Linked Data Query Processing

Tutorial at ICDE 2014

Olaf Hartig [1], M. Tamer Özsu [2]

Cheriton School of Computer Science, University of Waterloo
200 University Ave. West, Waterloo, Ontario, Canada N2L 3G1
[1] ohartig@uwaterloo.ca [2] tamer.ozsu@uwaterloo.ca

***Abstract*—The publication of Linked Open Data on the Web has gained tremendous momentum over the last six years. As a consequence, we currently witness the emergence of a new research area that focuses on an *online* execution of *Linked Data queries*; i.e., declarative queries that range over Web data that is made available using the Linked Data publishing principles.**

These principles only require Web servers that respond to simple requests for data about given entities. Therefore, in contrast to approaches for querying a more traditional distributed database, Linked Data query processing approaches cannot assume that data sources provide query processing functionality. Additional challenges are the unbounded nature of the Web and the lack of a complete, up-to-date database catalog that lists all data sources.

Our tutorial provides an overview of the new area of Linked Data query processing. We introduce the foundations of Linked Data queries, discuss the specific challenges that need to be addressed, and review techniques for executing such queries.

I. Introduction

In recent years, the amount of Linked Open Data on the World Wide Web (WWW) has been increasing rapidly [1], [2], [3]. This is largely due to community efforts such as the Linking Open Data project [4], and increasing interests of enterprises and governments. Datasets made openly available on the WWW as Linked Data cover different domains, including life sciences (e.g., DrugBank, UniProt, PubMed), geographic locations (e.g., World Factbook, Geo Names), media and entertainment (e.g., MusicBrainz, Last.FM, BBC Programmes, New York Times' subject headings), products and offers (e.g., Best Buy's product catalog, Renault's car configurations). There are also cross-domain encyclopedic datasets such as Google's Freebase and DBpedia (the structured data counterpart of Wikipedia). Besides researchers and enterprises, many governments (e.g., US, UK) have started to make data of public interest available as Linked Open Data, including data on CO_2 emissions, mortality, energy, and postcodes.

All of these datasets are published according to the Linked Data principles [5], a simple and accessible paradigm that requires data providers to use: (i) HTTP [6], as data access protocol, (ii) HTTP-scheme-based URIs [7], as identifiers for entities described in the data, and (iii) RDF [8], as the data model to represent data. Any HTTP-URI in an RDF triple may then be understood as a *data link* that enables Linked-Data-aware software clients to retrieve more data by looking up the URI on the Web. The simplicity and convenience of these principles is one of the drivers behind the proliferation of Linked Data.

However, the simple, URI lookup-based data access method that goes along with these principles does not provide the powerful querying capabilities usually associated with data sources in a distributed database system. While some dataset providers offer additional query processing services to overcome this limitation [9], providing and maintaining such a service presents a much more significant investment (with often little return). Many datasets are therefore not available through such services or the services are often unreliable [10].

To support live querying use cases for Linked Data without assuming query processing capabilities by data providers, some research groups have started to study the problem of *Linked Data query processing*, that is, executing SQL-like queries over Linked Data *on the WWW*, by relying only on the Linked Data principles. The novelty and challenge of this problem lies in the characteristics that make the WWW different from traditional distributed database systems. In particular, the WWW is an open, virtually unbounded dataspace for which we cannot assume the existence of a complete, up-to-date database schema and catalog.

Our tutorial discusses these characteristics and introduces approaches proposed to address the challenges that arise as a result. Furthermore, we highlight open problems that provide opportunities for contributing to the emerging area of Linked Data query processing. The following sections of this short paper outline the topics covered in the tutorial.

II. Preliminaries

As a basis for the tutorial we provide a brief overview on the standards and best practices for publishing Linked Data on the WWW. In particular, we introduce the Resource Description Framework (RDF) [8], which is a graph-based data model that is commonly used for representing the data published as Linked Data, and we describe the aforementioned Linked Data principles [5] and the notion of data links established by the use of HTTP-scheme-based URIs [7].

III. Foundations

An awareness of query languages, query semantics, and formal properties of Linked Data queries is essential for understanding the possibilities and limitations of Linked Data query processing. Therefore, before we focus on systems-related topics for the major part of the tutorial, we provide an overview on these fundamental aspects of querying Linked Data.

