Retrieval of the most relevant combinations of data published in heterogeneous distributed datasets on the Web*

Shima Zahmatkesh

DEIB - Politecnico di Milano, Milan, Italy
shima.zahmatkesh@polimi.it

Abstract. Finding the most relevant data items among heterogeneous data published on the Web is getting a growing attention in recent years. Retrieving the most relevant data items from a collection of data is a challenge addressed by top-k databases. Accessing heterogeneous and distributed data sources is a challenge addressed by the Semantic Web. How to combine methods and techniques from those two fields is still an open research issue. This doctoral thesis will investigate how the presence of an ontology describing an integrated conceptual model of the data sources and the possibility to encode the users' information needs in top-k queries can make the query answering process faster, more efficient, and able to get more relevant results.
Keywords: Top-k query, Federated databases, heterogeneous data, OBDA, SPARQL, Query Optimization.

1 Relevancy

While a massive amount of data is getting published on the web, searching for data is also attracting a growing attention. Notably, most of the time, users try to satisfy their information needs integrating the results of multiple (vertical) search engines. Those users expect relevant answers to appear in the few first pages of the results, are sensible to correctness, but are rarely interested in completeness. As an example, imagine a student who may want to find the best university for studying; he would take in to account various criteria. He is certainly interested in finding information about the university such as its ranking and the quality of education program. However, he is also concerned with: the quality of life of the city where the university is located in, the public transportation in that city, and the possibility to find cheep accommodation. What he would be satisfied to collect is a set of resources that, once integrated, answer his information need.

2 Problem Statement

The problem that I want to address in this work is how to quickly find on the Web the most relevant answers to queries that span multiple domains and that include user preferences.

*This research is developed under the supervision of Professor Emanuele Della Valle.

The nature of the Web implies that the answers have to be found in a multitude of structured and unstructured information, stored in heterogeneous formats across multiple, distributed and possibly overlapping data sources.

Most important for my research work is the way that data is accessed. At a first glance it seems that it is easy to access data in the Web, as all the resources in the Web have a URL. However, in most of the cases, the URLs of the desired resources are unknown. So, usually, users employ search engines or search services to find those "unknown" URLs. It is worth to note that search engines provide only *sorted access* to result items that are return as a ranked list where more relevant results appear first. Note that *random access* to results in those ranked lists is not possible. For example, let assume to search the same term in two search engines (A and B), a user cannot know which is the position of a specific result A in the results of B. To cross check the results of two search engines one has to sequentially read the results of the two search engines.

Moreover, request for (*random*) access to a data resource over the web is more expensive than on a hard disk due to delays introduced by network transmission of the data and the overhead introduced by the usage of the HTTP protocol. Even the request for large amount of data could be expensive because of long transmission times and of protracted processing of the service.

Last but not least, accessing to data resources can be challenging and complex in the case that data is distributed over heterogeneous sources. Structured data can be in relational, XML or RDF formats that can be accessed using SQL, xQuery, SPARQL and, more and more often, Web APIs. Unstructured data like text and multimedia content are even more challenging due to lack of common standards for accessing search services.

The problem I intend to address is how to improve the retrieval of the most relevant combinations of data from a variety of distributed data sources published on the Web caring about the query latency (of the first results), which must be under-second, and relevancy of the first results, which really matters for the users, without posing too much emphasis on completeness, which has little importance in the considered application case. Resource consumption is another important metric in the problem space, because the solution has to scale to thousands of concurrent users as current search engines do.

3 State-of-the-Art[1]

Current search engines do not address this problem; they have just started to offer structured query answering (e.g., Google Knowledge Graph or Wolfram Alpha). Methods for top-k query answering in databases can quickly answer queries, which requires relevant answer first, but they do not scale to the amount of resources published on the Web and cannot deal with data heterogeneity. On the contrary, semantic technologies are able to deal with data heterogeneity. In particular, OBDA uses ontology as a conceptual integrated model for representing the schema of multiple databases and allow issuing federated queries against

[1]More detailed analysis of related work follows in Section 5

a set of heterogeneous data sources. But, semantic technologies are still not optimized to find the most relevant answer first. In the Semantic Web community, approaches to retrieve most relevant data resources are still using the naïve *materialize then sort* query execution schema. The works in top-k query answering using SPARQL are in their initial stage and no work has been done so far on top-k and federated SPARQL.

4 Research Question

Given a user-information need formulated as a top-k query over conceptual integrated model (OBDA), which describes multiple heterogeneous distributed, structured and unstructured data sources published on Web, is it possible to return the top-k best combinations of resources, which answer the information need, in less than a second and to incrementally obtain more results ordered by decreasing relevance in hundreds of milliseconds?

