SPORTAL: Searching for Public SPARQL Endpoints

Ali Hasnain[1], Qaiser Mehmood[1], Syeda Sana e Zainab[1], and Aidan Hogan[2]

[1] INSIGHT Centre for Data Analytics, National University of Ireland, Galway
[2] Center for Semantic Web Research, DCC, University of Chile

Abstract. There are hundreds of SPARQL endpoints on the Web, but finding an endpoint relevant to a client's needs is difficult: each endpoint acts like a black box, often without a description of its content. Herein we briefly describe SPORTAL: a system that collects meta-data about the content of endpoints and collects them into a central catalogue over which clients can search. SPORTAL sends queries to individual endpoints offline to learn about their content, generating a best-effort VoID description for each endpoint. These descriptions can then be searched and queried over by clients in the SPORTAL user interface, for example, to find endpoints that contain instances of a given class, or triples with a given predicate, or more complex requests such as endpoints with at least 1,000 images of people. Herein we give a brief overview of SPORTAL, its design and functionality, and the features that shall be demoed at the conference.

1   Introduction

Finding public SPARQL endpoints that contain content relevant for a client's needs is not easy. Say, for example, a client is interested in data about movies and wants to find related SPARQL endpoints on the Web. They could try to use a traditional search engine with keywords like "`movie sparql`" or something similar, but many of the webpages returned may not actually be SPARQL endpoints. Another option would be to use a specialised service such as the VoID Store[3], which allows for performing searches over dataset descriptions provided by publishers that sometimes include a link to a SPARQL endpoint (or even multiple endpoints); however, the system relies on publishers creating their own VoID files [1], keeping them up to date, etc. A better option might be to use the Datahub catalogue[4] to find movie datasets with a SPARQL endpoint, but about half of the endpoints listed in Datahub are no longer working [2].

Rather than relying on publisher-submitted meta-data about the content of endpoints – which may be out of date and is in any case not available for the majority of endpoints [2,4] – we instead propose a system, which we call the SPARQL PORTAL (SPORTAL), to run SPARQL queries against a given list

---

[3] `http://void.rkbexplorer.com/`; (l.a. 2016-08-29)
[4] `http://datahub.io/`; (l.a. 2016-08-29)

of endpoints to compute meta-data about their content, with a particular emphasis on schema data. The meta-data that SPORTAL collects is based on the computable subset of VoID [1,3] and some further extensions thereof.[5]

Computing dataset descriptions for endpoints in this way has a number of advantages: (1) SPORTAL can be updated on demand by rerunning queries against the endpoints (we currently update every 15 days), hence excluding offline endpoints and reflecting changes to content; (2) we assume no external descriptions or services other than a working SPARQL (1.1) endpoint; (3) the provenance of the descriptions we compute are given by the queries we use, and the time we run them against the endpoint. However, likewise, there are a number of disadvantages: (1) the queries needed to generate a detailed dataset description can be expensive, and may fail due to performance limitations or result-size thresholds of public endpoints [2], thus SPORTAL will have incomplete descriptions for many operational endpoints; (2) most of the queries require support for SPARQL 1.1, which although growing, is not yet universal [2].

In our previous work [3], we introduced SPORTAL, where we (1) proposed a set of what we call "self-descriptive queries" that can be used to generate a dataset description from an endpoint with incremental expressivity/complexity, (2) evaluated the feasibility of running these queries in a local setting for four SPARQL implementations (4Store, Fuseki, Sesame, Virtuoso) over four different datasets, (3) performed experiments for a list of 618 public SPARQL endpoints collected from Datahub and Bio2RDF to see how well these queries performed in real-world settings and how much data we could collect for the central SPORTAL catalogue, and, (4) gave an overview of the online SPORTAL system that allows clients to search and/or query the catalogue of dataset descriptions.

Our additional contribution will be to demo the SPORTAL system, which is available online at `http://www.sportalproject.org/`. Herein, we first describe the process of data collection, recapitulating some of the main results from our previous study [3] (Section 2). Thereafter, we focus on the functionality of the SPORTAL system itself, which will be demoed at the conference (Section 3).

2   SPORTAL Data Collection

To each public SPARQL endpoint being catalogued, SPORTAL sends a sequence of queries of increasing complexity to learn about the content of that endpoint. In particular, SPORTAL uses `CONSTRUCT` queries that will directly generate VoID meta-data from the endpoint. For example, we send the following query[6] to the endpoint to generate meta-data about VoID class partitions [1]:

```
CONSTRUCT { <D> void:classPartition [ void:class ?c ] } WHERE { ?s a ?c }
```

Here, `<D>` represents an IRI for the dataset that we generate internally based on the endpoint URL. Many of the queries we use require SPARQL 1.1 features, in

---

[5] `http://ldf.fi/void-ext#`; (l.a. 2016-08-29)
[6] Prefixes used can be located at `http://prefix.cc/` (l.a. 2016-08-29)

particular aggregation and sub-queries; e.g., the following query uses aggregation and a sub-query to count triples in each property partition [1]:

```
CONSTRUCT { <D> void:propertyPartition [ void:property ?p ; void:triples ?x ] }
WHERE { SELECT (COUNT(?o) AS ?x) ?p WHERE { ?s ?p ?o } GROUP BY ?p }
```

The queries become increasingly complex, where, for example, the following query counts, for each class, the number of instances that have each property:

```
CONSTRUCT { <D> void:classPartition [ void:class ?c ;
                void:propertyPartition [ void:distinctSubjects ?x ] ] }
WHERE { SELECT (COUNT(DISTINCT ?s) AS ?x) ?c ?p WHERE { ?s a ?c ; ?p ?o } GROUP BY ?c ?p }
```

Each endpoint answers – or attempts to answer – each such query over its local dataset; when merged, the results for each query comprise a description of the dataset. We consider 29 such CONSTRUCT queries in total [3].

