Searching Linked Data and Services with a Single Query

Mohamed Lamine Mouhoub

Searching Linked Data and Services with a single query

Mohamed Lamine Mouhoub

PSL, Université Paris-Dauphine, 75775 Paris Cedex 16, France
CNRS, LAMSADE UMR 7243
mohamed.mouhoub@dauphine.fr

Abstract. The appearance of valuable Linked Data sources as part of the LOD in the last few years and the emergence of Semantic Web services has opened new horizons for web and data research. Moreover, machine-understandability of semantics allows for efficient search and integration of data. Based on the existing standards and techniques, we address the problem of searching data and services in the LOD. Furthermore, we aim to introduce an approach that integrates data queries with data from service calls and compositions. However, many challenges and issues need to be resolved in order to achieve such a goal. In this paper we briefly discuss some of such problems and solutions.

Keywords: #eswcphd2014Mouhoub, Semantic Web, Linked Data, Semantic Web Services, Service Discovery

1 Introduction

The appearance of valuable Linked Data sources such as DBPedia[1], YAGO[19], Freebase[2] and the advent of the Linked Open Data cloud[3] (LOD) has allowed access to terabytes of machine-understandable data. According to LODStats[4], a very recent LOD monitor[3], there are more than 61 billion triples over 2289 datasets in the LOD. However, many issues and challenges came up with the LOD such as :

– Some non-static content of the LOD can be outdated quickly unless the sources are updated frequently to provide fresh yet up-to-date information. For example, YAGO[5] which is a knowledge base extracted

[1] http://www.dbpedia.org/
[2] http://www.freebase.com/
[3] http://linkeddata.org/
[4] http://stats.lod2.eu/
[5] www.mpi-inf.mpg.de/yago-naga/yago/

from Wikipedia[6] hasn't been updated since late 2012. Another example is the population of Paris in DBpedia which is not up to date even in its DBpedia Live[7] version that comes right from Wikipedia.
- Data can be incomplete and needs complementary parts of information from other sources that don't necessarily belong to the LOD[14]. For example, events, places or friends around Paris are relatively important information for a user, yet these dynamic and social information doesn't reside in the LOD and require the usage of dedicated social APIs.
- Obviously, lots of data that lies within the WWW doesn't appear yet in the LOD. Lots of modeling and publishing efforts are needed to publish data as Linked Open Data.

On the other side, there are thousands of public Web Services (WS) that provide fresh and good quality information[14]. ProgrammableWeb[8] is one example of Service repositories that indexes more than 10 000 web service. Thus, results of queries on LOD can be enriched with fresh and complementary data from web services that hide lots of exploitable data. However, finding relevant WS and integrating their provided data with LOD data is quite a hard task and requires a lifting of web services to the semantic level. Semantic Web Services (SWS) are a key raw material for such an integration that are still not abundant but lots of research has been led on their description, discovery, and composition[9].

2 State of the Art

The aforementioned motivation covers a crossing of many Semantic Web applications including Linked Data management in the LOD and SWS publishing, discovery and composition. This section presents some state-of-the art work enumerated by their application category.

2.1 Query processing in the Linked Open Data cloud

Lots of recent and competitive works address the processing of distributed RDF stores. They can be classified into 2 branches[7, 4]:

1. 1) Distributed approaches: that target the distributed remote LOD sources. They can be divided into :1.a) Look-up based approaches

[6] http://www.wikipedia.com/
[7] http://live.dbpedia.org/
[8] http://www.programmableweb.com/

that download all the data from the remote LOD sources and run the queries locally. The lookup can be done either before execution like in [5] or on the fly during the query answering like in [6]. 1.b) Federation based approaches like [17] that send sub-queries to the remote SPARQL endpoints to be executed remotely then aggregates the returned results.

2. Parallel processing approaches like [4, 8] that consider the Linked Data as Big Data and use parallelization techniques like MapReduce among others to process the large amounts of RDF triples. However, parallel processing requires bringing all the data sets to the processing cluster or cloud infrastructure, quite like in the lookup based approaches.