5 Related Works

In this section, I extend the short review of the state-of-the-art presented in Section 3. I start from two important step-stones for my work (Ontology Based Data Access and federated databases) and then I cover top-k query answering in Databases and Semantic Web.

Ontology Based Data Access (OBDA) is a method that I aim to use to address the heterogeneity problem in my research. I chose OBDA because it appears to be a mature approach. Its foundational theory was set in the beginning of 2000s [1] and focused on the DL-Lite family [2] of ontological languages. In 2012, W3C published a recommendation for an ontological language (OWL-2QL) suitable for OBDA using results of those studies, and Gartner foresee its industrial uptake in the next 2-3 years [3].

Federated database is a collection of multiple distributed, autonomous, potentially heterogeneous databases. Federated database systems provides a uniform user interface, enabling users and clients to store and retrieve data with a single query even if the constituent databases are heterogeneous. Principles of federated database systems were set in [4]. In the Semantic Web domain, federation of currently supported SPARQL 1.1 whose syntax and semantics are described in [5].

The top-k query answering problem has been studied in the **database** community to go beyond the naïve *materialize then sort* query execution schema. This schema retrieves all the data resources that match the boolean part of the query, then order them all according to the user defined ranking function and, finally, report the k most relevant results to the user. The state of the art in relational databases contains many algorithms to compute the top-k answer without materializing the answer to the boolean query. The key idea is to consider ranking as a first-class construct and interleave the computation of intermediate results

with their ordering. Ilyas et al. in [6] presented a survey on top-k query processing techniques in relational databases. They introduced various classifications for top-k query processing techniques based on multiple design dimensions, e.g., type of allowed data access method (sorted vs. random) or the type of operation (top-k selection query, top-k join query and top-k aggregate query).

For instance, the Threshold Algorithm (TA) [7] addresses the problem of answering top-k aggregated queries and uses both sorted and random access. The No Random Access algorithm [7] addresses the same problem but exploits only sorted access. The NRA-RJ [8], and Rank-Join algorithm [9] address the problem of top-k join using different mixes of sorted and random access.

RankSQL [10] is an example of DBMS that combines the algorithm presented in the previous paragraph. It introduces an algebraic framework to support efficient evaluation of the top-k queries in relational database systems by extending the relational algebra and query optimization. The key idea is to introduce a ranking operator and to make all other boolean operators rank-aware.

Some initial works on **top-k query answering** are also available in the **Semantic Web** community. Notably, it is possible to express top-k query in SPARQL by using projection functions together with ORDER BY and LIMIT clauses, but only few works investigated the optimization of this class of queries. Magliacane et al. [11] presented SPARQL-RANK, which is an extension of the SPARQL algebra and execution model that support ranking as a first-class SPARQL construct. The new algebra and the execution model provide the splitting of the ranking function and interleaving it with other operators. Wagner et al. [12] studied the top-k join problem in a Linked Data context by adapting the pull/bound rank join (PBRJ) [13] algorithm template for a push-based execution in the linked data setting. The authors of [14] extends SPARQL to querying RDFS annotated by bounded lattice (and thus comes with a partial ordering). Last, but not least, given that computation time is more important than accuracy and completeness, Wagner et al. addressed the problem of approximate top-k processing for the web of the data in [15].

The problem of the evaluation of top-k query in the context of ontology-based access has also been partially addressed. Straccia in [16] frames this problem in the context of relational databases generalizing the results of SoftFacts [17]– an ontology-mediated top-k information retrieval system over relational databases. [18] provides an interesting approach in the context of Web search.

6 Hypothesis

In order to operationalise my research question in hypotheses, I need to describe few classes of queries.

The basic one is the class of top-k SPARQL queries T that was shown to be optimizable in [11,12,15]. E.g., give me the top-5 authors who wrote the largest number of paper that are highly cited. This class of queries can be declared in SPARQL 1.1 and it can be evaluated faster and using less memory (compared to state of the art engines using materialize-then-sort processing schema) by

introducing ranking as first class construct in SPARQL algebra (see SPARQL-RANK algebra [11]) and by using split-and-interleave processing schema.

In my work I intend to investigate the class of top-k SPARQL queries that also include textual matching. Let me name this class top-k textual SPARQL queries T_t. E.g., give me the top-5 authors who wrote the largest number of paper whose title contains "rank", "top-k", and "query". This class cannot be expressed in SPARQL 1.1; few extension exists in proprietary systems (e.g., jena-text and virtuoso full text search).

This class can be split in two subclasses, those that include federated SPARQL and those that do not. Let me name them, centralized top-k textual SPARQL queries T_{tc} and federated top-k textual SPARQL queries T_{tf}.