A. Query Languages

As of today, there exist two proposals for query languages that explicitly target the use case of querying Linked Data on the WWW. Both of these languages, NautiLOD [11] and LDPath [12], are navigational. While queries in these languages are similar in nature to XPath expressions for XML data [13] or regular path queries for graph databases [14], the specified navigation paths refer to the Web graph that emerges from the existence of data links.

However, instead of supporting NautiLOD or LDPath, almost all approaches to Linked Data query processing focus on a fragment of SPARQL [15], which is the standard query language for the RDF data model. Therefore, although not originally defined for this purpose, our tutorial introduces SPARQL as a language for expressing queries over Linked Data on the WWW.

The basic building block of SPARQL queries are RDF graph patterns. The (standard) semantics of SPARQL is based on sub-graph matching; that is, the expected result of evaluating a SPARQL query over a given RDF data graph is defined in terms of sub-graphs of the data graph that match the query.

B. Query Semantics

Multiple proposals exist for adapting the standard SPARQL semantics such that SPARQL can be used to query Linked Data in a well-defined manner [16], [17], [18], [19]. The most prevalent approaches are a full-Web query semantics and several reachability-based query semantics.

Informally, the scope of a SPARQL query under *full-Web query semantics* is the complete set of all Linked Data on the WWW. *Reachability-based query semantics* restrict such a scope to data that is reachable by traversing recursively a well-defined set of data links.

C. Theoretical Properties

Unsurprisingly, the computational feasibility of (satisfiable) SPARQL queries under full-Web semantics is very limited [17]. In practice, there cannot exist a query execution approach that guarantees an execution of such a query that both terminates and returns the complete query result. Moreover, if such a query is non-monotonic, it is not even possible to guarantee a *nonterminating* execution that eventually returns all elements of the complete result [17].

In contrast, it is possible to design query execution approaches that guarantee complete, terminating executions of SPARQL queries under reachability-based query semantics (assuming Linked Data on the WWW is finite) [17].

IV. Source Selection Strategies

For the execution of Linked Data queries, it is necessary to retrieve data by looking up URIs. There exist three classes of approaches for selecting the URIs that a query execution system looks up during the execution of a query: index-based approaches, live exploration approaches, and hybrid approaches.

A. Index-Based Source Selection

Index-based approaches rely on a pre-populated index which is used for identifying URIs to look up during query execution time [20], [21]. Hence, in contrast to index structures that store the data itself, the index-based source selection approaches use data structures that index URIs as pointers to data.

A typical example for such a data structure uses patterns of RDF triples as index keys [22]. Given such a pattern, the corresponding index entry is a set of URIs such that looking up each of these URIs provides us with some data that contains an RDF triple that matches the pattern. Further index structures (for source selection) have been studied [20], [21], [23], [24].

After populating an initial version of such an index, it is necessary to maintain the index. Maintenance may include adding additionally discovered URIs and keeping the index up to date [21]. The latter is necessary because what data can be retrieved from indexed URIs might change over time. We note that the challenges for such an index maintenance are similar to maintaining materialized views in a data warehouse. However, to our knowledge, no work exists that studies approaches to maintain indexes for Linked Data query execution.

B. Live Exploration

Live exploration approaches make use of the characteristics of Linked Data, in particular, the existence of data links. That is, to execute a given Linked Data query, live-exploration-based systems perform a recursive URI lookup process during which they incrementally discover further URIs that can be scheduled for lookup [25], [26], [27], [28], [29], [30]. Thus, such a system explores the WWW by traversing data links *at query execution time*. While the data retrieved during such an exploration allows for a discovery of more URIs to look up, it also provides the basis for constructing the query result. Hence, live-exploration-based systems may support naturally the aforementioned reachability-based query semantics [25].

An interesting characteristic of live exploration approaches is the potential for serendipitous discovery of initially unknown data sources. Furthermore, live exploration does not require any a-priori information; instead, a live-exploration-based system might readily be used without having to wait for the completion of an initial data load phase or any other type of preprocessing. On the downside, possibilities for parallelizing data retrieval are limited because of the recursive nature of the lookup process. Hence, in comparison to index-based source selection, an efficiently implemented index-based system—which may determine all URIs for lookup at the beginning of a query execution—might answer a Linked Data query faster than a live-exploration-based system (assuming both systems eventually look up the same set of URIs during the execution). On the other hand, the initialization of such an index-based system may take a significant amount of time.

However, given that both strategies (index-based source selection and live exploration) may be implemented in a multitude of ways, and an unbounded number of query semantics may be supported, it is a challenging (and still open) research problem to compare both strategies in a fair manner.

