Semantic Web Technologies and Big Data Infrastructures: SPARQL Federated Querying of Heterogeneous Big Data Stores

Stasinos Konstantopoulos[1], Angelos Charalambidis[1], Giannis Mouchakis[1], Antonis Troumpoukis[1], Jürgen Jakobitch[2], and Vangelis Karkaletsis[1]

[1] Institute and Informatics and Telecommunications, NCSR 'Demokritos'
Aghia Paraskevi 15310, Athens, Greece
{konstant,acharal,gmouchakis,antru,vangelis}@iit.demokritos.gr
[2] Semantic Web Company, Vienna, Austria
j.jakobitsch@semantic-web.at

Abstract. The ability to cross-link large scale data with each other and with structured Semantic Web data, and the ability to uniformly process Semantic Web and other data adds value to both the Semantic Web and to the Big Data community. This paper presents work in progress towards integrating Big Data infrastructures with Semantic Web technologies, allowing for the cross-linking and uniform retrieval of data stored in both Big Data infrastructures and Semantic Web data. The technical challenges involved in achieving this, pertain to both data and system interoperability: we need a way to make the semantics of Big Data explicit so that they can interlink and we need a way to make it transparent for the client applications to query federations of such heterogeneous systems. The paper presents an extension of the Semagrow federated SPARQL query processor that is able to seamlessly federated SPARQL endpoints, Cassandra databases, and Solr databases, and discusses future directions of this line of work.

Keywords: Federated query processing; SPARQL; Big Data infrastructures.

1 Introduction and Motivation

Scalable, efficient, and robust data services are re-shaping the way that data analysis techniques are applied to the heterogeneous data cloud. Modern, distributed data storage and processing solutions offer unprecedented levels of scalability, robustness, fault tolerance and elasticity. They are, however, almost all outside the scope of Semantic Web technologies, as data values typically lack explicit semantics and cross-links.

The ability to cross-link large scale data with each other and with structured Semantic Web data, and the ability to uniformly process Semantic Web and other data adds value to both the Semantic Web and to the Big Data community; extending the scope of the former to include a vast data domain and increasing the opportunities for the latter to process data in novel ways and combinations. The technical challenges involved in achieving this, pertain to both *data* and *system interoperability*. We need a way to make the semantics of Big Data explicit so that they can interlink and we need a way to make it transparent for the client applications to query federations of such heterogeneous systems.

In this context, *federated SPARQL query processing* is a natural place to look for a starting point in this effort. Semagrow is such a federated query processing system[3] that provides a single SPARQL endpoint that federates multiple remote SPARQL endpoints, transparently optimizing queries and dynamically integrating heterogeneous data models by applying the appropriate vocabulary transformations [2]. Semagrow hides schema heterogeneity and also applies methods from database research and artificial intelligence that take into account data contents to optimize querying plans.

The opportunity to extend Semagrow beyond federations of triple stores in a formally coherent way was presented by the results of the *Comma Separated Values on the Web (CSVW)* working group of the W3C[4] which has recently finalized a suite of recommendations on the semantics of tabular data [4] and on how to map tabular data to a semantically equivalent RDF graph [3]. The CSVW recommendations provide a formal grounding to both interoperability issues identified above: they provide for the formal interpretation of tabular data without formal semantics and they also provide a mapping through which SPARQL queries can be answered by such tabular data.

To make this more concrete, let us assume an example use case from the pilots of the Big Data Europe project where our work is encouched [1]: a Cassandra or Solr database of text and related metadata (such as authorship, time and place of publication, etc.) needs to be cross-linked with RDF data such as Geonames or DBPedia. CSVW allows us to both formally specify how to map placenames in the databases to location URIs and also allows us to specify how to interpret Cassandra columns as RDF properties.

In the remainder of this paper we will describe two alternative approaches to extend Semagrow so that it can construct federations of data sources other than SPARQL endpoints: using Semagrow-side connectors (Sections 2) or data source side SPARQL adapters (Section 3). We then discuss the relative merits of each and provide future research directions for this line of work (Section 4).

2 Federating Cassandra Databases

Cassandra offers the scalability, fault-tolerance, and elasticity of modern distributed stores, but the *Cassandra Query Language (CQL)*[5] is oriented towards processing tables and lacks the expressivity needed to join across tables. Moreover, Cassandra poses further restrictions on what queries that can be answered

[3] Cf. http://semagrow.github.io
[4] Cf. https://www.w3.org/2013/csvw
[5] Cf. http://cassandra.apache.org/doc/cql3/CQL.html

even when expressible within CQL. For example, filtering predicates can only be applied to attributes declared as indexed by the schema of each database.

