Efficient, Scalable, and Provenance-Aware Management of Linked Data

Marcin Wylot

1 Motivation and objectives of the research

The proliferation of heterogeneous Linked Data on the Web requires data management systems to constantly improve their scalability and efficiency. Despite recent advances in distributed Linked Data management, efficiently processing large-amounts of Linked Data in a scalable way is still very challenging. In spite of their seemingly simple data models, Linked Data actually encode rich and complex RDF graphs mixing both instance and schema-level data. At the same time, users are increasingly interested in investigating or visualizing large collections of online data by performing complex analytic queries. The heterogeneity of Linked Data on the Web also poses new challenges to database systems. The capacity to store, track, and query provenance data is becoming a pivotal feature of Linked Data management systems. In this thesis, we tackle a number of key problems relating to the efficient, scalable and provenance-aware processing of Linked Data.

The size of Linked Data is steadily growing, thus a modern Linked Data management system has to be able to deal with increasing amounts of data. However, in a Linked Data context, variety is also a salient property. Since Linked Data is often schema-free (schemas are typically not strictly enforced), standard databases techniques cannot be adopted to manage the data. Even though organizing Linked Data in a form of a table is possible, querying such a giant triple table becomes very costly due to the multiple nested joins required. Moreover, Linked Data often comes from multiple sources and can be produced in various ways for any given scenario. Heterogeneous data can incorporate knowledge on provenance, which can be further leveraged to provide users with a reliable and understandable description of the way the query was answered, that is, the way the answer was derived. Furthermore, it can enable a user to tailor queries with provenance data, including or excluding data of specific lineage (i.e., described in a systematic way).

We divide the problem we tackle into three sub-problems. Hence, we investigate three fundamental research questions:

(Q1) How to efficiently store and query vast amounts of Linked Data in the cloud?
The Linked Data community is still missing efficient and scalable data infrastructures. New kinds of data and queries (e.g., unstructured and heterogeneous data, graph and analytic queries) cannot be efficiently handled by legacy systems. Small Linked Data graphs can be handled in-memory or by standard database systems. However, Big Linked Data with which we deal nowadays are singularly hard to manage. Modern Linked Data management systems have to face vast amounts of heterogeneous, inconsistent, and schema-free data. We analyze and evaluate in this context several well-known Linked Data management systems. We describe their

strong and weak points, and we show that the current approaches are overall suboptimal. Following that, we propose our own distributed Linked Data management system.

(Q2) How to store and track provenance in Linked Data processing? Within the Web community, there have been several efforts to develop models and syntaxes to interchange provenance, which resulted in the recent W3C PROV recommendation. However, less attention has been given to the efficient handling of provenance data within Linked Data management systems. While some systems store quadruples or named graphs, to the best of our knowledge, no current high-performance triple store is able to automatically derive provenance data for the results it produces. We present in that sense new approaches to store and track provenance.

(Q3) What is the most effective query execution strategy for provenance-enabled queries?
With the heterogeneity of Linked Data, users may want to tailor their queries based on provenance specifications, e.g., "find me all the information about Paris, but exclude all data coming from commercial websites". To support such use-cases, the most common mechanism used within Linked Data management systems is named graphs. This mechanism was recently standardized in RDF 1.1. Named graphs associate a set of triples with a URI. Using this URI, metadata including provenance can be associated with the graph. While named graphs are often used for provenance, they are also used for other purposes, for example to track access control information. Thus, while Linked Data management systems (i.e., triple stores) support named graphs, there has only been a relatively small number of approaches specifically focusing on provenance within the triple store itself and much of it has been focused on theoretical aspects of the problem. We describe in that context our methods and implementation to handle provenance-aware workloads.

2 Description of the approach

To answer the aforementioned research questions, we propose different techniques to store and process Linked Data. We divide them into three parts: storing and querying Linked Data in the cloud, storing and tracking provenance in Linked Data, and querying over provenance data.

The first part addresses the problem of efficient storage of Linked Data (Research Question **Q1**); we propose a novel hybrid storage model considering Linked Data both from a graph perspective (by storing molecules[1]) and from a "vertical" analytics perspective (by storing compact lists of literal values for a given attribute). Our molecule-based storage model allows to efficiently partition data in the cloud such as to minimize the number of expensive distributed operations (e.g., joins). Contrary to previous approaches, our techniques perform an analysis of both instance and schema information prior to partitioning the data. We extract graph patterns from the data and combine them with workload information in order to find effective ways of co-locating, partitioning and allocating data on clusters of commodity machines.

[1] molecules are similar in their simplest form to property tables and store, for each subject, the list or properties and objects related to that subject

Our approaches enable efficient and scalable distributed Linked Data management in the cloud, and support both transactional and analytic queries efficiently. We also propose efficient query execution strategies leveraging our compact storage model and taking advantage of advanced data co-location strategies enabling us to execute most of the operations fully in parallel. Specifically, we make the following contributions:

- a new data partitioning algorithm to efficiently and effectively partition the graph and co-locate related instances in the same partitions;
- a new system architecture for handling fine-grained Linked Data partitions at scale;
- novel data placement techniques to co-locate semantically related pieces of data;
- new data loading and query execution strategies taking advantage of our system's data partitions and indices;
- an extensive experimental evaluation showing that our methods are often two orders of magnitude faster than state-of-the-art systems on standard workloads.

