Noname manuscript No.
(will be inserted by the editor)

A Database Perspective on Consuming Linked Data on the Web

Olaf Hartig · Andreas Langegger

Abstract During recent years an increasing number of data providers adopted the Linked Data principles for publishing and connecting structured data on the Web, thus creating a globally distributed dataspace – the Web of Data. While the execution of structured, SQL-like queries over this dataspace opens possibilities not conceivable before, query execution on the Web of Data poses novel challenges. These challenges provide great opportunities for the database community.

In this article we introduce the concept of Linked Data and discuss different approaches to query the Web of Data. Our goal is to provide a general understanding of this new research area and of the challenges and open issues that must be addressed.

Keywords Linked Data · Web of data · query processing

1 Introduction

The term *Linked Data* originates from a Web architecture note [3] that introduces a set of simple principles. Basically, these principles – which became known as the *Linked Data principles* – propose to publish and to connect structured data on the Web in a manner similar to the approach which content providers have used for Web documents for the last twenty years. A wide-spread application of these principles enables an evolution of the World Wide Web into a single, globally distributed dataspace, often called the *Web of Data*.

Since the Linked Data principles have been proposed in 2006, a grass-roots movement started to publish and interlink multiple open databases on the Web following these principles [6]. Today an increasing number of data publishers such as the BBC, Thomson Reuters, The New York Times, the Library of Congress, and the UK government adopt this practice. This ongoing effort resulted in bootstrapping the Web of Data which, today, comprises billions of RDF triples including millions of links between datasets [7][1]. The published datasets include data about books, movies, music, radio and television programs, reviews, scientific publications, genes, proteins, medicine, clinical trials, geographic locations, people, companies, statistical and census data, etc.

This connected dataspace presents interesting opportunities for the next generation of Web-based applications [16,15]: Data from different providers may be aggregated easily; fragmentary information from multiple sources may be integrated to achieve a more complete view. While a few applications, such as the BBC music guide[2] [18] have used Linked Data to significant benefit, the typical deployment methodology so far has been to harvest data of interest from the Web in order to create a private, disconnected repository for each specific application. This approach can only be the beginning and new concepts for consuming Linked Data are required in order to exploit the Web of Data to its full potential. The approaches, patterns, and tools necessary are very different from situations when identifiers for entities are known a priori, local, whole-repository

Olaf Hartig
Humboldt-Universität zu Berlin, Institut für Informatik
Unter den Linden 6, 10099 Berlin, Germany
Tel.: +49 30 2093-3022
Fax: +49 30 2093-3010
E-mail: hartig@informatik.hu-berlin.de

Andreas Langegger
Johannes Kepler University Linz
Altenberger Straße 69, 4040 Linz, Austria

[1] For current statistics we refer to the Wiki of the Semantic Web Interest Group at `http://esw.w3.org/topic/TaskForces/CommunityProjects/LinkingOpenData/DataSets/Statistics`.

[2] `http://www.bbc.co.uk/music`

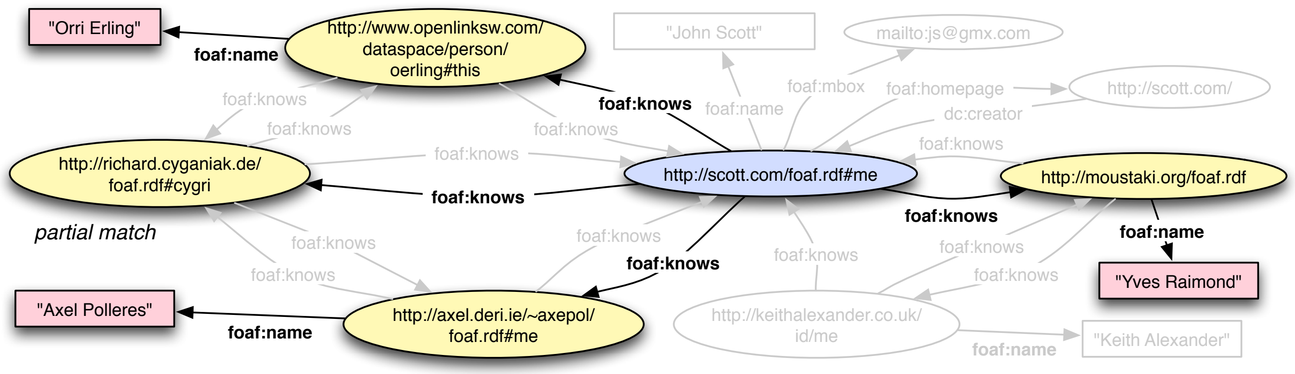

"Orri Erling"
foaf:name
http://www.openlinksw.com/dataspace/person/oerling#this
foaf:knows
"John Scott"
mailto:js@gmx.com
foaf:mbox
foaf:name
foaf:homepage
http://scott.com/
dc:creator
http://richard.cyganiak.de/foaf.rdf#cygri
http://scott.com/foaf.rdf#me
http://moustaki.org/foaf.rdf
foaf:name
"Yves Raimond"
partial match
http://axel.deri.ie/~axepol/foaf.rdf#me
"Axel Polleres"
http://keithalexander.co.uk/id/me
"Keith Alexander"

Fig. 1 Graph representation of an example RDF graph.

queries are possible, and access to the repository is reliable.

