Ontology-based top-k query answering over massive, heterogeneous, and dynamic data[*]

Daniele Dell'Aglio

Dipartimento di Elettronica, Informazione e Bioingegneria – Politecnico of Milano,
P.za L. Da Vinci, 32. I-20133 Milano - Italy
daniele.dellaglio@polimi.it

Abstract. A relevant kind of task, which is getting more and more attention in the recent years, is the selection of the most relevant elements in a number of data collections, e.g., the opinion leaders given a set of topics, the best hotel offers, and the best cities to live in. At the moment the problem is addressed through ad-hoc solutions tailored on the target scenario, setting up infrastructures able to manage the expected data loads. Even if these solutions work, they could start to have problems when the data volume, velocity and variety increase. In my research activity I will study the problem of computing the top k relevant items given a collection of data sets with both streaming and static data, an ontology describing them, and a set of top-k queries where each scoring function describes the relevance as a combination of several criteria.

1 Introduction

Relevancy. Big Data are characterized by the so-called three Vs [1]: volume (high amount of data), velocity (highly dynamic in data) and variety (data with structural and semantic heterogeneity). When those data are analysed and queried, it often happens that there is a huge number of answers, but only a small part of them is relevant (on the basis of some criteria). Let's consider, for example, Expedia[1]: it processes travel solutions and it provides as search result the top k results (i.e., the first result page) ordered by a set of user-provided criteria. In most of the cases, users find the solution they are looking for in the first page, without moving to the next ones. The input data used by Expedia can be described through the Big Data dimensions[2]. Variety is given by the fact that data are related to different domains (flight companies, hotels, and car rental services), are gathered by several sources and have to be integrated. Velocity is a critical dimension for the Expedia service: availability and price of flight tickets and hotel rooms are dynamic and the quality of the returned results is strictly

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

[1] Cf. http://www.expedia.com

[2] Even if the whole data computation process is not made by Expedia and there are intermediate data providers (e.g., Amadeus), I consider this process as executed by a black box system to highlight the features of the data.

dependent on it. Finally, data about about flights, hotels and car rental have relevant volumes.

Even if Expedia does not have Big Data problems (its infrastructure is able to cope with the data it processes), a system like Expedia that pushes further the aforementioned input data dimensions (e.g., use more domains and more dynamic data) could have problems in maintaining the quality of the supplied services. The main goal of my research is the development of methodologies and algorithms for computing top k results (ordered by a given criteria) in a Big Data scenario.

In the last decade, the research activities in this domain have addressed different sub-problems: the finding the most relevant elements can be expressed through *top-k queries*, i.e., queries that asks for the top k tuples from a dataset, given an order expressed through a scoring function [2]; the velocity is addressed by *stream computation* and *on-line streaming algorithms* to process data in real time [3]; data variety and data access can be addressed through *ontologies* to obtain an holistic view on heterogeneous data sets, exploiting the Ontology Based Data Access (OBDA) approach [4].

How to combine these methods and techniques is an open research issue: RDF stream engines [5], top-k ontological query answering [6] and top-k computation over data streams [7] are examples of novel research trends that are gathering more and more attention in the recent years. In my activity I will study how those elements can be combined in order to efficiently perform data analyses in a Big Data context.

Problem Statement. Figure 1 depicts the framework I will consider to probe the problem presented above. Data are gathered from collections of data sets $\mathbb{D}$ and data streams $\mathbb{S}$. The data model is described by an ontology $\mathcal{M}$, and the information needs are defined through a set $\mathbb{Q}$ of continuous queries, containing a subset $\mathbb{K}$ of top-k continuous queries. The problem I investigate in my activity is how to optimize the query answering in this setting.

It is worth to note that the existence of a set $\mathbb{Q}$ of queries is not a stretch: in stream processing applications is common to develop network of queries [8], where each query produces streams and consumes outputs of other queries.

The remaining of the paper is structured in the following way: Section 2 describes the related works; Section 3 presents the research questions and the relative hypotheses that will drive my research activity. Section 4 describe the approach I follow to test the hypotheses and Section 5 ends with some final considerations.