C. Hybrid Approaches

Hybrid source selection combines an index-based approach with a live exploration approach in order to achieve the advantages of both approaches without inheriting their respective shortcomings. An example of such a hybrid approach is to exploit a pre-populated index to obtain an initial version of a (ranked) list of URIs to look up; additional URIs discovered during the query execution are then integrated into the list [22].

An alternative hybrid approach that may be studied might use an index only to prioritize discovered data links and, thus, to control a live exploration process. This process may than feed back information for updating, for expanding, or for reorganizing the index.

V. Source Ranking Strategies

In addition to applying a source selection approach to select URIs that have to be looked up for a given Linked Data query, a query execution system may rank the set of selected URIs such that the ranks represent a priority for the lookup of these URIs. Such a data source ranking may allow the system to minimize its response time or to maximize the subset of the query result computed in a given amount of time.

Existing work on data source ranking for Linked Data queries focuses on index-based source selection [20], [22]; information required for the proposed ranking methods is assumed to be available in the corresponding index structures.

VI. Query Execution Process

The actual process of executing a Linked Data query may consist of two separate phases: During the first phase, a query execution system selects URIs and uses them to retrieve data from the queried Web; during a subsequent, second phase, the system generates the query result using the data retrieved in the first phase [20], [21], [24]. Instead of separating these two phases, it is also possible to integrate the retrieval of data into the result construction process [26], [22], [27], [28], [29], [25], [30], [31], [32]. We refer to Linked Data query execution approaches that apply the latter idea as *integrated execution approaches*. Analogously, *separated execution approaches* clearly separate data retrieval from result construction by two consecutive phases.

A. Separated Execution Approaches

Due to the clear separation of data retrieval and result construction, separated approaches are straightforward to implement. In particular, the aforementioned index-based strategy for source selection lends itself naturally to such an implementation [20], [21], [24]. However, it is also easy to develop a separated execution approach based on live exploration.

A disadvantage of separated execution approaches is that any element of the query result can only be reported after completing the data retrieval phase. Looking up a large set of selected URIs or retrieving the complete set of reachable data may take a prohibitively long time; it may even exceed the resources of the query execution system.

B. Integrated Execution Approaches

Query execution systems that implement an integrated execution approach might begin returning first elements of a query result early; i.e., before data retrieval has been completed.

As for separated approaches it is possible to use any type of source selection as a basis for an integrated execution approach. In fact, a manifold of integrated approaches are conceivable for each source selection strategy, some of which have already been studied in the literature [26], [27], [22], [28], [29], [30], [31]. These approaches use different implementation techniques, including:

- an application of the symmetric hash join operator for implementing integrated execution approaches in a push-based manner [22], [28],
- another push-based implementation that uses the Rete match algorithm [30], and
- a pull-based implementation that makes use of the well-known iterator model [26], [27], [32].

VII. Query Planning and Optimization

An essential step of any procedure for processing queries is the query planning phase. While the usual objective for selecting a query execution plan is to minimize the overall execution time, possible optimization criteria in the context of Linked Data queries are more diverse and include minimizing response time or network traffic, or maximizing the degree of result completeness (under full-Web query semantics).

Indexes for source selection may store some information for assessing query execution plans w.r.t. these optimization criteria. However, although the aforementioned source ranking methods can be conceived as a form of query optimization, there does not yet exist any work that investigates more traditional cost-based query optimization techniques in the context of index-based approaches to Linked Data query processing.

For live-exploration-based source selection, information to assess query plans cannot assumed to be available initially; instead, it may only be obtained incrementally during query execution. Hence, in this case, heuristics are required for selecting an initial plan. Such plan selection heuristics have been proposed for the aforementioned, iterator-based approach to implement (integrated) Linked Data query execution [27].

For the same implementation approach an extension of the iterator model has been introduced that allows for runtime query optimization [26]. This extension avoids a blocking behavior of iterators that may otherwise happen as a result of the need to wait for a completion of certain URI lookups.

VIII. Open Problems

Throughout the tutorial we highlight open research problems, including the following:

- While there exist navigational query languages for Linked Data on the WWW [11], [12], there does not exist research on execution techniques for the queries that can be expressed using these (or similar) languages.
- We identify a lack of comprehensive experimental comparisons of approaches to Linked Data query processing.