In this article we introduce the concept of Linked Data and discuss how to consume Linked Data from the Web by applying different approaches to execute structured, SQL-like queries over data from multiple providers. Due to the distributed nature as well as the openness of the Web, these query approaches face various challenges, some of which have been addressed by the database community in similar contexts. Some challenges, however, did not arise before in the context of database systems. Most of them can be attributed to a complete lack of information about data sources that might be relevant for answering a query; even the existence of potentially relevant sources might not be known in advance. Hence, to tap the full potential of the Web it is necessary to develop novel concepts for automatic discovery and on-the-fly integration. In our discussion of applicable query approaches we point out these challenges; we also bring up open issues that must be solved in the future.

The article is structured as follows: in Section 2 we introduce the foundations of the Web of Data by describing the Linked Data principles and by referring to the implemented standards and best practices. Section 4 focuses on the possibilities to adapt traditional approaches such as data warehousing and query federation to consume Linked Data; we describe how these approaches could be applied to query the Web of Data and we discuss the advantages and limitations of these applications. In addition to the traditional approaches, in Section 5 we outline novel query approaches that emerge for the Web of Data. While these approaches also allow to execute SQL-like queries over Linked Data from multiple providers they do not require to know all data sources in advance that may contribute to a query result. Finally, we summarize the content of this article and conclude with future prospects in Section 6.

2 Technical Foundations of Linked Data

More than 4 years ago Berners-Lee formulated the principles of Linked Data [3], which enable the creation of the Web of Data, as follows:

1. *Use URIs as names for things.*
2. *Use HTTP URIs so that people can look up those names.*
3. *When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).*
4. *Include links to other URIs (within this information) so that they can discover more things.*

Hence, the Web of Data is based on three (Semantic) Web technologies: the Hypertext Transfer Protocol (HTTP) [13], Uniform Resource Identifiers (URIs) [5], and the Resource Description Framework (RDF) [17]. In this section we describe how the Linked Data principles propose to use these technologies for the publication and linking of data on the Web.

The Linked Data principles require to identify an entity via a single HTTP scheme based URI. This URI does not only serve as a globally unique identifier but it also provides access to a structured data representation of the identified entity. Hence, looking up such a URI via the HTTP protocol yields data about the entity identified by the URI. According to the third principle, this data should be represented using RDF. RDF is a generic, graph based data model that represents information based on triples of the form (*subject*, *predicate*, *object*). By definition, each element of such an RDF triple can be a URI; objects can also be literal values (e.g. a string or a number) instead, and subjects and objects can also be local identifiers for unnamed entities. A set of RDF triples is called an RDF graph. Figure 1 depicts a graph representation of an example RDF graph; each RDF triple is represented by a directed edge from a vertex representing the subject to a vertex for the object.

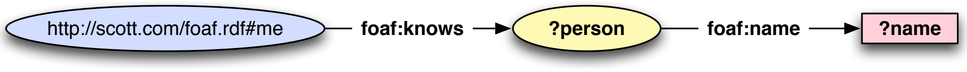


Fig. 2 Graph representation of a basic graph pattern (BGP) consisting of two triple patterns with query variables `?person` and `?name`.

?person	?name
`<http://www.openlinksw.com/dataspace/person/oerling#this>`	"Orri Erling"
`<http://moustaki.org/foaf.rdf>`	"Yves Raimond"
`<http://axel.deri.ie/~axepol/foaf.rdf#me>`	"Axel Polleres"

Table 1 Tabular presentation of solutions for the sample BGP (cf. Figure 2) in the sample RDF graph (cf. Figure 1).

The predicate URI in an RDF triple specifies the relationship between the subject and the object of the triple. For example, the triple (`<http://scott.com/foaf.rdf#me>`, `foaf:name`, "John Scott") states that the person identified by the given subject URI is called John Scott and (`<http://scott.com/foaf.rdf#me>`, `foaf:knows`, `<http://axel.deri.ie/~axepol/foaf.rdf#me>`) states that he knows another person identified by the given object URI. The semantics of predicate URIs as well as classes of entities are defined in vocabularies. The RDF Vocabulary Definition Language (RDFS) [9] and the Web Ontology Language (OWL) [31] allow users to define such vocabularies. Since RDFS and OWL allow to represent the definition of a vocabulary in the form of an RDF graph, vocabularies can also be published as Linked Data on the Web. The terms introduced in vocabularies should be identified with dereferencable HTTP URIs. This practice enables a Linked Data aware application to retrieve and utilize the definition of terms used in the currently processed data.

The Linked Data principles require to respond to a URI look-up request with an RDF graph that contains data about the entity identified by the URI; however, the principles do not prescribe what such an RDF graph should look like, what data is necessary, what vocabularies should be used, etc. Nonetheless, a common practice is to provide a set of RDF triples that contain the requested URI or that are closely related to them. However, the principles require that the provided RDF graphs should include RDF links pointing to data from other data sources on the Web. An *RDF link* is an RDF triple where the subject is a URI in the namespace of one data provider and the object is another URI in the namespace of another provider. By connecting data of different sources via RDF links the Web evolves into a platform where self-describing data of any type can be posted, consumed, and integrated in a standardized manner.

