Generation of SPARQL Benchmark Queries for Linked Open Data. In: International Semantic Web Conference (ISWC) (2012)
2. Gupta, S., Szekely, P., Knoblock, C.A., Goel, A., Taheriyan, M., Muslea, M.: Karma: A System for Mapping Structured Sources into the Semantic Web. In: Extended Semantic Web Conference (ESWC), Poster&Demo (2012)
3. Hacques, F., Skaf-Molli, H., Molli, P., Hassad, S.E.: PFed: Recommending Plausible Federated SPARQL Queries. In: International Conference on Database and Expert Systems Applications (DEXA) (2019)
4. Heyvaert, P., De Meester, B., Dimou, A., Verborgh, R.: Declarative Rules for Linked Data Generation at Your Fingertips! In: Extended Semantic Web Conference (ESWC), Poster&Demo (2018)
5. Köpcke, H., Rahm, E.: Frameworks for entity matching: A comparison. Data & Knowledge Engineering **69**(2), 197–210 (2010)
6. Michel, F., Faron Zucker, C., Montagnat, J.: A Mapping-based Method to Query MongoDB Documents with SPARQL. In: International Conference on Database and Expert Systems Applications (DEXA) (Sep 2016), `https://hal.archives-ouvertes.fr/hal-01330146`
7. Moreau, B., Serrano Alvarado, P., Desmontils, E., Thoumas, D.: Querying non-RDF Datasets using Triple Patterns. In: International Semantic Web Conference (ISWC), Poster&Demo (Oct 2017)
8. Moreau, B., Terpolilli, N., Serrano-Alvarado, P.: A Semi-Automatic Tool for Linked Data Integration. In: International Semantic Web Conference (ISWC), Poster&Demo (Oct 2019)
9. Rakhmawati, N.A., Saleem, M., Lalithsena, S., Decker, S.: QFed: Query Set for Federated SPARQL Query Benchmark. In: International Conference on Information Integration and Web-based Applications & Services (2014)