The choice of linked data processing approach depends on the usage scenario. Parallel and lookup based approaches fit best for high performance in terms of response time but require lots of bandwidth and costs to download or keep the data sets up to date. Federated approaches allow to get updated data directly from the sources without downloading as the queries are processed at their corresponding sources. However, this late advantage is also a shortcoming because delegating queries to remote sources can delay considerably the execution time if some sources are slow or busy.

2.2 Semantic Service Discovery

A lot of research has been carried out in the last decade on semantic service discovery. The survey in [10] shows some of the latest SWS discovery tools and compares them based on many criteria. There are 3 famous benchmarks for SWS discovery that allow evaluating the existing approaches : SWS challenge[9], Semantic service selection (S3) contest[10] and The Web Service Challenge (WS-Challenge)[11]. According to the surveys in [10, 9], the ongoing semantic service discovery research can be categorized based on many criteria :
1) Service description : Depending on the description elements; i.e. functional elements (Inputs, Outputs, Preconditions and Effects IOPE) and non-functional elements (QoS, etc), service discovery approaches can be categorized into functional-based and non-functional-based approaches.
2) Technique : Despite description elements and language, the research on SWS discovery can be categorized into : 2.a) Logic-based approaches

[9] http://sws-challenge.org/wiki/index.php/Main_Page
[10] http://www-ags.dfki.uni-sb.de/ klusch/s3/
[11] http://ws-challenge.georgetown.edu/

that reason on the semantic description elements of SWS using a semantic reasoning technique. 2.b) Non-logic based are mostly based on textual similarity of concept names. 2.c) Hybrid techniques that combine both techniques .

2.3 Data and service search

Aggregated Search of Data and Services[12] proposes to answer an SQL-like data query on XML datasets and RDBMS and propose relevant services to the latter. It generates a semantic graph for I/O of WSDL services using a user provided ontology and Wordnet[12]. After that it matches the query keywords with the generated service semantic graph keywords to find relevance and propose services to the user. It uses a non-logic based textual similarity to discover services. Therefore, this approach needs to be adapted to allow a search for semantic data and services as semantics allow for advanced reasoning that offers more accurate results. Moreover, many shortcomings can be considered such the error rates due to the automatic generation of semantic descriptions.

ANGIE[14] is a tool to enrich an RDF store with information from RESTful and SOAP-based services. The approach relies on mappings between the concepts in the RDF store and elements of XML data provided by services. It assumes that WS are described according to a common global schema defined by a provided ontology like YAGO. The idea is to make the data provenance transparent to the user and invoke web services on-the-fly to answer queries. To answer a user query, services are composed and invoked following an execution plan generated on basis of the global schema and not on the descriptions of WS. At the execution level, services in repositories are matched with the composition requirements, then their results are mapped with the global schema. However, the global schema assumption can only be true in domain-specific WS repositories and doesn't fit to the diversity and heterogeneity nature of the LOD. Furthermore, the evolution of SW repositories (new services, new ontologies, etc) makes it difficult to maintain a global schema even for a specific domain.

Sig.ma[20] is an entity search tool that uses Sindice[11] to extract all related facts for a given entity. Sindice is a offers a platform to index, search and query documents with semantic markup in the web. It crawls the web continuously to index new documents and update the indexed

[12] http://wordnet.princeton.edu/

ones. Both Sig.ma and Sindice are document-based and don't offer SWS discovery features or search for data using SWS.

3 Problem and contributions

The semantic web is a fast growing research field and its standards and technologies are being more and more adopted especially by organizations and open data providers[13]. With such promising future, research can be pushed further on semantic web services and how to better involve them in the LOD[1].

As we stated in section 1, users need to search for linked data and complete the LOD answers with semantic web services. In parallel developers need an easy way to discover useful services to integrate in their mashups and applications built on Linked Data. To our knowledge, there is no existing tool that fully supports such a search over the LOD and existing similar aforementioned approaches are limited to some constraints and usages that are incompatible with the nature and the content of the LOD. Therefore, many techniques and tools can be put together and reused to achieve the determined goal. However, there are many issues that can be identified when it comes to putting all the pieces together :