Last, but not least, those classes of queries can be evaluated under different entailment regimes. In this work, I intend to investigate the cases of simple RDF entailment $T^{\emptyset}$ and the case of an extended version of OWL2QL T^{QL+eq} where it is possible to express simple equations between numerical values. E.g., we would like to express in OWL that the population density of a city is the ratio between the number of inhabitants and the area of the city, so that one can ask for cities ranked by population density even if some of the data sources to access only contain the number of inhabitants and the area of the cities.

Now that I have those classes, I can state my hypotheses as follows:

- *H.1*: Using an extended version of SPARQL, which treats ranking and textual matching as first class constructs, (namely SPARQL-rank$_{tc}$) will make the evaluation of $T^{\emptyset}_{tc}$ queries faster and less memory eager than existing SPARQL engines using materialize-then-sort processing schema
- *H.2*: Extending SPARQL-rank$_{tc}$ to include aspects of federated SPARQL (namely SPARQL-rank$_{tf}$) will make the evaluation of $T^{\emptyset}_{tf}$ queries faster and less memory eager than existing federated SPARQL engines using materialize-then-sort processing schema
- *H.3*: Users with information needs that cannot be homogeneously formulated on heterogeneous data sources, can declare such a need as a query of the class T^{QL+eq}_{tf} and SPARQL-rankT_{tf} will be able to evaluate it.

7 Approach

As the first step, I started an analysis of the state-of-the-art. Reviewing the works done in the domain of top-k query processing in database community is giving me ideas and is guiding me to use top-k query answering in Web domain. I am also becoming familiar with the concept from Web Information Retrieval. My next step is broadening my understating of federated SPARQL and OBDA. I am also working in identifying real use cases that that will be used in the evaluation phase. Finding the suitable datasets and a set of queries are the expected results of this step.

In the next step, I design the evaluation framework that is used to compare my work with the existing ones in order to investigate the hypotheses presented

above. The expected output is a benchmark for top-k SPARQL query answering and a set of the evaluation metrics for fair comparison of alternative approaches.

In parallel to the previous step, I start the main activity of my research that consists of three activities testing the three hypotheses. In the first one, I focus on top-k query and the presence of text searching (*H.1*). Then, I could evaluate *H.2* by extending the work done in testing *H.1* from local system to federated ones and finally, I will focus on the heterogeneity of the data (*H.3*).

8 Evaluation Plan

An evaluation framework is needed to compare the results of my investigations with the existing and appearing solutions. At this stage of the work, I foresee to use the following evaluation metrics and targets:

- *Query latency*: the time required to execute a query and compute the results. I aim to reduce it by two order of magnitude for accessing the first k results and two-three order of magnitude for incrementally obtaining the next results ordered by decreasing relevance.
- *Resource consumption*: I intend to focus on memory usage and I aim to reduce it by one-two order of magnitude.
- *Relevancy of the results*: as metric I indent to use the normalized Discounted Cumulative Gain (nDCG) which is widely used in information retrieval.
- Ability of user to formulate information need.

As dataset for *H.1*, I plan to use DBpedia and Wikipedia or the linked data version of DBLP and Google Scholar. For *H.2*, and *H.3*, I am considering the possibility to exploit Web Data Common[2], a project that extracts structured data from public web pages.

9 Preliminary Results

In my master thesis and in the first months of my PhD I worked on setting up the evaluation framework and I started the investigation of *H.1*.

As for the **evaluation framework**, I extended the DBpedia SPARQL Benchmark (DBPSB) [19] with the capabilities required to compare SPARQL engines on top-k queries and I proposed the Top-k DBpedia SPARQL Benchmark (namely, Top-k DBPSB) that uses the same dataset, performance metrics, and test driver of DBPSB. Top-K DBPSB was run against three SPARQL Engines (Virtouso, Jena TDB, and Sesame). The results of the extensive experimental evaluation confirms that existing solution are poorly optimized for top-k SPARQL queries.

As for the initial investigation of *H.1*, I am comparing the execution time of top-k SPARQL query involving text search between Jena ARQ and the Jena

[2]http://webdatacommons.org/

Text in Apache Jena 2.11.1. As a use case I am considering the need to find authors that have publication in a set of domains, which are defined using a set of keywords. For example, I am try to find the authors who write publications in the two domains: "RDF stream processing" (through the keywords such as "rdf stream", "continuous sparql", and "stream reasoning") and "top-k SPARQL query answering" (through the keywords such as "rdf", "sparql", "top-k", "top k", "order" and "reasoning"). As dataset I am using the dump of DBLP in a RDF store[3]. As expected, the results show that the execution time in Jena Text is one order of magnitude better than in Jena ARQ. I expect to be able to improve by another order of magnitude introducing ARQ-Rank [11].