In the second part of this thesis, we present techniques supporting the transparent and automatic derivation of detailed provenance information from arbitrary queries (Research Question **Q2**). We introduce new physical models to store provenance data and several new query execution strategies to derive provenance information. We make the following contributions in that context:

- a new way to express the provenance of query results at two different granularity levels by leveraging the concept of provenance polynomials[2];
- two new storage models to represent provenance data in a native Linked Data store compactly, along with query execution strategies to derive the aforementioned provenance polynomials while executing the queries;
- a performance analysis of our techniques through a series of empirical experiments using two different Web-centric datasets and workloads.

In the third part of this thesis, we investigate how Linked Data management systems can effectively support queries that specifically target provenance, that it, provenance-enabled queries (Research Question **Q3**). To address this problem, we propose different provenance-aware query execution strategies and test their performance with respect to our provenance-aware storage models and advanced co-location strategies. We make the following contributions in that context:

- a characterization of provenance-enabled queries (queries tailored with provenance data);
- five provenance-oriented query execution strategies;
- storage models and indexing techniques extensions to handle provenance-aware query execution strategies;
- an experimental evaluation of our query execution strategies and an extensive analysis of the datasets used for the experimental evaluation in the context of provenance data.

[2] provenance polynomials are algebraic structures representing how the data is combined to derive query answers using different relational algebra operators (e.g., UNION, JOINS)

3 Major results

Our experimental evaluations show that our new molecule-based storage model represents an optimal way of co-locating Linked Data in a very compact manner, resulting in excellent performance when executing both transactional and analytic queries in the cloud and in orders of magnitude improvement over state-of-the-art systems. Moreover, when extended to store provenance data, it remains efficient from a storage consumption perspective, and allows time-efficient tracing of the queries' lineage. Finally, evaluating the presented provenance-aware techniques, we show that because provenance is prevalent within Linked Data and is highly selective, it can be used to *improve* query processing performance, which is a counterintuitive result as provenance is often associated with additional overhead.

4 Description of the evaluation methods used to validate the results

To empirically evaluate our approach, we implemented the storage models and query execution strategies described in this thesis in both centralized and distributed models. We evaluated them against various data collections and workloads and we compared them with state-of-the-art systems and techniques. The datasets and queries we used, as well as our own source code, were published with the corresponding papers.

To evaluate the performance of out methods, we used a series of datasets and benchmarks, including:

- Billion Triples Challenge (BTC) data
- the Web Data Commons (WDC)
- the DBPedia dataset
- the Lehigh University Benchmark (LUBM)
- the BowlognaBench Benchmark.

We compared the runtime execution for various queries encompassing different kinds of typical query patterns, including star-queries of different sizes and up to 5 joins, object-object joins, object-subject joins, and triangular joins. In addition, we included analytic queries and queries with UNION and OPTIONAL clauses. We also took advantage of further standard metrics to empirically evaluate our techniques, including loading time, size of the resulting primary or secondary indices, amount of main-memory used, time to pre-materialize views, or number of (distributed) joins.

5 Significance of the work, open issues, and future directions of work

The thesis makes a number of important contributions in Linked Data management. We tackle a series of long-lasting, key technical issues relating to RDF and Linked Data, which have been slowing down its adoption in industry and large-scale efforts:

1. how to efficiently process analytic queries over Big Linked Data. As we show in our experiments, even relatively simple analytics can take several minutes

on state-of-the-art systems, which is unacceptable in most environments where users or further processes are waiting on the answers; instead, we devise new physical storage structures and indices to speed up both transactional queries and analytics on large RDF graphs;

2. how to effectively distribute RDF datasets over clusters of commodity machines; this is an important problem in today's environments where *scaling-out* is the only option for very large datasets. As we empirically demonstrate, naive partitioning schemes like min-cut yield extremely poor results in practice; instead, we introduce dedicated molecule-based partitioning schemes that take into account the peculiarities of the RDF graph;
3. how to compactly store provenance information to track the lineage of the query results, which is crucial for Linked Data regrouping information authored by several sources. Simply storing the provenance of the triples using quads is prohibitive both in terms of storage and processing overhead; instead, we investigate several storage models and discuss their strengths and weaknesses in detail;
4. how to tailor workload queries using provenance queries, and how to jointly and efficiently process both kinds of queries through various query optimizations.

This work also presents a number of clear tradeoffs and technical choices. We often trade space for performance (e.g., when storing both molecules and vertical structures at the physical level); we believe that it this the right way to go in today's environment where space is typically abundant. However, this also implies a higher complexity for a number of operations, like out-of-place updates, as we discuss in the thesis. Also, handling provenance information during query execution imposes some clear overhead, which can however be mitigated thanks to the increased selectivity of the provenance queries as we show in our extensive evaluation.

The presented work can be extended in several important directions. In our work, we leveraged templates defined by the types of the various resources and identified an interesting problem in automatic templates discovery based on frequent patterns for *untyped* elements. Despite the fact that Linked Data is generally schema-free, it tends to exhibit frequent patterns which allow to reconstruct a reliable schema for the considered data. Such emerging schema templates could be leveraged to cluster the molecules. In addition, one could investigate dynamic storage models to enable further optimization for memory consumption and query execution. In this context, we could leverage techniques borrowed from machine learning to decide which templates could be extended to higher scopes in order to improve performance. In terms of provenance handling, one could extend our current implementation to output PROV, which would open the door to queries over the provenance of the query results and the data itself – merging both internal and external provenance. Another interesting question worth investigating is whether provenance can be leveraged to partition Linked Data in the cloud e.g., if molecules sharing the same provenance should be co-located on one node to eliminate distributed operations.