2 Related work

The raising of data stream sources introduced new problems about how to manage, process and query infinite sequences of data with high frequency rate. Two proposed approach are the Data Stream Management Systems (DSMSs) and Complex Event Processors (CEPs) [3]: the firsts transform data streams in timestamped relations (usually through the *window* operator) to be processed with

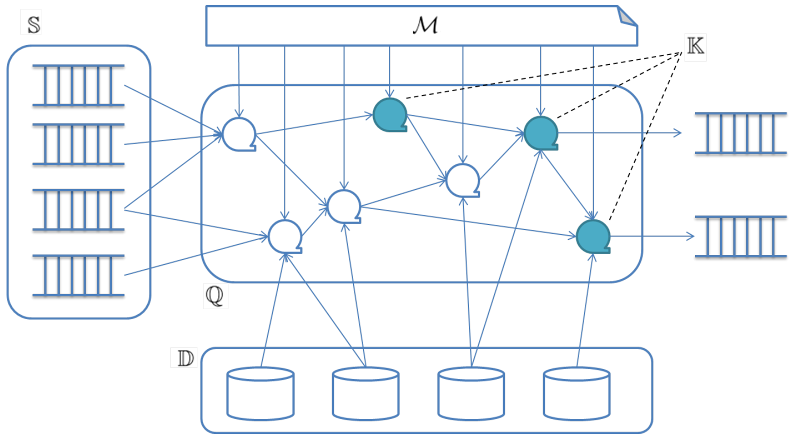

Fig. 1. Elements involved in the problem.

well known techniques such as algebras [8]; the seconds look for patterns in the streams to identify when complex events occur [9]. Recently, those paradigms have been studied by the Semantic Web community, that come out with different relevant results. On the one hand, the DSMS model inspired the design and the development of C-SPARQL [10], SPARQL$_{stream}$ [11] and CQELS [12]. On the other hand, the CEP model inspired the EP-SPARQL [13] system. RDF stream engines are at the basis of my research: they are the first step towards data streams and ontologies[3].

A parallel topic that joins Semantic Web and Stream Processing is the execution of reasoning tasks over data streams. The work in [14] proposes a method to solve reasoning tasks over an ontology stream (i.e., a sequence of timestamped ontologies). The work in [15] focuses explicitly on the ontological query answering in RDF stream processors: it proposes an algorithm, inspired to DRed [16], to incrementally maintain the ontological entailment of the content of a window. The approach is data-driven, and the entailment is updated when the window slides.

The top-k query answering problem has been widely studied in the Data Base Management System area [2]: the general idea is to extend the algebras introducing top-k related operators as first-class citizens, and to provide physical operators able to determine the top k tuples, ordered by a given criteria, without scanning the whole dataset.

Top-k query over data streams and top-k ontological query answering are research trends in an initial stage. [7] is one of the first works focusing on how to maintain a top-k answer over a stream. Regarding the top-k ontological query answering, SPARQL-RANK [17] proposes an extension of SPARQL to optimize

[3] In the following, with RDF stream engines I will indicate the RDF stream engines following the DSMS paradigm.

the top-k query answering; anyway, the approach works only under the RDF entailment regime. SoftFacts [18] is a top-k retrieval system that uses an ontology layer to manage the conceptual model (using OWL-QL as ontological language), and relational database systems to store and query the data.

3 Assumptions, research questions and hypotheses

The operational semantics of RDF stream engines such as C-SPARQL, CQELS and SPARQL$_{stream}$) can be described by the model proposed by Stream and CQL [8], and depicted in Figure 2.