Usually, the RDF graphs that can be retrieved by looking up a URI on the Web of Data are part of a *linked dataset* which is an RDF graph that contains data about multiple entities. Typical approaches to create a linked dataset are Linked Data interfaces over native RDF stores and wrappers over relational databases or over Web APIs. Some wrappers materialize the created linked dataset, others convert the data on the fly as a response to URI look-up requests.

In addition to publishing a linked dataset following the Linked Data principles, some data providers offer an RDF dump or a SPARQL endpoint for their linked datasets. An *RDF dump* is a large RDF document that contains the RDF graph which makes up the whole linked dataset. A *SPARQL endpoint* is an HTTP-based query service that executes SPARQL queries over the linked dataset. SPARQL [27], the query language for RDF, is based on RDF graph patterns and subgraph matching.

The basic building block from which to construct more complex SPARQL query patterns is called *basic graph pattern* (BGP). A BGP is a set of triple patterns which are RDF triples that may contain query variables at the subject, predicate, and object position (cf. Figure 2). Solutions in SPARQL are defined in the context of BGP matching: Each solution is a set of variable bindings that, basically, represents a matching subgraph in the queried RDF graph. For instance, the subgraphs highlighted in Figure 1 match the sample BGP in Figure 2; the solutions in Table 1 correspond to these subgraphs. More complex query patterns are represented by SPARQL algebra expressions that process BGP solutions to calculate the result of a SPARQL query.

Applications may access Linked Data on the Web by querying the SPARQL endpoint provided for a linked dataset. While such an access may already provide the application with valuable data, this approach ignores the great potential of the Web of Data; it does not exploit the possibilities of this huge dataspace that integrates a large number of interlinked datasets. These possibilities can be achieved by the execution of complex, structured queries over data spanning multiple datasets; i.e., answers to such queries correspond to subgraphs of the union of multiple linked datasets. In the remainder of this article we discuss different approaches that enable the execution of such queries.

	data warehousing	search engines	query federation	active discovery query federation	link traversal
Universe of Discourse (UoD):	loaded data	Web of Data	known data sources	Web of Data	Web of Data
Required source interfaces:	mainly RDF dumps	arbitrary	SPARQL endpoints	SPARQL endpoints	Linked Data (look-up) interface
Access to original data:	no	no	yes	yes	yes
Supporting data structures:	indices and statistics	crawled index	statistics	(statistics)	-
Response and throughput:	fast / fast	fast / fast	slow / medium	slow / slow	medium / slow
Recall (w.r.t. UoD):	100%	<100%	100%	<100%	<100%
Precision:	100%	<100%	100%	100%	100%
Up-to-dateness:	low	medium	high	high	high

hybrid approaches

Table 2 Query processing approaches and some of their properties.

3 Classification of Query Approaches

Table 2 provides an overview of the query approaches discussed in this article, including their main properties. Distinguishing properties are: the universe of discourse, required source interface(s), access to original data, supporting data structures, response time and throughput, precision and recall, and up-to-dateness. The property *universe of discourse* refers to the entirety of all data sources that are taken into account for query answering. For traditional approaches this universe is limited to loaded data or known sources, other approaches (e.g. active discovery based query federation or link traversal based query execution) operate, in principle, over the whole Web of Data. The *required source interface(s)* property represents what type of interfaces each query approach uses to access data sources on the Web. Hence, this property indicates what interfaces a data source must provide to be considered by which query approach. Typically, each source in the Web of Data provides at least a Linked Data interface that responds to URI look-up requests. Additional interfaces with which several sources expose their linked datasets are SPARQL endpoints and RDF dumps. The property *access to original data* indicates whether a query approach directly uses the original data from sources during query execution. Query approaches that are based on local copies of data do not have this property. *Supporting data structures* are the important data structures used for each approach to facilitate efficient query execution over potentially unknown and highly dynamic sources. The properties *response time* and *throughput* characterize the query execution performance of the approaches. While response time presents the time after which the first query result is available, throughput is an indicator for the time to calculate the complete query result set that can be expected by the corresponding approach. *Recall* measures the completeness of the result set with respect to the universe of discourse; *precision* measures the number of correct query results. Finally, the property *up-to-dateness* represents how up-to-date a result returned by the corresponding query approach is (in the worst case). This property directly depends on the *access to original data* property of each approach. Based on the overview in the Table 2 the remainder of this article discusses the approaches and their properties in detail.

4 Applying Traditional Query Approaches

The database literature discusses two main classes of approaches for query answering over distributed data provided by autonomous sources: data warehousing and query federation. In this section we describe the application of these traditional approaches to the execution of complex, structured queries over multiple datasets on the Web of Data, assuming SPARQL as the query language to formulate these queries. We discuss the requirements, the necessary adaptations, and the characteristics of these applications. Furthermore, we describe the advantages and shortcomings of the approaches.

4.1 Data Warehousing

Data warehousing is an approach where data is collected and stored in a central database, referred to as a *data warehouse* [11]. Queries are executed over this central database. The idea of a data warehouse can be applied to set up a query service over a collection of

Linked Data copied from multiple sources on the Web. The most effortless approach of building such a collection is loading RDF dumps of relevant linked datasets into a large RDF store. The provenance of the datasets copied to the store may be tracked using the concept of Named Graphs [10] that are RDF graphs named by URIs. If certain datasets are not available as RDF dumps it might be feasible to crawl the data by looking up URIs or to extract the data by accessing a SPARQL endpoint. These approaches to collect data are also implemented in search engines for the Web of Data [24, 12].