- How to understand the user query and how to convert it into a service request ? How to deal with the different query templates that use UNION, OPTIONAL, FILTER, etc. The parameters of the service request must extracted from the data query, definitions of concepts must be extracted. Inputs, outputs and conditions must be distinguished.
- Is there a need for extending SPARQL 1.0 (or SPARQL 1.1) for sufficient expressivity in order to adapt the user query to the service requests. If yes, then how to define this syntax [2] and how to rewrite these no-standard queries to send them to SPARQL endpoints in the LOD.
- How to address the heterogeneity of SWS descriptions? How to request services from multiple SWS deployed in a distributed fashion in the LOD? How to select, aggregate and rank the discovered services.
- How to compose services on the fly in order to answer the user query and integrate the results with linked data.
- How to expand the relevance criteria for service requests by entailing hierarchy and similarity between concepts ?

[13] http://www.huffingtonpost.com/steve-hamby/semantic-web-technology_b_1228883.html

– How to measure the accuracy of the resulting services and verify the coherence of the resulted data ? I/O are not sufficient to find accurate services and compositions. As shown in [21], the SWS in our case are data providers and they have specific constrainsts on I/O. [15] introduces a representation method for IOPE that explicitly defines the link between I/O based on Preconditions and Effects using SPARQL.

Our goal is to provide a provide a platform that allows to combine a search of linked data and services to find both data and relevant services that can be mashed up with. Such a search often requires distinct queries : a) queries that lookup in the LOD to find data and b) service requests that discover relevant services in some SWS repositories. Our contribution is to provide a platform that allows to search for both starting from a data query, i.e. a query intended to search only for data. Starting from such a query, we automatically extract service requests and find relevant services to that data or generate service compositions that provide complementary data. Section 4 shows briefly our approach and some of what we have achieved so far.

Preliminarily, we assume that Semantic Web Services are described using any RDF based description language/ontology: OWL-S, WSMO, MSM, etc. These services are published in public repositories and are either stored in RDF stores behind SPARQL points or simply as RDF dumps. Many tools already provide that feature (iServe[13] LIDS[18]). The Inputs/Outputs are important parts of the descriptions that should reference to existing ontologies. Moreover, the links between Inputs and Outputs are essential for high accuracy service discovery and composition.

4 Proposed approach

The goal of our approach is not to replace the existing efforts but to push the research further by building on top of them. Based on the statements in section 1, we want our approach to focus on Linked data and Semantic Web Services and we differentiate it by exploring the potential of the existing standards and techniques in Semantic Web.

When a SPARQL query is submitted by a user or an agent, it launches two disjoint processes that can be joint later by aggregation. One process is dedicated to manage the query answering in the LOD data sources. The later sources are distributed and either accessible via SPARQL endpoints or dumped in RDF files. An appropriate technique among the ones mentioned in section 2.1 must be used depending on the data source na-

tures and the appropriate optimization and rewriting are performed at this level.

The other process is dedicated to discover and possibly compose services that are relevant to the data query. To deal with the heterogeneity of the SWS descriptions and the distributed deployments of repositories containing them, we choose to issue service requests in SPARQL queries adapted to each description language. This allows us to natively select SWS and perform logical reasoning on their descriptions to extend the search capabilities. Furthermore, this allows us deal with the heterogeneous descriptions more effectively without intermediate mapping tools.

The data query is analyzed to extract elements that can be used as I/O for a service request. Outputs are simply the selected variables of the query. Inputs are the bound values that appear in the triples of the query. The links between Inputs and Outputs can be established as Preconditions and Effects and represented using SPARQL as in [15].

SPARQL operators like OPTIONAL, UNION, FILTER, etc can reveal the preferences of the user for service discovery and composition. For instance, the I/O extracted from an Optional block probably mean that the user wants services that don't necessarily provide the optional parts.

A crucial step follows in which ontological concepts are attributed to I/O in order to make accurate service requests. Typically, and rdf:type property in the query declares the concept of a given element. However, many issues arise when this declaration is missing in the query. Moreover expanding the search of relevant concepts to sub, super or similar concepts requires finding such concepts in the corresponding ontologies or in the LOD. Among the issues is the cost of requests from the servers that store the ontologies or the services.