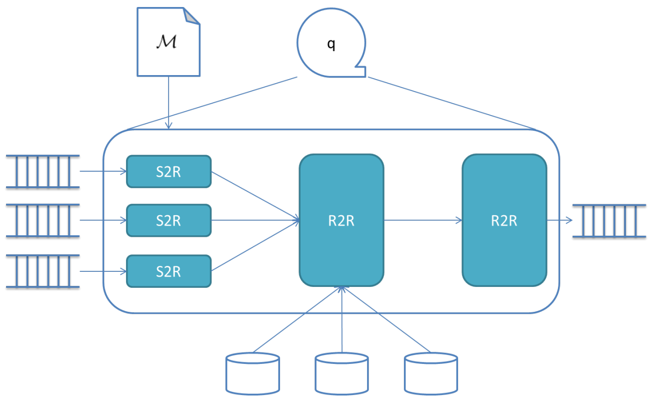

Fig. 2. General model of a continuous SPARQL query

The query answering process can be represented in three logical steps. First, the input RDF streams are transformed in sets of mappings (using SPARQL algebra terminology) through the S2R operators[4], usually sliding windows. The resulting sets of mappings and data from static data sets are transformed in a new set of mappings through a R2R operator, (the boolean expression part of the query, compliant with SPARQL 1.1/1.0). Finally a R2S operator converts the mappings in the output stream.

The reasoning technique in [15] works on the window operator: it uses the ontology $\mathcal{M}$, the window content and the static data to compute the materialization. The latter is then used as input by the R2R operator. This materialization method works under some assumptions:

- TBox assertions are not in the input stream;
- the input of the query is one window over one stream (and optionally static data sets);

[4] I maintain the operators' names as defined by CQL: S2R (stream-to-relation), R2R (relation-to-relation) and R2S (relation-to-stream).

– the same statement cannot be in both the input stream and the static data.

At the beginning of my research these assumptions hold, and I plan to investigate if they can be relaxed (in particular the second and the third ones).

Additionally, I make an assumption on the static data sets: they are available in the first or the secondary memory of the machine that execute the query. Top k algorithms need fast accesses to the data sources, and the problem of designing algorithms to execute top k queries over remote repositories (exposed via SPARQL endpoints) is out of scope in my activity.

Research questions. Starting from the problem statement and exploiting the RDF stream engine paradigm presented above, I define the following research questions:

Q.1 In RDF stream processors the query are registered before the arrival of the data, and they provide sequence of time ordered results depending on the content of the windows. How can these facts be exploited to: 1) design more efficient algorithms and 2) improve the expressiveness of the ontological language used to define the ontology $\mathcal{M}$?

Q.2 The queries in $\mathbb{Q}$ are translated in a set of logical plans (algebras) and then in physical plans to be executed. How the topology of the query network, the model $\mathcal{M}$, and the scoring functions defined by the top-k queries in $\mathbb{K}$ can be used to optimize the plans?

The two questions aim at probing the optimization of the query answering process by two different points of view. Question Q.1 focuses on how stream processing engines manages the queries: the queries in $\mathbb{Q}$ are registered in the system before the data arrive, and each query is evaluated multiple times on different portions of the input streams. Question Q.2 takes into account the query plans optimization, and in particular how the ontology $\mathcal{M}$ and the scoring functions in top-k queries $\mathbb{K}$ can be used to improve the logical plans and the physical plans of the queries.

Hypotheses. In the attempt to investigate the answers for these research questions, I formulated a set of hypotheses that will lead my activity. Regarding the question Q.1, the hypotheses are:

H.1.1 The available stream reasoning techniques can be extended to work with queries with multiple windows.

H.1.2 The available stream reasoning techniques can be optimized when there are multiple queries.

H.1.3 Due to the fact that both the conceptual model and the query are fixed, it is possible to improve the expressiveness of the ontological language used to define $\mathcal{M}$, maintaining the query answering problem over data streams treatable.

H.1.4 It is possible to move from a purely data-driven approach (i.e., materialization) to a hybrid query-driven and data-driven approach (i.e., query rewriting and materialization) to improve the memory consumption and the response time.

