Storing and Provisioning Linked Data as a Service

Johannes Lorey

Hasso Plattner Institute,
Prof.-Dr.-Helmert-Str. 2-3, 14482 Potsdam, Germany
johannes.lorey@hpi.uni-potsdam.de

Abstract. Linked Data offers novel opportunities for aggregating information about a wide range of topics and for a multitude of applications. While the technical specifications of Linked Data have been a major research undertaking for the last decade, there is still a lack of real-world data and applications exploiting this data. Partly, this is due to the fact that datasets remain isolated from one another and their integration is a non-trivial task. In this work, we argue for a Data-as-a-Service approach combining both warehousing and query federation to discover and consume Linked Data. We compare our work to state-of-the-art approaches for discovering, integrating, and consuming Linked Data. Moreover, we illustrate a number of challenges when combining warehousing with federation features, and highlight key aspects of our research.

1 Introduction and Motivation

In recent years, the semantic and technical foundations of Linked Data have been widely studied and formalized. Well-known technologies, such as the Resource Description Framework (RDF) model for representing structured linked information or the SPARQL Protocol and RDF Query Language (SPARQL) for retrieving this information, have been established. Whereas in principle Linked Data allows leveraging information from multiple different providers quite easily, real-world applications oftentimes only rely on a single dataset or a handful of RDF data sources. The challenges for utilizing Linked Data include:

- Data Discovery, i.e., finding suitable resources for an information need,
- Data Integration, i.e., merging multiple data sources to allow homogeneous access to the combined information contained in all sources, and
- Data Consumption, i.e., retrieving and processing relevant data items.

For developers, finding suitable Linked Data sources for their application need can be cumbersome. While there exist a number of metadata models for describing the contents of Linked Data sources, such information is provided for few datasets, in different formats, and becomes outdated quickly. Moreover, using public SPARQL endpoints to query actual data is tedious: Typical infrastructures are not designed for large-scale access by multiple users. On the other hand, materializing data locally poses new challenges, such as updating issues.

P. Cimiano et al. (Eds.): ESWC 2013, LNCS 7882, pp. 666–670, 2013.
© Springer-Verlag Berlin Heidelberg 2013

Therefore, we propose a Linked-Data-as-a-Service [10] approach combining aspects of both data warehousing and distributed query processing. We use a scalable infrastructure allowing users to create private SPARQL endpoints comprising datasets relevant for their application and to incorporate a federation of public SPARQL endpoints. We address a number of optimization issues for managing locally maintained data as well as remotely retrieved information, including metadata generation and semantic caching strategies.

2 State of the Art and Open Challenges

Data Discovery. There exist a number of projects to assist interaction with individual or sets of Linked Data sources, such as SIG.MA[1], RKB Explorer[2], or the Information Workbench[3]. Their focus mostly lies on visualizing and analyzing information provided during set-up time, while support for discovering, adding, and updating resources during run-time is limited. Typically, these tools are designed to allow information exploration and analysis in combination with a certain degree of UI customization. Whereas these features allow for straightforward interpretation of the contained information, the tools might be insufficient for application developers to further process and extend the knowledge base.

A more general approach to discover suitable knowledge for an application is by analyzing metadata of the available datasets. Usually, RDF is used to present information about Linked Data sources and their contents themselves. Whereas this common model allows easy access to metainformation, specific details may differ due to the variety of vocabularies available to describe Linked Data sources, either semantically (e.g., using Dublin Core[4]) or structurally (e.g., using VoID[5]). Certain Linked Data catalogues, such as the Data Hub[6], provide even further metadata, while many datasets are published without any such information.

Data Integration. In the Web of Data, there exist different techniques to integrate multiple data sources. Typically, ontologies and metadata are utilized for Linked Data integration [5,8]. Here, it is assumed that information from different sources can be mapped to and aligned with one another using common vocabulary elements. In the case of a distributed setting, queries are usually rewritten based on this ontological information and issued against a federation of suitable SPARQL endpoints. Other approaches for Linked Data integration rely on expressive rules to match entities or concepts from different data sources [9].

