Towards Efficient Processing of RDF Data Streams

Alejandro Llaves, Javier D. Fernández, and Oscar Corcho

Ontology Engineering Group, Universidad Politécnica de Madrid, Madrid, Spain
{allaves,jdfernandez,ocorcho}@fi.upm.es

Abstract. In the last years, there has been an increase in the amount of real-time data generated. Sensors attached to *things* are transforming how we interact with our environment. Extracting meaningful information from these streams of data is essential for some application areas and requires processing systems that scale to varying conditions in data sources, complex queries, and system failures. This paper describes ongoing research on the development of a scalable RDF streaming engine.

Keywords: RDF Stream Processing, Scalability

1 Introduction to RDF Stream Processing

The Resource Description Framework (RDF) format is the W3C standard for data interchange on the Web.[1] With the growth of the Semantic Web,[2] many organizations started to publish their data as Linked Data[3], e.g. the UK Government[4] or the European Environmental Agency[5].

The origins of Linked Stream Data [14,10] are in the enrichment of the Sensor Web with spatial, temporal, and thematic metadata [15]. Standardization initiatives such as the W3C Semantic Sensor Network Incubator Group[6] and its SSN ontology [5] have fostered the publication of sensor datasets as Linked Stream Data and its consequent integration with other datasets. A more recent initiative is the W3C RDF Stream Processing (RSP) Community Group, which works on defining a common model for producing, transmitting, and continuously querying data streams encoded in RDF.[7] Within the RSP group, several RDF stream processing systems that appeared during the last years are being studied, such as CQELS Cloud [9], C-SPARQL [3], INSTANS [13], or morph-streams [4].

In the Web of Things context, extracting information from data streams is complex because of the heterogeneity of the data, the rate of data generation, and

[1] http://www.w3.org/RDF/
[2] http://www.w3.org/standards/semanticweb/
[3] http://www.w3.org/DesignIssues/LinkedData.html
[4] http://data.gov.uk/
[5] http://semantic.eea.europa.eu/
[6] http://www.w3.org/2005/Incubator/ssn/
[7] http://www.w3.org/community/rsp/

the volume. One use case would be real-time monitoring of public transportation in a city. Here, decisions on unexpected events, such as a car crash, should be taken on short time slots based on a set of spatio-temporal data streams coming from different providers. For instance, by diverting a bus line route. Linked Stream Data may help to integrate datasets from different providers and to solve interoperability problems. Yet, remaining challenges require solutions from a stream processing approach.

2 Challenges on RDF Stream Processing

The development of scalable services for real-time stream processing involves support for high throughput, management of complex queries, low latency response, fault-tolerance, and statistics extraction, among others. Nowadays, cloud services offer solutions to some of these problems, e.g. by applying elastic load balancing in presence of input data bursts. We mainly focus on the **efficient processing of user queries over RDF streams**, which requires parallelization at the query operator level.

Distributed computing refers to the processing of data in distributed systems. The **continuous transmission of data** between sources and processing nodes, and among nodes may cause response delays that should be minimized when possible.

The **integration of historical and real-time data with background knowledge** is challenging in Web-scale environments. Many RSP systems combine background knowledge with real-time processing, but historical data management is often overlooked [11]. The efficient management of historical data is essential to detect trends in data, extract statistics, or compare old data with current data to identify anomalies [11].

3 Standards and Recommendations

The following list includes some of the main standards and recommendations (besides the previously defined RDF) that are used in the context of RDF Stream Processing on the Web of Things:

- SPARQL Protocol and RDF Query Language (SPARQL) is a W3C recommendation language for RDF querying.[8]
- GeoSPARQL is an Open Geospatial Consortium standard that defines a geographic vocabulary and query language for RDF data.[9]
- Relational Databases to RDF Mapping Language (R2RML) is a W3C recommendation language to define mappings from relational databases to RDF datasets.[10]
- The SSN ontology [5] is the de facto standard in the Semantic Web to represent sensor network data and observations.

[8] http://www.w3.org/TR/sparql11-overview/
[9] http://www.opengeospatial.org/standards/geosparql
[10] http://www.w3.org/TR/r2rml/

4 Issues on Geospatial Representation

The Web of Things is a great source of spatio-temporal data. Various vocabularies allow representing geospatial data as Linked Data, e.g. GeoSPARQL, GeoJSON,[11] or Basic Geo,[12] but so far there is no agreement in which one to use as standard. Additionally, the coordinate reference systems are often not specified in the dataset, which difficults data visualization and information extraction. These and other issues were discussed on the W3C Linking Geospatial Data workshop held on March 2014.[13]

5 Efficient Processing of Queries over RDF Streams

Heterogeneous data streams are generated from different sources, at different rates, and include multiple domains. Our purpose is to build a distributed stream processing engine capable of adapting to changing conditions while serving complex continuous queries. Some sources already generate Linked Data streams [2]. Otherwise, we provide a layer serving an ontology-based access to non RDF data stream sources. Adapters for various input formats, such as CSV or REST APIs, are used to convert heterogeneous streams to RDF.

5.1 Adaptive query processing for data streams

To reach an efficient query processing over data streams we will focus on query execution planning. Traditional databases include a query optimizer that designs an execution plan based on the registered query and data statistics. In a distributed stream processing environment, there are several aspects to contemplate: changing rates of the input data, failure of processing nodes, and distribution of workload, among others. Adaptive Query Processing (AQP) techniques [6] allow adjusting the query execution plan to varying conditions of the data input, the incoming queries, and the system. Additionally, it is used to correct query optimizer mistakes and cope with unknown statistics [1].