In fact, it is an open question how to compare experimentally different types of such approaches in a fair and meaningful manner (e.g., approaches that implement an index-based source selection strategy vs. live exploration approaches, separated execution approaches vs. integrated execution approaches).

- Related to the lack of experimental comparisons, there do not exist well-defined benchmarks to test Linked Data query processing systems (neither for specific types of Linked Data query processing approaches nor for Linked Data query processing in general).
- Query optimization in the context of Linked Data query processing is a largely unexplored area. Interesting topics of research in this context include source ranking for live-exploration-based source selection, adaptive query processing, and multi-objective query optimization.
- For some applications it is desirable (or even required) to combine a Linked Data query processing approach with other query paradigms such as query federation or queries over a centralized collection of data. Such combinations represent another interesting topic of research that only a few works have began to investigate [28], [33], [34], [35].

IX. Conclusion

Since Linked Data query processing is a very young topic, our goal is to increase awareness of this topic in the database community. Consequently, our tutorial primarily targets:

- students who are looking for a novel research topic,
- experienced researchers who are interested in new challenges, as well as
- industry participants who seek novel ways for using and exploiting Linked Data.

References

[1] C. Bizer, T. Heath, and T. Berners-Lee, "Linked Data – The Story So Far," *Int. Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 5, no. 3, pp. 1–22, 2009.

[2] P. Mika and T. Potter, "Metadata Statistics for a Large Web Corpus," in *Proc. of the 5th Linked Data on the Web Workshop (LDOW)*, 2012.

[3] H. Mühleisen and C. Bizer, "Web Data Commons – Extracting Structured Data from Two Large Web Corpora," in *Proc. of the 5th Linked Data on the Web Workshop (LDOW)*, 2012.

[4] C. Bizer, T. Heath, D. Ayers, and Y. Raimond, "Interlinking Open Data on the Web," in *Proc. of the Poster Session at the 4th European Semantic Web Conference (ESWC)*, 2007.

[5] T. Berners-Lee, "Design Issues: Linked Data." [Online]. Available: http://www.w3.org/DesignIssues/LinkedData.html

[6] R. Fielding, J. Gettys, J. C. Mogul, H. Frystyk, L. Masinter, P. J. Leach, and T. Berners-Lee, "Hypertext Transfer Protocol – HTTP/1.1," RFC 2616, Jun. 1999. [Online]. Available: http://tools.ietf.org/html/rfc2616

[7] T. Berners-Lee, R. Fielding, and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax," RFC 3986, Jan. 2005. [Online]. Available: http://tools.ietf.org/html/rfc3986

[8] G. Klyne and J. J. Carroll, "Resource Description Framework (RDF): Concepts and Abstract Syntax," W3C Recommendation, Feb. 2004. [Online]. Available: http://www.w3.org/TR/rdf-concepts/

[9] L. Feigenbaum, G. T. Williams, K. G. Clark, and E. Torres, "SPARQL 1.1 Protocol," W3C Recommendation, Mar. 2013. [Online]. Available: http://www.w3.org/TR/sparql11-protocol/

[10] C. Buil-Aranda, A. Hogan, J. Umbrich, and P.-Y. Vandenbussche, "SPARQL Web-Querying Infrastructure: Ready for Action?" in *Proc. of the 12th International Semantic Web Conference (ISWC)*, 2013.

[11] V. Fionda, C. Gutierrez, and G. Pirró, "Semantic Navigation on the Web of Data: Specification of Routes, Web Fragments and Actions," in *Proc. of the 21th World Wide Web Conference (WWW)*, 2012.

[12] S. Schaffert, C. Bauer, T. Kurz, F. Dorschel, D. Glachs, and M. Fernandez, "The linked media framework: Integrating and interlinking enterprise media content and data," in *Proc. of the 8th International Conference on Semantic Systems (I-Semantics)*, 2012.

[13] J. Clark and S. DeRose, "XML Path Language (XPath)," W3C Recommendation, Nov. 1999. [Online]. Available: http://www.w3.org/TR/xpath

[14] L. Libkin and D. Vrgoc, "Regular Path Queries on Graphs with Data," in *Proc. of the 15th Int. Conference on Database Theory (ICDT)*, 2012.

[15] S. Harris, A. Seaborne, and E. Prud'hommeaux, "SPARQL 1.1 Query Language," W3C Recommendation, Mar. 2013. [Online]. Available: http://www.w3.org/TR/sparql11-query/

[16] P. Bouquet, C. Ghidini, and L. Serafini, "Querying The Web Of Data: A Formal Approach," in *Proc. of the 4th Asian Semantic Web Conference (ASWC)*, 2009.