The data warehousing approach offers the best performance because data is already in place and no network communication is required. The query processor may utilize central indices and statistics, benefiting from a large body of work on efficient RDF storage and querying (e.g. [1,23,32]). Hence, the data warehousing approach has fast response times and throughput (cf. Figure 2). However, implementing this approach might turn out to be a major challenge; the problems discussed by Widom [33] in the general context of data warehousing also apply to the outlined adaptation of this approach. For instance, setting up such a collection is not trivial: potentially large RDF dumps must be retrieved from the Web; load times can be significant, rendering this approach impractical for ad-hoc processing of analytical queries. Furthermore, updates to the original sources are not reflected (at least not per se) in the created, central collection. For this reason, the queried data might not be up-to-date. However, first approaches emerge that aim to address the open issue of keeping the copied datasets in sync with their original sources; e.g. Umbrich et al. [30]. In addition to application-specific implementations of data warehousing approaches the outlined effort to set up and operate a data warehouse may induce the emergence of another kind of data providers: certain parties may specialize in providing Web accessible query services over domain-specific collections of linked datasets.

4.2 Search Engines for the Web of Data

An alternative approach are the aforementioned search engines for the Web of Data such as Sindice [24] and Falcons [12]. These search engines crawl the Web by following RDF links, they index discovered data, and they provide query interfaces to their indexes. In contrast to search engines for the traditional Web, Linked Data search engines often support some special features such as triple pattern matching in addition to keyword search over literals. However, today's search engines cannot answer complete SPARQL queries.

The universe of discourse for search engines is the complete Web of Data because, in principle, data from all linked datasets on the Web may contribute to a query answer. However, since it is hardly possible to crawl and mirror the whole Web of Data, such search engines typically have a recall of less than 100% (cf. Figure 2) with respect to their universe of discourse, the Web of Data. Furthermore, these search engines, usually, have less than 100% precision because they may provide some wrong results, similar to traditional search engines. While indexed data is re-crawled regularly it might, nonetheless, be dated upon query time because data sources may update their data more frequently than a search engine re-crawls the data.

4.3 Query Federation

In contrast to data warehousing and search engines where queries against the central store can be executed in the same way as in a local database system, the query federation approach is based on distributing the processing of queries to multiple, autonomous sources [29]. Instead of copying data into a central store, a mediator, usually called *query federator*, analyzes and decomposes the user query into several sub-queries. These sub-queries are distributed to autonomous data sources which, then, execute these sub-queries and return the results. Hence, sub-queries and intermediate results are transmitted over the network at query time. This approach has significant benefit, since there is no need to synchronize copied data and no additional storage space is required. The downside is, that query execution is much slower because data has to be transmitted via the network on demand. Especially, when many distributed joins are involved, the performance of a query federation approach may significantly degrade compared to a data warehouse. As a consequence, additional techniques such as caching and specialized optimization algorithms are needed in order to compensate these shortcomings. Similar to the data warehousing approach, query federation provides complete, exact matches (i.e. 100% recall and 100% precision) with a universe of discourse constrained to a known set of data sources.

A significant amount of research has already been done in the area of federated database systems [29] and distributed query processing [19]. Although the algorithms are typically based on the relational algebra, they can by adjusted for SPARQL query processing over the Web of Data, assuming each relevant linked dataset is exposed via a SPARQL endpoint. The analogies between SPARQL graph pattern matching and the relational algebra have been discussed by Pérez et al. [25]. Prud'hommeaux [26] proposes a first approach for federated SPARQL query processing in the context of a biomedical application. Quilitz and Leser [28] present a generic implementation called DARQ, which is based on the popular query processor Jena ARQ. Another generic approach, proposed by Langegger [20], makes use of statistical RDF data summaries to facilitate data source selection and improve query optimization.

Due to the nature of RDF graphs which are represented by a set of triples, SPARQL query processing typically involves a great number of join operations (one for each triple pattern in the worst case). In case of query federation which understands an RDF graph to be distributed among several data sources, a query optimizer requires a-priori knowledge about the content of the distributed source graphs. The overall goal of the federator is to include only local subplans for data sources that are relevant to a query (i.e. which are known to contribute any intermediate results). In case of traditional federated database systems this information is obtained from the schema information stored in the database catalog. In contrast to a relational database, a linked dataset has no fixed schema; it can contain arbitrary predicates. A possible solution is the notion of so-called capabilities as proposed for DARQ [28]. These capabilities are, for example, selectivity constants for all RDF predicates occurring in each dataset. Based on these constants, the federator is able to determine relevant data sources and optimize joins based on selectivities. While DARQ uses simple predicate selectivities that are described explicitly in a configuration file, Langegger presents an approach that is based on extensive RDF statistics, including histograms [20]. These statistics are generated automatically and they can be requested via an extension to the SPARQL protocol. In contrast to DARQ, which supports query patterns with bound predicates only, the latter approach supports all forms of SPARQL query with all kinds of query patterns.

