LODatio: A Schema-Based Retrieval System for Linked Open Data at Web-Scale

Thomas Gottron[1], Ansgar Scherp[2,1], Bastian Krayer[1], and Arne Peters[1]

[1] Institute for Web Science and Technologies, University of Koblenz, Germany
[2] Data and Web Science, University of Mannheim, Germany
{gottron,scherp,bastiankrayer,arne}@uni-koblenz.de

Abstract The Linked Open Data (LOD) cloud has grown to an enormous source for semantic data. Its distributed and decentralized approach is one reason for its success but also poses challenges. A main difficulty is to identify those data sources on the LOD cloud which provide the information a user is actually interested in. With LODatio, we have developed a prototype retrieval system to support users in finding the right data sources for a given schema-oriented information need. Beyond classical search system functions such as retrieval, ranking, result set size estimation and providing result snippets, LODatio provides sophisticated support for the users in refining and expanding their information need.

1 Introduction

The aim of Linked Open Data (LOD) [4] is to publish and interlink open data of different quality and of different purpose on the web using the Resource Description Framework (RDF). This data forms a huge web-scale RDF graph—the so-called LOD cloud—which represents an enormous source of structured information. Since its advent in 2007, LOD as gained widespread popularity and is supported by non-profit organizations such as libraries as well as major industries. It lies in the nature of the data to be distributed across several de-centralized data sources. As on the classical web of documents, there is no central instance managing or indexing all the information. With the growing number of data sources and the tremendous increase of pure amount of data published, it becomes more and more important to identify sources in the LOD cloud which provide information satisfying schema-oriented information needs. Schema-oriented information needs comprise the search for certain types of resources or resources having certain properties. Existing Semantic Web search engines such as Swoogle [2], SemSearch [7], Sig.ma [11], and Sindice [8] typically focus on the search for specific resources, though. They usually allow for a keyword-based querying approach of traditional search or to enter an URI to start browsing. Thus, schema-oriented information needs as indicated above are not supported by these systems.

In order to provide for a system addressing schema-oriented information needs, we have designed the retrieval system LODatio. LODatio allows for identifying those LOD data sources that provide RDF resources satisfying a schema-oriented information need specified via a SPARQL query. The advantage of using SPARQL as query language is that it allows for a precise definition of the schema-oriented constraints on the data

P. Cimiano et al. (Eds.): ESWC 2013, LNCS 7955, pp. 142–146, 2013.

and that the same query can subsequently be used to query the relevant data sources. However, for future versions of LODatio we intend to implement an additional interface for non-expert users allowing for the formulation of keyword-based queries which will then be translated to SPARQL automatically [9]. LODatio uses a schema-based index [6] to identify for given SPARQL query patterns the relevant sources for LOD. The index contains meta data about these data sources such as their URI, the number of compliant resources and a limited number of examples. Making use of this meta data in an nifty way, we implemented several services to actively support the user in his information seeking task. These services entail relevance based ranking of data sources, estimation of the overall number of relevant resources, provision of example data in the form of result snippets, as well as user support for refining or generalizing the query. The latter two services support typical search tactics to overcome situations in which the user finds too few or too many results [1]. Inspired by web search engines, we refer to this services as *Did you mean?* and *Related queries*.

The motivation for all of these features are the positive effects similar features have had on classical web retrieval. Query related result snippets, for instance, allow users to interact faster and more successful with a retrieval system [10]. Query recommendations, instead, have been found to be accepted frequently and to help the users in overcoming problems in formulating their information need [5]. In general, such active user support features have been called for and proposed as a research agenda for retrieval systems [1].

The live demo of the LODatio prototype (cf. Figure 1) is available online at `http://lodatio.west.uni-koblenz.de:8080`. It is accessible via a web interface allowing for end user access to the system. Example queries provide an easy entrance point to the system and allow for a quick formulation of a first query.

2 Main Features of LODatio

Motivated by successful features of modern web search engines, we have implemented a range of services that form the backbone of the LODatio system:

Ranked Retrieval. Based on the lookup capabilities of the underlying index, we can identify relevant data sources. To provide a ranking, we use a naive ranking function which presents data sources in decreasing order of the number of complying resources they provide.

Result Snippets. For each relevant data source, we can provide example resources and their labels. This allows the user to get a first impression of the data he might encounter at a listed data source and, thereby, supports him in the decision whether or not the data is relevant to his information need.