These queries – in particular the latter more complex ones – would be expensive to compute, particularly over large datasets. Hence in our previous work [3], we performed a variety of experiments to ascertain how feasible it would be to answer these queries over current SPARQL implementations and public endpoints. The most reliable implementation appeared to be Virtuoso, which managed to successfully run 27/29 queries on datasets of around 1 million triples, but could only run 8/29 queries over a subset of DBpedia with 114.5 million triples, 53,200 unique predicates and 447 unique classes.

In experiments over 618 endpoints taken from the DataHub and Bio2RDF [3], 307 (49.7%) responded to a simple SPARQL 1.0 query (i.e., were operational) and 168 (27.2%) responded to a simple SPARQL 1.1 query (i.e., support SPARQL 1.1). Considering just these 307 operational endpoints, non-empty/non-error responses to our queries varied from 94% for the first query above listing classes, to about 25% for the latter query. We did not verify the completeness nor the correctness of answers; we could only report that non-empty results were returned.

We refresh the data collected from the DataHub/Bio2RDF endpoints every 15 days. We refer the reader to our previous work [3] for more details on queries, runtimes, result sizes, and so forth.

3 SPORTAL Interfaces

Over the data collected by running these dataset-description queries on endpoints, we build a number of interfaces to help clients find endpoints of interest.

SPARQL Interface: All collected meta-data are indexed in a public SPARQL endpoint that clients can query programmatically. Just as an example, we can find the largest five datasets/endpoints with instances of foaf:Person:

```
SELECT DISTINCT ?ep ?ts
WHERE {
 ?ds void:sparqlEndpoint ?ep ;
  void:triples ?ts ; void:classPartition
   [ void:class foaf:Person ] .
}
ORDER BY DESC(?ts) LIMIT 5
```

?ep	?ts
http://commons.dbpedia.org/sparql	1229690546
http://live.dbpedia.org/sparql	563358498
http://lod.kaist.ac.kr/sparql	326078469
http://data.oceandrilling.org/sparql	284665595
http://data.utpl.edu.ec/.../lod/sparql	215627469

A variety of queries are supported per the data we collect. One can also ask, e.g., for the most frequent classes/properties across all endpoints, endpoints with the most instances of a given class, endpoints that have images of people, etc. The SPARQL endpoint (with a YASGUI interface [5]) is available at the following location `http://www.sportalproject.org/yasgui/yasgui.html`.

User Interface: The two main features of the U.I. we provide are class and property search, where a user can enter a substring such as `person`/`knows` to autocomplete a list of class/property URLs (in descending order of the number of endpoints using the term) and then search that class/property URL to find endpoints with data using that class/property (in descending order of the number of instances/triples using that term, where available). We also offer endpoint search based on autocompleting URL substrings (e.g., `dbpedia`), showing the meta-data we have for that endpoint. We also have some views of statistics, including success rates of queries, distributions of property/class terms, etc. The U.I. front-page is available at `http://www.sportalproject.org/`.

4   Conclusions

Though by the nature of the data collection process, the SPORTAL catalogue is incomplete – e.g., results for the previous example query may miss endpoints with `foaf:Person` instances that failed to return a triple count – the system does offer useful (partial) results when looking for relevant public endpoints in a manner that, to the best of our knowledge, no existing service does.

We will demonstrate both the SPARQL interface and the user interface at the conference, showing examples of queries and searches that can be executed, discussing possible use-cases for SPORTAL (e.g., SPARQL federation, finding datasets to link to, etc.), as well as possible future directions for SPORTAL and alternative strategies for finding relevant endpoints.

Aside from improvements to the interface, possible future plans for the tool include discovery of new endpoints and integration with SPARQLES [2].

References

1. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing Linked Datasets. In: Linked Data On the Web (LDOW). CEUR (2009)
2. Buil-Aranda, C., Hogan, A., Umbrich, J., Vandenbussche, P.Y.: SPARQL Web-Querying Infrastructure: Ready for Action? In: ISWC, pp. 277–293. Springer (2013)
3. Hasnain, A., Mehmood, Q., e Zainab, S.S., Hogan, A.: SPORTAL: Profiling the Content of Public SPARQL Endpoints. IJSWIS 12(3), 134–163 (2016)
4. Paulheim, H., Hertling, S.: Discoverability of SPARQL Endpoints in Linked Open Data. In: ISWC Posters & Demos. pp. 245–248. Springer (2013)
5. Rietveld, L., Hoekstra, R.: YASGUI: feeling the pulse of Linked Data. In: EKAW. pp. 441–452 (2014)