While the foundations of these techniques have been long established for relational data management and adapted for the use of Linked Data, they rely on a number of requirements which are not always satisfied in real-world scenarios.

[1] `http://sig.ma`
[2] `http://www.rkbexplorer.com`
[3] `http://iwb.fluidops.com`
[4] `http://dublincore.org/documents/dces/`
[5] `http://www.w3.org/TR/void/`
[6] `http://datahub.io/group/lodcloud/`

Consider the case of ontology-based data integration: Some methods assume a common ontology, whereas in a distributed setting individual data sources may use different ontologies. Thus, the problem of data integration becomes an issue of ontology mapping [2]. Moreover, even within a single cross-domain ontology, such as the one provided for DBpedia[7], there is often no clear-cut semantic differentiation between individual concepts or between their properties, leading to ambiguity and challenges in data integration. Additionally, as with Linked Data itself, ontologies are subject to revision, thus causing further problems when integrating datasets adhering to different releases of the same vocabulary.

Data Consumption. Linked Data is usually consumed using one of two setups, either by accessing a central repository containing one or more RDF datasets, or by querying publicly available SPARQL endpoints. Given an adequate hard- and software infrastructure, the first method enables high-performance access to the data at hand. However, maintaining the warehoused data requires sophisticated approaches for index creation, compression, and updating [2]. Gathering information by querying (a federation of) public endpoints alleviates some of these challenges, but may degrade execution performance. Typically, such optimization issues are addressed by different federated query processing techniques.

As with data integration, most research in retrieving RDF Data by federated query processing is based on related work in distributed relational data management. However, Linked Data exhibits several novel features that enable new possibilities for query execution against a federation of data sources. First and foremost, as entities in the Web of Data are identified by unique, dereferencable, and connected URIs [3], relevant resources can be iteratively determined during query evaluation. Again, ontologies can be used to aid this process by homogenizing the schemas of different sources. However, many challenges in real-world applications settings influence the success of distributed query processing, including latency and bandwidth restrictions [2], or reduced endpoint availability.

3 Proposed Approach and Previous Work

As both data warehousing and distributed query processing offer a number of benefits, we propose a hybrid approach to store and provision Linked Data for application developers. Using a Cloud infrastructure, our framework allows for scalable processing of large-scale RDF dumps. In addition, a mediator-based architecture [7] enables ad-hoc integration of new data sources by continuously materializing SPARQL query results. We plan on maintaining a lightweight metadata catalogue that is iteratively extended and updated with information generated within our architecture and gathered from outside sources. Leveraging this metadata collection, users can deploy customized SPARQL endpoints suitable to their information and scalability needs, which are then populated using publicly available data dumps and results retrieved from SPARQL endpoints.

[7] `http://dbpedia.org/ontology/`

In previous work, we focused on metadata generation [4] and ontology reconciliation [1]. These steps are necessary prerequisites for our hybrid framework where data from different sources may be added ad-hoc to a deployed endpoint and the metadata catalogue itself. Currently, we are analyzing real-world SPARQL query logs to identify typical human and machine agent query sequences. Our goal here is to deduce suitable caching strategies to store frequently accessed data. Moreover, we want to identify resources related to requested data and store this information for subsequent queries. Additionally, we hope that automating this process can assist in determining conceptual gaps between different datasets and ontologies by comparing user behavior.

We have implemented a prototype of our framework using the infrastructure provided by Amazon Web Services[8]. In particular, we use a customized virtual machine template to deploy SPARQL endpoint instances that are consequently populated with openly available RDF dumps and results retrieved while executing SPARQL queries. Monitoring the load received by the endpoint allows to scale the resources available to the system, e.g., by increasing the allocated memory. Moreover, we plan on combining the results derived from our query log analysis with run-time information to establish cost-efficient retainment policies.