5 Novel Query Approaches for the Web of Data

The traditional query approaches discussed in the previous section require to know and to select potentially relevant data sources in advance. By selecting these sources an application developer restricts her application to data from the selected sources. Due to this restriction the application cannot tap the full potential of the Web; the restriction prevents a serendipitous discovery and utilization of relevant data from unknown sources. However, the fundamental assumption of all query approaches proposed and studied by the database community so far is the knowledge of the existence of data sources that may contribute to a query result. In this section we discuss two approaches, *active discovery based query federation* and *link traversal based query execution*, which are not based on this fundamental assumption and, thus, go beyond existing research. Hence, the universe of discourse for these approaches is the whole Web of Data.

5.1 Active Discovery based Query Federation

A possible approach to overcome the problem of ignoring unknown data sources would be a combination of federated query processing with an active discovery of unknown but potentially relevant linked datasets that are exposed via SPARQL endpoints. In addition to the known sources, the query federator may take these discovered endpoints into account when it distributes the sub-queries. While we are not aware of any system that implements this idea we consider it worth investigating; it may combine the advantageous properties of query federation (e.g. no need to collect and manage data locally, up to date query results) with the possibility to determine more complete results by benefiting from additional data of formerly unknown sources. Hence, taking the complete Web as the universe of discourse such a combined approach, which we call *active discovery based query federation*, may have a higher recall than the traditional, more static query federation approaches introduced in Section 4.3. However, even if the recall might be higher it also might be difficult to reach 100%, considering it is impossible or impractical to discover and to integrate all linked datasets on the Web that are potentially relevant for the current query.

While we are not aware of any active discovery based query federation system we only outline ideas and requirements for the implementation of this approach. The discovery of additional data sources might either happen on the fly on a per-query basis so that additional SPARQL endpoints are searched for that may answer certain sub-queries of the current user query. Alternatively, the discovery process could be separated from the query execution in order to pro-actively search for further endpoints; this search could be based on an analysis of the last seen queries. In both cases the actual search for additional SPARQL endpoints may utilize a search engine or another type of central repository, or it could be based on an approach that automatically follows RDF links similar to the link traversal based query approach discussed in the next Section 5.2. To decide about the relevancy of SPARQL endpoints it requires descriptions of the capabilities of an endpoint and of the content of linked datasets exposed via such an endpoint. While these descriptions could be published as Linked Data as well, there are no agreed-upon vocabularies to describe this meta-data, yet. Nonetheless, there are first approaches such as the vocabulary of interlinked datasets [2], the service description vocabulary for SPARQL [34], and the RDFstats vocabulary [21].

The dynamic nature of the active discovery based query federation approach also poses new challenges to query planning and optimization. While the discovered SPARQL endpoints provide access to additional datasets, the characteristics and selectivities of these datasets are unknown to the federator. Obtaining such knowledge might be difficult or even impossible. However, given this knowledge has been obtained, it could be used to adjust the query plan during runtime by generating and distributing additional sub-queries to newly discovered endpoints. With the integration of such endpoints the initial query plan may become inefficient, therefore an adaptation of the plan appears reasonable. While the idea of adaptive query processing is not new, active discovery based query federation presents a new scenario that could benefit from existing approaches or that may even require novel concepts for query plan adaptation.

5.2 Link Traversal based Query Execution

While active discovery based query federation is an augmentation of the idea of federated databases, *link traversal based query execution* [14] introduces a novel query execution paradigm that is significantly different from existing approaches.

The approach is inspired by an algorithm which follows RDF links in order to obtain more data about the entities that are presented in the Tabulator Linked Data browser [4]. This idea, which we call *automated link traversal*, exemplifies the enormous potential of the Web of Data because it allows to use data that is serendipitously discovered from formerly unknown sources. Consequently, link traversal based query execution applies the idea of automated link traversal to the execution of SPARQL queries over the Web of Data. Hence, in contrast to federation based approaches that are restricted to linked datasets exposed via a SPARQL endpoint, the link traversal approach requires only an adherence to the Linked Data principles.

The general idea of this query approach is to intertwine query pattern matching over a continuously growing dataset with the traversal of links in order to discover data that might be relevant to answer the executed query. By using the data retrieved from looking up the URIs in a query as a starting point, the query engine evaluates parts (e.g. certain triple patterns) of the query. The intermediate solutions resulting from this partial evaluation usually contain further URIs. These URIs link to additional data which may provide further, intermediate solutions for the same or for other parts of the query. To determine results for the whole query the query engine alternately evaluates query parts and looks up URIs as the following example illustrates:

Example 1 Consider a Linked Data based application that allows its users to learn about software projects, their relationships such as required software packages, the developers, etc. Such an application may benefit from employing the link traversal based approach: Answers to queries require data from multiple providers, published at different locations on the Web such as different project Web sites and developers' online profiles. The application may issue SPARQL queries as illustrated in Figure 3. This query asks for software created by the creator(s) of the software identified by URI `http://.../sw1`. Link traversal based query execution typically starts with an empty queried dataset. A query engine that implements the link traversal based approach, first, obtains some seed data for pattern matching by looking up URIs in the query: for the URI `http://.../sw1` in the sample query the engine may retrieve RDF graph g_{sw1} (cf. Figure 4). The engine adds this graph to the queried dataset. Now, the engine alternates between i) constructing intermediate solutions from RDF triples that match a pattern of the query in the queried dataset and ii) augmenting the dataset by looking up URIs which are part of these intermediate solutions. After looking up `http://.../sw1` the queried dataset contains an RDF triple that matches the first triple pattern in the sample query (line 2 of Fig-

```
SELECT ?name WHERE {
   <http://.../sw1> dc:creator ?d .
   ?other dc:creator ?d .
   ?other rdfs:label ?name .
}
```

Fig. 3 A SPARQL query that asks for things created by the creator(s) of the software identified by URI `http://.../sw1`.