The four hypotheses are related to the query answering process and how it can be improved. Stream reasoning technique in [15] assumes the existence of one window (and optionally a static data source): with hypothesis H.1.1 I want to verify if and under which conditions it is possible to relax this constraint. The hypothesis H.1.2 test if, given the set $\mathbb{Q}$ of queries, it is possible to make synergies and do not maintain $|\mathbb{Q}|$ materializations separately. Hypothesis H.1.3 is related to the expressiveness of the ontological language: the technique presented in [15] works on RDFS+, but it could be possible to extend it to support more expressive languages. Finally, through the hypothesis H.1.4, I want to test if in the stream context it is possible to exploit the query registration process to rewrite the query using $\mathcal{M}$, reducing the memory consumption of the materialization and the time required to maintain it.

In parallel, I aim to probe the question Q.2 through the study of the following hypotheses:

H.2.1 Exploiting the network of queries, it is possible to optimize the query plans in RDF stream engines.
H.2.2 The presence of $\mathcal{M}$ and $\mathbb{K}$ allows to optimize the query plans at logical level.
H.2.3 It is possible to exploit $\mathcal{M}$ and $\mathbb{K}$ to design physical operators that perform faster.

In this set of hypotheses I focus on the optimization of the query plans. In stream processing engines, such as Aurora [19], groups of query plans are optimized; in hypothesis H.2.1 I want to test if optimization techniques for stream processing engines can be applied and extended to RDF stream engines. Hypothesis H.2.2 and H.2.3 aim at testing that the ontology $\mathcal{M}$ and the top-k queries $\mathbb{K}$ enable optimizations in the query plans (respectively at logical and physical level).

Reflections. The problem I am going to investigate is the optimization of continuous ontology-based top-k query answering. As explained above, even if sub-problems are addressed, at the best of my knowledge there are no results that address this problem. I think that the main reason is that methods and instruments at the basis of this activity have lacked until some time ago. For example, RDF stream processing engine and stream reasoning topics are novel and open research trends, and the research groups of these fields put the most of the effort in principles and foundation definitions. I believe that these results compose a solid basis to build my research activity in the next years.

4 Research Plan

Approach. My research starts from a deep state-of-the-art analysis on data management and description logic fields; some relevant results are presented in Section 2. In parallel, there will be the identification of a set of use cases, to determine a set of real problems from which elicit the requirements that will lead my activities. At the moment I am considering a social listening scenario:

the idea is to analyse social networks data and mobile phone records to extract knowledge.

The next step of my activity is the design of the evaluation framework and the evaluation metrics. Even if some hypotheses will be investigated through a theoretical approach, others will require an empirical approach. As consequence, it is important to define this tool from the beginning: it allows to measure the progresses of the activity and to evaluate the experiments.

As result of the state of the art analysis, the requirements elicitation and the evaluation framework definition, there will be an environment to support the core activity of my research, the tests of the hypotheses. I will follow a three-step plan. In the first step, I will focus on the hypotheses related to question Q.1, and consequently on continuous ontological query answering over multiple streams. In this phase the top-k element is missing: at the moment the research on top-k query answering is more mature than the one on inference over data streams, so I believe it is necessary to work on the latter before studying the synergies between them. In the second step, I will target Q.2, through the test of the relative hypotheses. Finally, in the third step I will bring together the results obtained. The output of each step will be a set of approaches, supported by prototypes to prove their feasibility and to evaluate them.

Evaluation Plan As explained in the previous section, an evaluation framework is required since the first steps of my activity. The metrics I aim at defining are related to the following dimensions: response time (the time required to compute the answers of the query), memory consumption, and correctness of the results.

In the previous months I started to work on the design of the evaluation framework. The starting point was the analysis of benchmarks for RDF stream processors. At the moment, at the best of my knowledge, there are two available benchmarks: SRBench [20] and LSBench [21]. The two works address different features of the RDF stream processors, such as the degree of SPARQL 1.1 adoption, the time performance and the throughput. A feature that has not been investigated yet is the correctness of the results provided RDF stream engines. I started to work on this aspect, and [22] reports some initial considerations. The most relevant one is that the available operational semantics of the RDF stream engines are not enough to model the different behaviours shown by the systems. As consequence, I am contributing in defining a framework to validate the results provided by the system.