10 Reflections

Previous work defines a SPARQL rank-aware algebra and extending operators to deal with sorted solution mappings. However, those works do not address the problem of query planning, which is also only partially solved in the relational world [10]. Combining text searching with structured query answering is an active field of research both in database and Semantic Web area, but the usage of top-k query answering methods (*H.1*) has not been explored, yet. Focusing on federated data resources and heterogeneity of the data is one of the most active fields of research in the Semantic Web and database domain, but also in this case the proposed works have not considered the top-k query processing approach (*H.2*). The combination of the top-k query with OBDA has been done in [16], but they consider the OBDA as a layer over the top-k query processing. There is not any exploration in interleaving ordering and reasoning, which require the combination of techniques in database and knowledge representation. To the best of my knowledge, there is not any proposed works that combine the OBDA and the Federation in top-k query processing (*H.3*).

References

1. Lenzerini, M.: Data integration: A theoretical perspective. In Popa, L., Abiteboul, S., Kolaitis, P.G., eds.: PODS, ACM (2002) 233–246
2. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The dl-lite family and relations. J. Artif. Intell. Res. (JAIR) **36** (2009) 1–69
3. Lapkin, A.: Hype cycle for big data (2012)
4. Ceri, S., Pelagatti, G.: Distributed Databases Principles and Systems. McGraw-Hill, Inc., New York, NY, USA (1984)
5. Aranda, C.B., Arenas, M., Corcho, Ó., Polleres, A.: Federating queries in sparql 1.1: Syntax, semantics and evaluation. J. Web Sem. **18**(1) (2013) 1–17
6. Ilyas, I.F., Beskales, G., Soliman, M.A.: A survey of top-k query processing techniques in relational database systems. ACM Comput. Surv. **40**(4) (2008)
7. Fagin, R., Lotem, A., Naor, M.: Optimal aggregation algorithms for middleware. In Buneman, P., ed.: PODS, ACM (2001)

[3] http://dblp.l3s.de/dblp++.php

8. Ilyas, I.F., Aref, W.G., Elmagarmid, A.K.: Joining ranked inputs in practice. In: VLDB, Morgan Kaufmann (2002) 950–961
9. Ilyas, I.F., Aref, W.G., Elmagarmid, A.K.: Supporting top-k join queries in relational databases. VLDB J. **13**(3) (2004) 207–221
10. Li, C., Chang, K.C.C., Ilyas, I.F., Song, S.: Ranksql: Query algebra and optimization for relational top-k queries. In Özcan, F., ed.: SIGMOD Conference, ACM (2005) 131–142
11. Magliacane, S., Bozzon, A., Valle, E.D.: Efficient execution of top-k sparql queries. In Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E., eds.: International Semantic Web Conference (1). Volume 7649 of Lecture Notes in Computer Science., Springer (2012) 344–360
12. Wagner, A., Tran, D.T., Ladwig, G., Harth, A., Studer, R.: Top-k linked data query processing. In Simperl, E., Cimiano, P., Polleres, A., Corcho, Ó., Presutti, V., eds.: ESWC. Volume 7295 of Lecture Notes in Computer Science., Springer (2012) 56–71
13. Schnaitter, K., Polyzotis, N.: Optimal algorithms for evaluating rank joins in database systems. ACM Trans. Database Syst. **35**(1) (2010)
14. Lopes, N., Polleres, A., Straccia, U., Zimmermann, A.: Anql: Sparqling up annotated rdfs. In Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B., eds.: International Semantic Web Conference (1). Volume 6496 of Lecture Notes in Computer Science., Springer (2010) 518–533
15. Wagner, A., Bicer, V., Tran, T.: Pay-as-you-go approximate join top-k processing for the web of data. In Presutti, V., d'Amato, C., Gandon, F., d'Aquin, M., Staab, S., Tordai, A., eds.: ESWC. Volume 8465 of Lecture Notes in Computer Science., Springer (2014) 130–145
16. Straccia, U.: On the top-k retrieval problem for ontology-based access to databases. In Pivert, O., Zadrozny, S., eds.: Flexible Approaches in Data, Information and Knowledge Management. Volume 497 of Studies in Computational Intelligence. Springer (2013) 95–114
17. Straccia, U.: Softfacts: A top-k retrieval engine for ontology mediated access to relational databases. In: SMC, IEEE (2010) 4115–4122
18. Fazzinga, B., Gianforme, G., Gottlob, G., Lukasiewicz, T.: Semantic web search based on ontological conjunctive queries. J. Web Sem. **9**(4) (2011) 453–473
19. Morsey, M., Lehmann, J., Auer, S., Ngomo, A.C.N.: Dbpedia sparql benchmark - performance assessment with real queries on real data. In Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N.F., Blomqvist, E., eds.: International Semantic Web Conference (1). Volume 7031 of Lecture Notes in Computer Science., Springer (2011) 454–469