Result Set Size Estimation. For a given SPARQL query, we can estimate the total amount of matching resources on the entire LOD cloud. This is achieved by aggregating the number of complying resources over all relevant data sources. In addition to supporting the end users in judging the amount of suitable data, the result set estimation plays an important role in the next two services.

Did You Mean? For queries that provide only few or even no results, the user might want to generalize his information need. LODatio supports this task by taking the original SPARQL query and iteratively removing or relaxing its constraints provided in the

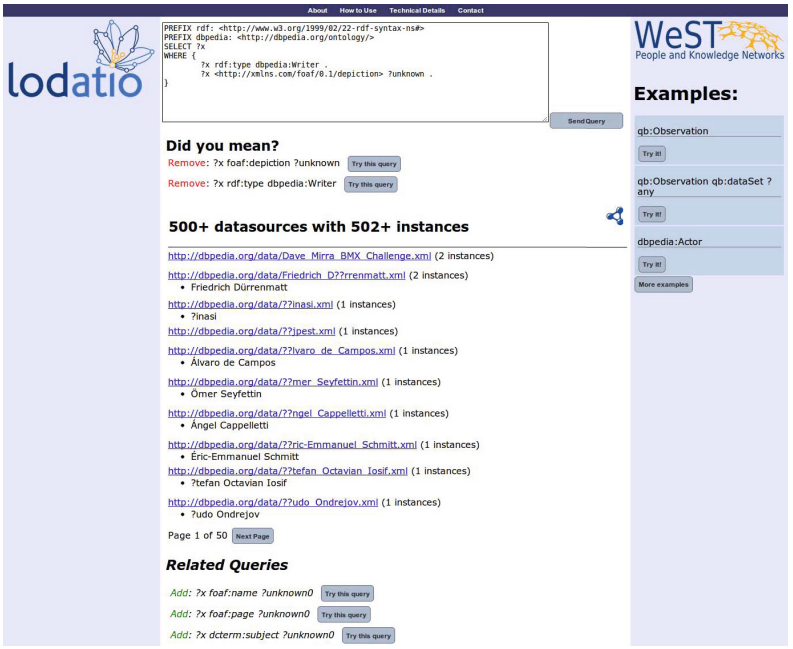

Fig. 1. Web frontend of the LODatio prototype demo

form of query patterns. The resulting new, generalized query candidates are analysed for the estimated size of their result set. Finally, the user receives suggestions from this set of generalized query candidates that extend the current result set in a modest way.

Related Queries. As complement to the *Did you mean?* function, we provide a service which recommends more specific queries. Here, we make use of the schema index and look up query patterns to extend the original information need of the users. This allows for a directed extension of the constraints in a way that does not lead to empty result sets. The quality of the extension is again measured by the size of the obtained new result set.

3 The LODatio Prototype Demo

In LODatio, the users state their information need as an initial SPARQL query. This query is entered in the box at the top of the prototype as depicted in Figure 1. As we do not assume that users in general are capable or willing to write SPARQL queries, we are well aware that a future extension of LODatio is to add a feature for automatically mapping keyword-based input to a SPARQL query, e.g. following the approach of [9]. For the sake of allowing users an easy access to the LODatio system, we currently provide illustrative example queries at the right side of the query box.

User Interaction. After specifying their information need as SPARQL query, the users submit this initial query to LODatio by pressing the “Send Query” button. Subsequently, the different features of LODatio as described in Section 2 are used to fill the

result screen as depicted in the screenshot shown in Figure 1. The first three features of *ranked retrieval*, *result snippets*, and *result set size estimation* are used to provide the content of the middle section of the user interface presenting the result list of Linked Data sources relevant to the query. The ranked result list is paginated, as the size of the set of relevant data sources can easily become very large. The users can navigate through the pages by clicking on the "Next Page" button as shown in Figure 1 (the "Previous Page" button is not shown as the screenshot depicts the first result page). For each entry in the result list, the users can click on the URI of the data sources to directly inspect the information that is behind a specific data source. Furthermore, up to three example resources are used as illustrative *result snippets* for each relevant data source. Above of the page-based result list is an overview of the overall number of relevant data sources found as well as an estimation of the number of instances in these data sources.

Besides the navigable page-based result list, the users can also modify the query expressing their information need in order to either broaden or narrow down the result set. These features are shown in terms of *Did you mean?* and *Related Queries* query suggestions at the top and the bottom of the result list, respectively. The user can select one of these query suggestion by pressing the "Try this query" button next to the depiction of the query modification. In Figure 1, the *Did you mean?* query suggestions are to remove the property foaf:depiction or the RDF type dbpediaowl:Writer from the current query. Regarding the *Related Queries*, LODatio has retrieved from the underlying schema information the query suggestions to add the property foaf:name, foaf:page (for the homepage of a person), or dcterms:subject.