Excerpt from RDF graph g_{sw1} retrieved for `<http://.../sw1>` :

```
<http://.../sw1> rdfs:label "XY Lib" .
<http://.../sw1> dc:creator <http://.../d1> .
<http://.../sw3> sw:requires <http://.../sw1> .
```

Excerpt from RDF graph g_{d1} retrieved for `<http://.../d1>` :

```
<http://.../sw1> dc:creator <http://.../d1> .
<http://.../sw2> dc:creator <http://.../d1> .
```

Excerpt from RDF graph g_{sw2} retrieved for `<http://.../sw2>` :

```
<http://.../sw2> rdfs:label "AB app" .
<http://.../sw2> sw:requires <http://.../sw1> .
```

Fig. 4 Excerpts from RDF graphs retrieved during the link traversal based execution of the sample query in Figure 3.

ure 3). From this matching triple the query engine constructs an intermediate solution μ_1 (cf. Table 3). Similarly, the engine may construct μ_2 and μ_3 for the second and the third triple pattern (lines 3 and 4), respectively. While μ_2 does not refer to a new URI, μ_1 contains the previously undiscovered URI `http://.../d1` (so does μ_3). By looking up this URI the query engine retrieves RDF graph g_{d1} and adds it to the queried dataset. After this addition the queried dataset contains another matching triple for the second pattern, from which the query engine constructs intermediate solution μ_4. Notice, constructing μ_4 is only possible because the query engine discovered g_{d1} based on μ_1. The engine proceed with the execution strategy: It looks up `http://.../sw2`, retrieves RDF graph g_{sw2}, and constructs μ_5. Finally, the query engine can merge intermediate solutions that are compatible to create two results for the whole query: "XY Lib" and "AB app".

	?d	?other	?name
μ_1	`<http://.../d1>`	*unbound*	*unbound*
μ_2	*unbound*	`<http://.../sw1>`	“XY Lib”
μ_3	`<http://.../d1>`	`<http://.../sw1>`	*unbound*
μ_4	`<http://.../d1>`	`<http://.../sw2>`	*unbound*
μ_5	*unbound*	`<http://.../sw2>`	“AB App”

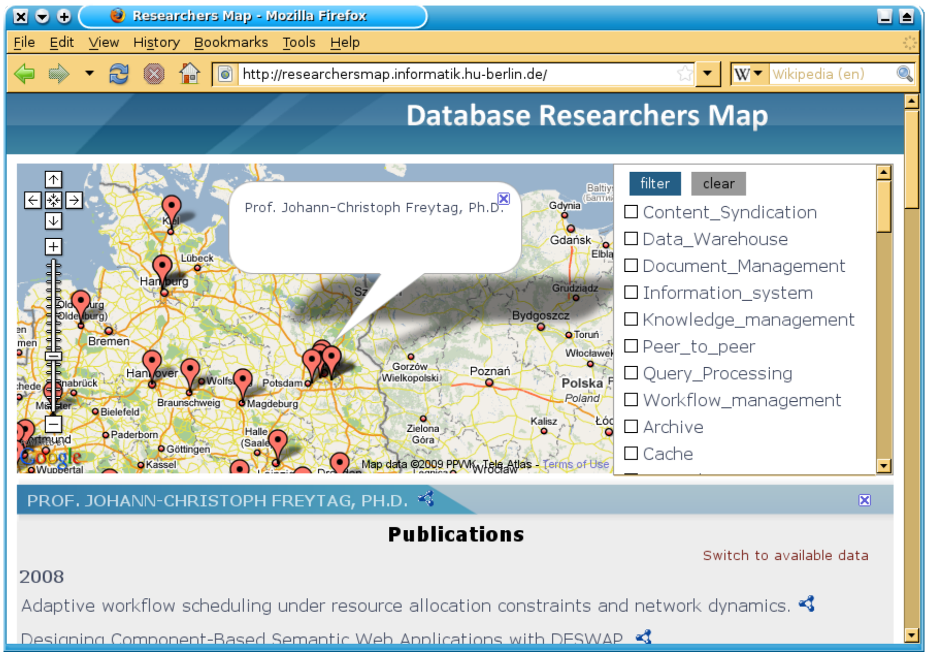
Table 3 Intermediate solutions as generated during the link traversal based execution of the query in Figure 3.

As the example demonstrates, the link traversal based approach proposes to evaluate a query over a dataset that is continuously augmented with potentially relevant data from the Web. The discovery of this data is driven by the URIs in intermediate solutions. Usually, not all the URIs looked up during query execution are in the same namespace, controlled by a single data provider. Instead, since the Linked Data principles require to provide RDF links to data from other sources it is likely that data from multiple providers is discovered and used to construct query results; this may include providers the query engine did not even know they exist before executing the query. Hence, link traversal based query execution is indeed different from traditional approaches which require at least information of the existence of data sources that may contribute to the query execution.