These access restrictions make the connectivity with Cassandra more challenging than, for example, in the case of a full flenged SQL system. On one hand, not every SPARQL query can be directly translated into an equivalent CQL query. The query planner takes these restrictions into consideration and produce only valid plans that will directed towards a Cassandra source. On the other hand, Semagrow must compensate for the missing expressivity of Cassandra's queries by executing the remaining operations on its execution engine.

The connector is an open-source extension of the core Semagrow system.[6]

3 Federating Solr Databases

An alternative approach to the one described above is to provide the means of exposing non-RDF data as a SPARQL endpoint. The difference is that the source presents itself as a SPARQL endpoint, rather than having a Semagrow-side connector. To incorporate large amounts of textual data into a Semagrow federation, it is required to make data available as RDF and queryable using the OpenRDF Sesame API. Our approach to achieve this goal is to use Apache Solr as a backend for an OpenRDF Sail implementation. First, the use of Apache Solr gives us all advantages of a robust, scalable indexing framework, second the creation of an OpenRDF Sail Implementation based on Apache Solr will make such a full text index available for use with the SPARQL query language. It should be noted that Apache Solr is itself clusterable and can therefor be scaled and made highly available.

As a first step, the foundation of the Apache Solr implementation has been layed out by creating a port of the OpenRDF Sesame API that is capable of creating Java 8 Streams of RDF Statements. Apache Solr's streaming capabilities can be translated to Java 8 Streams. Currently work is undertaken to implement an Apache Solr specific version of the Semagrow Streaming Sail implementation. This will in fact combine the best of multiple worlds. It will be possible to use Lucene Search Syntax inside custom SPARQL functions, generally known as *magic predicates*. Additionally it will be possible to incorporate such an implementation seamlessly into the Semagrow federation.

4 Discussion and Conclusions

The syntactic integration of diverse data sources can be implemented either by a Semagrow-side connector or by a SPARQL adapter in the data source side that translates SPARQL queries to valid queries of the wrapped data source. While the SPARQL adapter seems to be compatible with every SPARQL client it is not always trivial to develop one. The translation of an arbitrary SPARQL query to a single query of the target language might not be fully supported due

[6] Cf. https://github.com/semagrow/connector-cassandra

to limitation imposed by the underlying data source. In such cases the SPARQL adapter needs to plan and implement the missing functionality needed in order to support every SPARQL query.

On the other hand, a Semagrow-side connector is aware about the limitations of the source and the query planner will produce plans that are directly translated into the target query language. The query planner might also be presented with more optimization decisions since it considers the global execution plan of the query, and thus produce more efficient execution plans than using SPARQL adapters to achieve the same effect. In other words, the Semagrow-side connector will leverage the query planner of Semagrow to decide how to better access the underlying sources and the execution engine to compensate for the possible limitations. Based on the above the Semagrow-side approach seems more appropriate in situations where there is a significant expressivity gap between SPARQL and the target query language of the source whereas SPARQL adapters are preferable when not.

The current implementation of the aforementioned Big Data connectors rely on the fact that the underlying data storage supplies a central endpoint that interacts with external requests. This fact simplifies the connectivity between systems, but the central endpoint of data exchange may become a bottleneck in data intensive scenarios. Our immediate future plan is to more tightly integrate Semagrow with Cassandra, so that the Semagrow execution engine is itself distributed and exploits data locality in its interactions with the individual Cassandra nodes.

Acknowledgements

The work described here was carried out in the context of Big Data Europe: Empowering Communities with Data Technologies. Big Data Europe has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644564. For more details, please visit https://www.big-data-europe.eu

References

[1] BigDataEurope: D5.2: Domain-specific big data integrator instances. Tech. rep., Public Deliverable (Jun 2016), https://www.big-data-europe.eu/results

[2] Charalambidis, A., Troumpoukis, A., Konstantopoulos, S.: Semagrow: Optimizing federated SPARQL queries. In: Proc. 11th Intl Conf. on Semantic Systems (SEMANTiCS 2015), Vienna, Austria (Sep 2015)

[3] Tandy, J., Herman, I., Kellogg, G.: Generating RDF from tabular data on the Web. W3C Recommendation, 17 December 2015 (Dec 2015), https://www.w3.org/TR/csv2rdf

[4] Tennison, J., Kellogg, G., Herman, I.: Model for tabular data and metadata on the Web. W3C Recommendation, 17 September 2015 (Dec 2015), https://www.w3.org/TR/tabular-data-model