5 Conclusion

In the next years, the Big Data market will grow [23], and, as consequence, approaches and methodologies to manage, process and extract knowledge will become more and more important. The problem I am going to investigate is the optimization of the top k elements retrieval task in a context characterized by heterogeneous and massive data streams. I presented the research questions and the hypotheses that will lead the activity, and the plan I will follow to address them.

References

1. Laney, D.: 3D data management: Controlling data volume, velocity, and variety. Technical report, META Group (February 2001)
2. 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)
3. Cugola, G., Margara, A.: Processing flows of information: From data stream to complex event processing. ACM Comput. Surv. **44**(3) (2012) 15
4. Lenzerini, M.: Data integration: A theoretical perspective. In: PODS. (2002) 233–246
5. Della Valle, E., Ceri, S., van Harmelen, F., Fensel, D.: It's a streaming world! reasoning upon rapidly changing information. IEEE Intelligent Systems **24**(6) (2009) 83–89
6. Schlobach, S.: Top-k reasoning for the semantic web. In: ISWC. (2011) 55–59
7. Mouratidis, K., Bakiras, S., Papadias, D.: Continuous monitoring of top-k queries over sliding windows. In: SIGMOD Conference. (2006) 635–646
8. Arasu, A., Babu, S., Widom, J.: The CQL continuous query language: semantic foundations and query execution. VLDB J. **15**(2) (2006) 121–142
9. Luckham, D.C.: The power of events - an introduction to complex event processing in distributed enterprise systems. ACM (2005)
10. Barbieri, D.F., Braga, D., Ceri, S., Della Valle, E., Grossniklaus, M.: C-SPARQL: A continuous query language for RDF data streams. IJSC **4**(1) (2010) 3–25
11. Calbimonte, J.P., Corcho, O., Gray, A.J.G.: Enabling ontology-based access to streaming data sources. In: ISWC. (2010) 96–111
12. Le-Phuoc, D., Dao-Tran, M., Xavier Parreira, J., Hauswirth, M.: A native and adaptive approach for unified processing of linked streams and linked data. In: ISWC. (2011) 370–388
13. Anicic, D., Fodor, P., Rudolph, S., Stojanovic, N.: EP-SPARQL: a unified language for event processing and stream reasoning. In: WWW. (2011) 635–644
14. Ren, Y., Pan, J.Z.: Optimising ontology stream reasoning with truth maintenance system. In: CIKM '11
15. Barbieri, D.F., Braga, D., Ceri, S., Della Valle, E., Grossniklaus, M.: Incremental reasoning on streams and rich background knowledge. In: ESWC (1). (2010) 1–15
16. Volz, R., Staab, S., Motik, B.: Incrementally maintaining materializations of ontologies stored in logic databases. J. Data Semantics **2** (2005) 1–34
17. Magliacane, S., Bozzon, A., Della Valle, E.: Efficient execution of top-k sparql queries. In: ISWC. (2012) 344–360
18. Straccia, U.: Softfacts: A top-k retrieval engine for ontology mediated access to relational databases. In: SMC. (2010) 4115–4122
19. Abadi, D.J., Carney, D., Çetintemel, U., Cherniack, M., Convey, C., Lee, S., Stonebraker, M., Tatbul, N., Zdonik, S.B.: Aurora: a new model and architecture for data stream management. VLDB J. **12**(2) (2003) 120–139
20. Zhang, Y., Duc, P., Corcho, O., Calbimonte, J.P.: SRBench: A Streaming RDF/SPARQL Benchmark. In: ISWC. (2012) 641–657
21. Le-Phuoc, D., Dao-Tran, M., Pham, M.D., Boncz, P., Eiter, T., Fink, M.: Linked stream data processing engines: Facts and figures. In: ISWC. (2012) 300–312
22. Dell'Aglio, D., Balduini, M., Della Valle, E.: On the need to include functional testing in rdf stream engine benchmarks. In: BeRSys 2013. (2013)
23. Gens, F.: IDC Predictions 2012: Competing for 2020 (2012)