[17] O. Hartig, "SPARQL for a Web of Linked Data: Semantics and Computability," in *Proc. of the 9th Extended Semantic Web Conference (ESWC)*, 2012.

[18] A. Harth and S. Speiser, "On Completeness Classes for Query Evaluation on Linked Data," in *Proc. of the 26th AAAI Conference*, 2012.

[19] J. Umbrich, A. Hogan, A. Polleres, and S. Decker, "Improving the Recall of Live Linked Data Querying through Reasoning," in *Proc. of the 6th Int. Conference on Web Reasoning and Rule Systems (RR)*, 2012.

[20] A. Harth, K. Hose, M. Karnstedt, A. Polleres, K.-U. Sattler, and J. Umbrich, "Data Summaries for On-Demand Queries over Linked Data," in *Proc. of the 19th World Wide Web Conference (WWW)*, 2010.

[21] J. Umbrich, K. Hose, M. Karnstedt, A. Harth, and A. Polleres, "Comparing Data Summaries for Processing Live Queries over Linked Data," *World Wide Web*, vol. 14, no. 5–6, pp. 495–544, 2011.

[22] G. Ladwig and D. T. Tran, "Linked Data query processing strategies," in *Proc. of the 9th International Semantic Web Conference (ISWC)*, 2010.

[23] Y. Tian, J. Umbrich, and Y. Yu, "Enhancing Source Selection for Live Queries over Linked Data via Query Log Mining," in *Proc. of the Joint Int. Semantic Technology Conference (JIST)*, 2011.

[24] E. Paret, W. Van Woensel, S. Casteleyn, B. Signer, and O. De Troyer, "Efficient Querying of Distributed RDF Sources in Mobile Settings based on a Source Index Model," *Procedia CS*, 2011.

[25] O. Hartig and J.-C. Freytag, "Foundations of Traversal Based Query Execution over Linked Data," in *Proc. of the 23rd ACM Conference on Hypertext and Social Media (HT)*, 2012.

[26] O. Hartig, C. Bizer, and J.-C. Freytag, "Executing SPARQL Queries over the Web of Linked Data," in *Proc. of the 8th International Semantic Web Conference (ISWC)*, 2009.

[27] O. Hartig, "Zero-Knowledge Query Planning for an Iterator Implementation of Link Traversal Based Query Execution," in *Proc. of the 8th Extended Semantic Web Conference (ESWC)*, 2011.

[28] G. Ladwig and D. T. Tran, "SIHJoin: Querying Remote and Local Linked Data," in *Proc. of the 8th Extended Semantic Web Conference (ESWC)*, 2011.

[29] F. Schmedding, "Incremental SPARQL Evaluation for Query Answering on Linked Data," in *Proc. of the 2nd Int. Workshop on Consuming Linked Data (COLD)*, 2011.

[30] D. P. Miranker, R. K. Depena, H. Jung, J. F. Sequeda, and C. Reyna, "Diamond: A SPARQL Query Engine, for Linked Data Based on the Rete Match," in *Proc. of the Workshop on Artificial Intelligence meets the Web of Data (AImWD)*, 2012.

[31] A. Wagner, T. Tran, G. Ladwig, and A. Harth, "Top-K Linked Data Query Processing," in *Proc. of the 9th Extended Semantic Web Conference (ESWC)*, 2012.

[32] O. Hartig, "SQUIN: A Traversal Based Query Execution System for the Web of Linked Data," in *Proc. of the 2013 ACM SIGMOD International Conference on Management of Data*, 2013.

[33] J. Umbrich, M. Karnstedt, A. Hogan, and J. X. Parreira, "Hybrid SPARQL Queries: Fresh vs. Fast Results," in *Proc. of the 11th International Semantic Web Conference (ISWC)*, 2012.

[34] O. Hartig, "How Caching Improves Efficiency and Result Completeness for Querying Linked Data," in *Proc. of the 4th Linked Data on the Web Workshop (LDOW)*, 2011.

[35] S. Lynden, I. Kojima, A. Matono, A. Nakamura, and M. Yui, "A Hybrid Approach to Linked Data Query Processing with Time Constraints," in *Proc. of the 6th Linked Data on the Web Workshop (LDOW)*, 2013.