4 Methodology and Ongoing Research

We have discovered several issues that hinder a more widespread utilization of Linked Data by application developers. While there exist individual approaches to deal with these challenges, they lack applicability and refinement when considering real-world scenarios. Furthermore, there have been only few attempts to combine these solutions to boost the accessibility of Linked Data for developers. In the remainder of this PhD project, our focus lies on the following components:

Materialized View Selection and Management. Materializing results retrieved from SPARQL endpoints for optimized query execution in subsequent requests can improve the responsiveness and usability of Linked Data applications. Whereas selecting adequate data to retain for future access is essential in this process, establishing proper strategies for updating and discarding this information are also difficult challenges [6]. Our focus here lies on view selection for dynamic query workloads, where a shift in access pattern frequency is reflected in the retention strategy for the corresponding resource.

Query Analysis and Expansion. Instead of simply caching received results potentially useful for future requests, it might prove beneficial to rewrite incoming queries to retrieve more information than actually requested by the user. In turn, subsequent queries may be evaluated more efficiently by using this additional data. Similar techniques have been used in information retrieval to expand keyword queries and determine related resources in explorative search settings. We are currently evaluating the quality of different strategies for prefetching Linked Data, e.g., by using ontological and structural information.

[8] `http://aws.amazon.com/`

Data Integration and Ontology Mapping. A core concept of our proposed framework combining both centralized data storage and distributed query processing is data integration. As discussed earlier, this has been as widely-studied research area. Given the use-case scenario of our work, we focus on continuous data integration by leveraging existing ontology mapping techniques. Here, we hope to contribute new results by exploiting the heterogeneous characteristics of different request patterns and corresponding results.

Scalable Data Processing. Whereas public SPARQL endpoints exhibit serious availability and accessibility problems, for example limited bandwidth or long periods of downtime, most Cloud-based platforms offer certain service quality guarantees in the form of service-level agreements. As developers are potentially already deploying their application stack in such an environment, hosting the data there as well may tremendously improve query execution time while leveraging multi-tenancy to reduce the operational expenditure of such a set-up. Using our Cloud-based framework, we hope to verify and exploit these advantages.

References

1. Abedjan, Z., Lorey, J., Naumann, F.: Reconciling ontologies and the web of data. In: Proceedings of the International Conference on Information and Knowledge Management (CIKM), Maui, HI, USA, pp. 1532–1536 (October 2012)
2. Betz, H., Gropengießer, F., Hose, K., Sattler, K.U.: Learning from the history of distributed query processing - a heretic view on linked data management. In: Proceedings of the International Workshop on Consuming Linked Data, COLD (November 2012)
3. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. International Journal on Semantic Web and Information Systems 5(3), 1–22 (2009)
4. Böhm, C., Lorey, J., Naumann, F.: Creating voiD descriptions for web-scale data. Journal of Web Semantics 9(3), 339–345 (2011)
5. Correndo, G., Salvadores, M., Millard, I., Glaser, H., Shadbolt, N.: SPARQL query rewriting for implementing data integration over linked data. In: Proceedings of the International Conference on Extending Database Technology (EDBT), Lausanne, Switzerland, pp. 4:1–4:11 (March 2010)
6. Goasdoué, F., Karanasos, K., Leblay, J., Manolescu, I.: View selection in semantic web databases. Proceedings of the VLDB Endowment 5(2), 97–108 (2011)
7. Langegger, A., Wöß, W., Blöchl, M.: A semantic web middleware for virtual data integration on the web. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 493–507. Springer, Heidelberg (2008)
8. Makris, K., Gioldasis, N., Bikakis, N., Christodoulakis, S.: Ontology mapping and SPARQL rewriting for querying federated RDF data sources. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM 2010, Part II. LNCS, vol. 6427, pp. 1108–1117. Springer, Heidelberg (2010)
9. Schenk, S., Staab, S.: Networked graphs: a declarative mechanism for SPARQL rules, SPARQL views and RDF data integration on the web. In: Proceedings of the International World Wide Web Conference (WWW), New York, NY, USA, pp. 585–594 (April 2008)
10. Wang, L., von Laszewski, G., Younge, A., He, X., Kunze, M., Tao, J., Fu, C.: Cloud computing: a perspective study. New Generation Computing 28, 137–146 (2010)