Once the service request elements are gathered, service discovery queries are issued in SPARQL using specific templates that correspond to each SWS description language. We retain two possible kinds of use cases for service selection. In the first, the user would like to select services that provide him the same data as in his query, as if he is looking for SWS alternatives to LOD sources. In the second the user wants to find any services that provide parts of his desired outputs or consume some of his provided inputs. The latter case is practical in assisting the user in building mashups on the go. Automatic service composition comes next to provide composite services that provide answers or complementary information to the user query.

A last aggregation step integrates both results from data sources and services in case composite services where composed and invoked. Other-

wise, the relevant services to the query are ranked and ordered by their relevance.

The methodology of our research is guided by the global process described in this section. At a first stage, we need to focus on understanding the user queries and transforming them into service requests. At the same time, existing and emerging issues must be solved. Next, we can address the SWS discovery and composition and see what are the issues related to our goal that need to be resolved. On the data side, the query federation issues must be addressed, especially for SWS repositories. Caching and rewriting techniques must be conceived and tested in our particular usage scenario.

Evaluating the performance and measuring the accuracy of the established discovery and composition algorithms would allow us to partially validate our approach and confirm its feasibility. Evaluation can be made at two levels :

1. Low level : at this level, we will evaluate distinctly the performance of our query answering, service discovery and service composition using dedicated benchmarks and test collections such as FedBench[16] for data querying or OWL-S-TC[14] for service discovery.
2. High level : at this level, we will evaluate jointly all the features above put together to measure the accuracy and the performance of our search platform. A major constraint for this evaluation is to use data and service test collections that contain mutually equivalent data; i.e. common concepts, common ontologies, etc. Moreover, data and services should be deployed in a distributed fashion as in the LOD. Another challenge is to provide typical queries and typical answers in order to compare them to the platform answers and measure the accuracy.

5 Preliminary implementation

We have already implemented a prototype that allows via a GUI to write a SPARQL query and execute the data and service queries. On the data side, we currently use FedX [17] to process the data queries on the LOD. On the service side, we have written preliminary algorithms to write service requests from data queries and answer them from SWS repositories that have SPARQL endpoints. Figure 1 gives an overview of the service discovery process that we have made so far.