In earlier work Mendelzon and Milo introduce a two-phase approach that includes the traversal of links to execute relational queries on the traditional, document-centric Web: First, all “reachable documents are retrieved [by traversing all discoverable links], and then the query is evaluated on them.” [22] A similar approach has been formalized by Bouquet et al. for the Web of Data [8]. Link traversal based query execution differs from these two-phase approaches because it intertwines query evaluation with the traversal of links. Instead of following arbitrary links before evaluating the query, a query engine that implements the link traversal based approach follows only those links that turn out to be potentially relevant during query execution, i.e., those that correspond to triple patterns in the executed query. Hence, in contrast to Mendelzon and Milo and Bouquet et al. link traversal based query execution makes use of the fact that each link has a type, due to the use of RDF.

Due to its novelty the link traversal based approach poses several problems and exciting new research challenges. Due to the openness and the widely distributed nature of the Web we cannot assume to find all data that is relevant to answer a query. Hence, we should never expect results that are complete w.r.t. all data in the Web. Nonetheless, strategies are required that either allow for results as complete as possible or that report the first result(s) as early as possible; an alternative criterium could be to find the majority of the discoverable results as efficiently as possible. Efficiency, in general, is a challenge. For instance, looking up URIs on the Web causes delays during query execution that may have a significant negative impact on query execution times. For an iterator-based implementation of the link traversal based query execution paradigm, Hartig et al. [14] propose concepts to reduce the impact of these delays. Further interesting extensions of link traversal based query execution are a combination with in advance crawling or with the re-use of the queried dataset for multiple, subsequent queries. In both approaches query execution starts using data that is already available, which may improve result completeness and reduce query execution costs. A combination of these techniques with the query approach introduces the need for a proper caching solution with suitable replacement strategies and refreshing policies.

A first application that is built using a prototypical query engine implementing the link traversal based query execution paradigm is Researchers Map [15]. This application is a Linked Data based mash-up that provides a map of professors from the German database community; the list of professors in the map can be filtered by research interests; selecting a professor opens a list of her/his publications (cf. Figure 5). This mash-up is constructed solely based on the results of queries executed over the Web of Data. Due to its commitment to Linked Data, Researchers Map is a new kind of applications that enable users to be in full charge of their data. For instance, all that is needed to appear in the Researchers Map is a URI that identifies a professor and



2008

Adaptive workflow scheduling under resource allocation constraints and network dynamics.

Designing Component-Based Semantic Web Applications with DESWAP

Fig. 5 Screenshot of the Linked Data based mash-up *Researchers Map* that provides a map of professors (i.e. their workplaces) from the German database community.

links to the data provided by him. Hence, users can retain complete control over the authoritative source of data provided by them. Notice, Linked Data is the only option today to provide this freedom. Researchers Map illustrates the benefits of exploiting the potential of the Web of Data and it demonstrates the suitability of link traversal based query execution for applications.

6 Conclusions

In this article we give an overview on approaches for querying Linked Data. We discuss the advantages and disadvantages of these approaches; and, we point out open research problems in this context. Notice, while we presented the different approaches separately they can also be combined into hybrid approaches. For instance, it might be promising to combine the execution of queries over a fixed set of base datasets in a data warehouse with additional data retrieved during run-time by the application of a link traversal based approach.

The execution of queries as presented in this article is only part of the wider area of consuming Linked Data from the Web. To achieve the ultimate goal of using the Web of Data as if it is a giant, globally distributed database, further challenges and remaining issues must be addressed. These challenges include the seamless integration of Linked Data from multiple providers: to employ the query approaches discussed in this article it requires mappings between terms from different vocabularies used by data sources with similar content. Furthermore, it may be necessary to apply data fusion techniques to achieve a consistent and clean representation of data that different sources represent differently. Data fusion may also help to resolve data conflicts and data quality issues. Information quality, in general, is a challenge. Linked Data consuming applications such as query engines should assess the quality of each data item before making use of it. In particular, the issue of trustworthiness of data must play a central role in future research because on the Web everyone can publish anything. Hence, every RDF triple published on the Web can only be conceived as a claim from the corresponding data provider (instead of a fact). Further interesting research in the context of Linked Data consumption may study reasoning and knowledge discovery in data from multiple providers as well as end user interfaces to interact with the Web of Data.

Working on the approaches and issues introduced in this article will help to take advantage of the enormous potential provided by the evolution of the Web into a globally distributed space of Linked Data. This evolutionary step could enable users to benefit from a virtually unbound set of data sources. Hence, questions of consuming Linked Data present an exciting new research area that may also provide new challenges for the database community.

References

1. D. J. Abadi, A. Marcus, S. R. Madden, and K. Hollenbach. Sw-store: A vertically partitioned dbms for semantic web data management. *The VLDB Journal*, 18(2), Feb. 2009.