Implementation. We have implemented all of the above services and integrated them into the live LODatio prototype. As data background, we use a schema index computed on the data set provided for the Billion Triple Challenge 2012[1]. The data set represents a 17GB crawl of a part of the LOD cloud and contains nearly 1.5 billion RDF triple. A real-time computation of the LODatio services can be achieved on a single core machine. As data structure, we use in LODatio a schema-based index over linked data [6]. In this index, we encode schema related information of LOD in a look up data structure. The elements in this schema index correspond to RDF resources that satisfy certain types of SPARQL query patterns. The schema can precisely answer queries addressing (i) resources of a given type or set of types, (ii) resources having certain properties, or even (iii) resources of certain type that are linked to other resources of a given type via a specific set of properties. While more complex queries cannot be answered exactly they can be answered in an approximative way by combining the available patterns (i) to (iii). Looking up the information stored in the index allows for a fast retrieval of meta data of the RDF resources complying with the specified query patterns. This payload is composed of the URIs of relevant data sources, example resources and their labels as well as count information of how many matching resources can be found at this data source. The meta data in the payload of the schema-based index is at the core of the implementation of all the features discussed above. The most essential information is certainly the count information as it feeds also into the *Did you mean?* and *Related Queries* features.

[1] `http://challenge.semanticweb.org/2012`

Technically, the index is implemented in a hybrid architecture [3]. We use a triple store for the schema level of the index and store the payload, instead, in a relational database. This allows for a compact representation of the flexible schema information while leveraging RDBMS technology for looking up meta data information following a consistent schema format.

4 Summary

Our advanced retrieval system LODatio for identifying relevant data sources on the LOD cloud leverages a schema index to respond to a user's information need specified as a SPARQL query. The system provides core retrieval functionality such as ranked result lists, result snippets, and estimates of the total result set size. Furthermore, it supports two typical tactics of users when seeking information: generalizing and refining the information need. As future work we plan to evaluate the benefits on the user's side in extensive user studies.

Acknowledgements. The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 257859, ROBUST.

References

1. Bates, M.J.: Where should the person stop and the information search interface start? Information Processing and Management 26(5), 575–591 (1990)
2. Ding, L., Finin, T.W., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V., Sachs, J.: Swoogle: a search and metadata engine for the semantic web. In: CIKM. ACM (2004)
3. Gottron, T., Scherp, A., Krayer, B., Peters, A.: LODatio: Using a Schema-Based Index to Support Users in Finding Relevant Sources of Linked Data. In: K-CAP 2013: Proceedings of the Conference on Knowledge Capture (2013)
4. Heath, T., Bizer, C.: Linked Data: Evolving the Web Into a Global Data Space. Synthesis Lectures on the Semantic Web. Morgan & Claypool (2011)
5. Kelly, D., Cushing, A., Dostert, M., Niu, X., Gyllstrom, K.: Effects of popularity and quality on the usage of query suggestions during information search. In: Conference on Human Factors in Computing Systems (2010)
6. Konrath, M., Gottron, T., Staab, S., Scherp, A.: Schemex–efficient construction of a data catalogue by stream-based indexing of linked data. Web Semantics: Science, Services and Agents on the World Wide Web 16, 52–58 (2012)
7. Lei, Y., Uren, V.S., Motta, E.: Semsearch: A search engine for the semantic web. In: Staab, S., Svátek, V. (eds.) EKAW 2006. LNCS (LNAI), vol. 4248, pp. 238–245. Springer, Heidelberg (2006)
8. Oren, E., Delbru, R., Catasta, M., Cyganiak, R., Stenzhorn, H., Tummarello, G.: Sindice.com: a document-oriented lookup index for open linked data. IJMSO 3(1), 37–52 (2008)
9. Shekarpour, S., Auer, S., Ngomo, A.C.N., Gerber, D., Hellmann, S., Stadler, C.: Keyword-driven sparql query generation leveraging background knowledge. In: Web Intelligence, pp. 203–210 (2011)
10. Tombros, A., Sanderson, M.: Advantages of query biased summaries in information retrieval. In: SIGIR 1998, pp. 2–10 (1998)
11. Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., Decker, S.: Sig.ma: Live views on the web of data. J. Web Sem. 8(4), 355–364 (2010)