[14] http://projects.semwebcentral.org/projects/owls-tc/

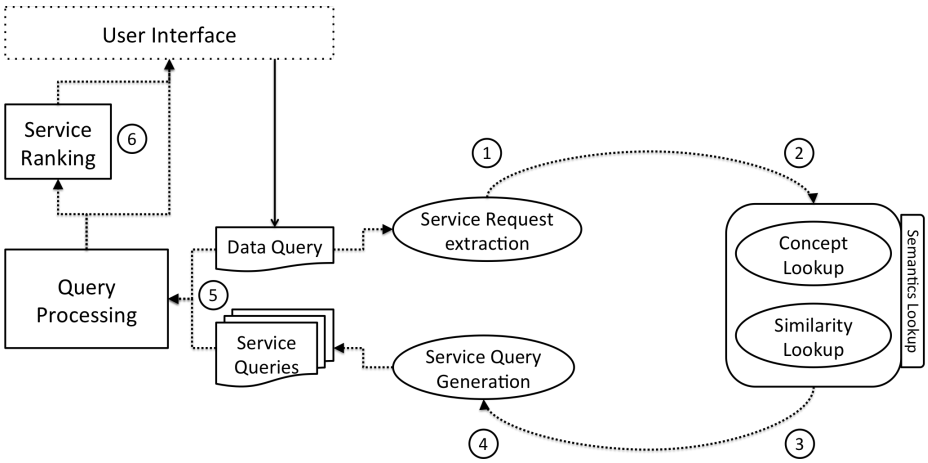

Fig. 1. Process of SWS discovery in using a data query

References

1. Tim Berners-Lee, James Hendler, Ora Lassila, et al. The semantic web. *Scientific american*, 284(5):28–37, 2001.
2. Carlos Buil-Aranda, Marcelo Arenas, Oscar Corcho, and Axel Polleres. Federating queries in sparql 1.1: Syntax, semantics and evaluation. *Web Semantics: Science, Services and Agents on the World Wide Web*, 18(1):1–17, 2013.
3. Ivan Ermilov, Michael Martin, Jens Lehmann, and Sren Auer. Linked open data statistics: Collection and exploitation. In Pavel Klinov and Dmitry Mouromtsev, editors, *Knowledge Engineering and the Semantic Web*, volume 394 of *Communications in Computer and Information Science*, pages 242–249. Springer Berlin Heidelberg, 2013.
4. Luis Galárraga, Katja Hose, and Ralf Schenkel. Partout: A distributed engine for efficient rdf processing. *CoRR*, abs/1212.5636, 2012.
5. Andreas Harth, Katja Hose, Marcel Karnstedt, Axel Polleres, Kai-Uwe Sattler, and Jürgen Umbrich. Data summaries for on-demand queries over linked data. In *Proceedings of the 19th international conference on World wide web*, pages 411–420. ACM, 2010.
6. Olaf Hartig, Christian Bizer, and Johann-Christoph Freytag. Executing sparql queries over the web of linked data. In *Proceedings of the 8th International Semantic Web Conference*, ISWC '09, pages 293–309, Berlin, Heidelberg, 2009. Springer-Verlag.
7. Olaf Hartig and Andreas Langegger. A database perspective on consuming linked data on the web. *Datenbank-Spektrum*, 10(2):57–66, 2010.
8. Zoi Kaoudi, Ioana Manolescu, et al. Triples in the clouds. In *ICDE-29th International Conference on Data Engineering*, 2013.
9. Matthias Klusch. Service discovery. *in Encyclopedia of Social Networks and Mining (ESNAM), R. Alhajj and J. Rokne, Eds. Springer*, 2014.
10. Le Duy Ngan and Rajaraman Kanagasabai. Semantic web service discovery: State-of-the-art and research challenges. *Personal Ubiquitous Computing*, 17(8):1741–1752, December 2013.

11. Eyal Oren, Renaud Delbru, Michele Catasta, Richard Cyganiak, Holger Stenzhorn, and Giovanni Tummarello. Sindice. com: a document-oriented lookup index for open linked data. *International Journal of Metadata, Semantics and Ontologies*, 3(1):37–52, 2008.
12. Matteo Palmonari, Antonio Sala, Andrea Maurino, Francesco Guerra, Gabriella Pasi, and Giuseppe Frisoni. Aggregated search of data and services. *Information Systems*, 36(2):134–150, 2011.
13. Carlos Pedrinaci, Dong Liu, Maria Maleshkova, David Lambert, Jacek Kopecky, and John Domingue. iserve: a linked services publishing platform. In *CEUR workshop proceedings*, volume 596, 2010.
14. Nicoleta Preda, Fabian M Suchanek, Gjergji Kasneci, Thomas Neumann, Maya Ramanath, and Gerhard Weikum. Angie: Active knowledge for interactive exploration. *Proceedings of the VLDB Endowment*, 2(2):1570–1573, 2009.
15. Marco Luca Sbodio, David Martin, and Claude Moulin. Discovering semantic web services using sparql and intelligent agents. *Web Semantics: Science, Services and Agents on the World Wide Web*, 8(4):310–328, 2010.
16. Michael Schmidt, Olaf Görlitz, Peter Haase, Günter Ladwig, Andreas Schwarte, and Thanh Tran. Fedbench: A benchmark suite for federated semantic data query processing. In *The Semantic Web–ISWC 2011*, pages 585–600. Springer, 2011.
17. Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt. Fedx: optimization techniques for federated query processing on linked data. In *The Semantic Web–ISWC 2011*, pages 601–616. Springer, 2011.
18. Sebastian Speiser and Andreas Harth. Integrating linked data and services with linked data services. In *The Semantic Web: Research and Applications*, pages 170–184. Springer, 2011.
19. Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007.
20. Giovanni Tummarello, Richard Cyganiak, Michele Catasta, Szymon Danielczyk, Renaud Delbru, and Stefan Decker. Sig.ma: Live views on the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 8(4):355–364, 2010.
21. Roman Vaculín, Huajun Chen, Roman Neruda, and Katia Sycara. Modeling and discovery of data providing services. In *Web Services, 2008. ICWS'08. IEEE International Conference on*, pages 54–61. IEEE, 2008.