2. K. Alexander, R. Cyganiak, M. Hausenblas, and J. Zhao. Describing linked datasets. In *Proceedings of the 2nd Linked Data on the Web Workshop (LDOW) at WWW*, 2009.
3. T. Berners-Lee. Design Issues: Linked Data. Online at `http://www.w3.org/DesignIssues/LinkedData.html`.
4. T. Berners-Lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets. Tabulator: Exploring and analyzing lnked data on the semantic web. In *Proceedings of the 3rd Semantic Web User Interaction Workshop (SWUI) at ISWC*, Nov. 2006.
5. T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifier (URI): Generic syntax. RFC 3986, 2005.
6. C. Bizer, T. Heath, D. Ayers, and Y. Raymond. Linking open data. In *Proceedings of the Poster Session at the 4th European Semantic Web Conference*, 2007.
7. C. Bizer, T. Heath, and T. Berners-Lee. Linked data – the story so far. *International Journal on Semantic Web and Information Systems (IJSWIS), Special Issue on Linked Data*, 5(3):1–22, 2009.
8. P. Bouquet, C. Ghidini, and L. Serafini. Querying the web of data: A formal approach. In *Proceedings of the 4th Asian Semantic Web Conference (ASWC)*, 2009.
9. D. Brickley and R. V. Guha. RDF vocabulary description language 1.0: RDF schema. W3C Recommendation, Feb. 2004. Online at `http://www.w3.org/TR/rdf-schema/`.
10. J. J. Carroll, C. Bizer, P. J. Hayes, and P. Stickler. Named graphs. *Journal of Web Semantics*, 3(4):247–267, 2005.
11. S. Chaudhuri and U. Dayal. An overview of data warehousing and olap technology. *SIGMOD Record*, 26(1):65–74, 1997.
12. G. Cheng and Y. Qu. Searching linked objects with falcons: Approach, implementation and evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS), Special Issue on Linked Data*, 5(3):49–70, 2009.
13. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol – HTTP/1.1. RFC 2616, 1999.
14. O. Hartig, C. Bizer, and J.-C. Freytag. Executing SPARQL queries over the web of linked data. In *Proceedings of the 8th International Semantic Web Conference (ISWC)*, Nov. 2009.
15. O. Hartig, H. Mühleisen, and J.-C. Freytag. Linked data for building a map of researchers. In *Proceedings of 5th Workshop on Scripting and Development for the Semantic Web (SFSW) at ESWC*, June 2009.
16. T. Heath. How will we interact with the web of data? *IEEE Internet Computing*, 12(5):88–91, Sept. 2008.
17. G. Klyne and J. J. Carroll. Resource description framework (RDF): Concepts and abstract syntax. W3C Recommendation, Feb. 2004. Online at `http://www.w3.org/TR/rdf-concepts/`.
18. G. Kobilarov, T. Scott, Y. Raimond, S. Oliver, C. Sizemore, M. Smethurst, C. Bizer, and R. Lee. Media meets semantic web – how the bbc uses dbpedia and linked data to make connections. In *Proceedings of the 6th European Semantic Web Conference (ESWC)*, June 2009.
19. D. Kossmann. The State of the Art in Distributed Query Processing. *ACM Computing Surveys*, 32(4):422–469, 2000.
20. A. Langegger. *A Flexible Architecture for Virtual Information Integration based on Semantic Web Concepts*. Dissertation, J. Kepler University Linz, January 2010.
21. A. Langegger and W. Wöß. RDFStats – an extensible rdf statistics generator and library. In *Proceedings of the International Workshop on Database and Expert Systems Applications (DEXA)*, 2009.
22. A. O. Mendelzon and T. Milo. Formal models of web queries. *Information Systems*, 23(8):615–637, 1998.
23. T. Neumann and G. Weikum. Rdf-3x: a risc-style engine for rdf. In *Proceedings of the 34th International Conference on Very Large Data Bases (VLDB)*, 2008.
24. E. Oren, R. Delbru, M. Catasta, R. Cyganiak, H. Stenzhorn, and G. Tummarello. Sindice.com: A document-oriented lookup index for open linked data. *International Journal of Metadata, Semantics and Ontologies*, 3(1), 2008.
25. J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of sparql. In I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *5th International Semantic Web Conference, Athens, USA,*, volume 4273 of *LNCS*, Berlin, Heidelberg, 2006. Springer.
26. E. Prud'hommeaux. Case study: Federate for drug research. Online at `http://www.w3.org/2004/10/04-pharmaFederate/`, 2007.
27. E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF. W3C Recommendation, Jan. 2008. Online at `http://www.w3.org/TR/rdf-sparql-query/`.
28. B. Quilitz and U. Leser. Querying distributed RDF data sources with SPARQL. In *Proceedings of the 5th European Semantic Web Conference (ESWC)*, volume 5021 of *Lecture Notes in Computer Science*, pages 524–538. Springer Verlag, June 2008.
29. A. P. Sheth and J. A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22(3):183–236, Sept. 1990.
30. J. Umbrich, M. Hausenblas, A. Hogan, A. Polleres, and S. Decker. Towards dataset dynamics: Change frequency of linked open data sources. In *Proceedings of the 3rd Linked Data on the Web Workshop (LDOW) located at the 19th International World Wide Web Conference (WWW)*, 2010.
31. W3C OWL Working Group. OWL 2 web ontology language – document overview. W3C Recommendation, Oct. 2009. Online at `http://www.w3.org/TR/owl-overview`.
32. C. Weiss, P. Karras, and A. Bernstein. Hexastore: Sextuple indexing for semantic web data management. In *Proceedings of the 34th International Conference on Very Large Data Bases (VLDB)*, 2008.
33. J. Widom. Research problems in data warehousing. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, pages 25–30, 1995.
34. G. T. Williams. SPARQL 1.1 service description. W3C Working Draft, Jan. 2010. Online at `http://www.w3.org/TR/sparql11-service-description/`.