First, we will analyze different strategies to process query operators, such as JOIN or FILTER. We will design Storm[14] topologies to efficiently process a set of common operators based on parallelizable tasks. Storm topologies are formed by spouts and bolts. Spouts are stream sources, whereas bolts are stream processors. Then, we will define a list of queries for a specific use case and will extend the topologies to fit the queries. While new data is entering the system, a dedicated bolt will manage stream data statistics to reassign a different topology if data stream conditions vary. In Storm, the coordination between the master node, which controls the assignation of tasks to spouts and bolts, and the worker

[11] http://geojson.org/
[12] http://www.w3.org/2003/01/geo/
[13] http://www.w3.org/2014/03/lgd/
[14] http://storm.incubator.apache.org/

nodes is managed by Zookeeper.[15] However, Zookeeper does not provide elastic load balancing off-the-shelf. The use of cloud services for this purpose, such as the one offered by Amazon EC2, will be addressed in the near future.

5.2 Compressing RDF streams

To date, universal compressors (e.g. gzip) and specific RDF compressors (e.g. HDT [7]) are commonly used to reduce RDF exchange costs and delays on the network. These approaches, though, consider a static view of RDF datasets, disregarding the dynamic nature of RDF stream management. A recent work [8] points out the importance of efficient RDF stream compression and proposes an initial solution leveraging the inherent data redundancy of the RDF data streams. Based on this proposal, we are currently working on a compressed data structure specifically designed for the particularities of dynamic RDF streams. In particular, we aim at providing a lightweight serialization of RDF streams which i) minimizes the data exchange among processing nodes while ii) serving a small set of operators on the compressed data.

5.3 Architecture overview

Our engine will address real-time processing on the Web of Things context following the Lambda principles [12]. Lambda is a 3-layer architecture designed to fit the requirements of Big Data: a batch layer (using Hadoop[16]) stores all the incoming data in an immutable master dataset and pre-computes batch views; a serving layer (NoSQL database) indexes views on the master dataset; and a speed layer manages the real-time processing issues and requests data views depending on incoming queries. The cloud platform to deploy our engine should provide computing services that scale on demand, as well as elastic load balancing.

Acknowledgements

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 257641, PlanetData network of excellence. We are thankful for discussions with members of the W3C RDF Stream Processing group and the Ontology Engineering Group (Universidad Politécnica de Madrid).

References

1. Babu, S., Bizarro, P.: Adaptive Query Processing in the Looking Glass. In: Proceedings of the Second Biennial Conference on Innovative Data Systems Research (CIDR), Jan. 2005 (2005)

[15] http://zookeeper.apache.org/

[16] http://hadoop.apache.org/

2. Balduini, M., Della Valle, E., DellAglio, D., Tsytsarau, M., Palpanas, T., Confalonieri, C.: Social Listening of City Scale Events Using the Streaming Linked Data Framework. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) The Semantic Web ISWC 2013, pp. 1–16. Springer Berlin Heidelberg (2013)
3. Barbieri, D.F., Braga, D., Ceri, S., Valle, E.D., Grossniklaus, M.: C-sparql: a continuous query language for rdf data streams. Int. J. Semantic Computing 4(1), 3–25 (2010)
4. Calbimonte, J.P., Jeung, H., Corcho, O., Aberer, K.: Enabling Query Technologies for the Semantic Sensor Web. International Journal on Semantic Web and Information Systems 8(1), 43–63 (2012)
5. Compton, M., Barnaghi, P., Bermudez, L., Garcia-Castro, R., Corcho, O., Cox, S., Graybeal, J., Hauswirth, M., Henson, C., Herzog, A., Huang, V., Janowicz, K., Kelsey, W.D., Phuoc, D.L., Lefort, L., Leggieri, M., Neuhaus, H., Nikolov, A., Page, K., Passant, A., Sheth, A., Taylor, K.: The ssn ontology of the w3c semantic sensor network incubator group. Web Semantics: Science, Services and Agents on the World Wide Web 17(0) (2012)
6. Deshpande, A., Ives, Z., Raman, V.: Adaptive Query Processing. Foundations and Trends in Databases 1(1), 1–140 (Jan 2007)
7. Fernández, J.D., Martínez-Prieto, M.A., Gutiérrez, C., Polleres, A., Arias, M.: Binary {RDF} representation for publication and exchange (HDT). Web Semantics: Science, Services and Agents on the World Wide Web 19(0), 22–41 (2013)
8. Fernández García, N., Arias Fisteus, J., Sánchez Fernández, L., Fuentes-Lorenzo, D., Corcho, O.: RDSZ : An approach for lossless RDF stream compression. In: In 11th European Semantic Web Conference (ESWC 2014). pp. 1–16. Crete, Greece (2014)
9. Le-phuoc, D., Nguyen, H., Quoc, M., Van, C.L., Hauswirth, M.: Elastic and Scalable Processing of Linked Stream Data in the Cloud. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) The Semantic Web ISWC 2013, vol. 287305, pp. 280–297. Springer Berlin Heidelberg (2013)
10. Le-phuoc, D., Parreira, J.X., Hauswirth, M.: Linked Stream Data Processing. In: Reasoning Web. Semantic Technologies for Advanced Query Answering, pp. 245–289. Springer Berlin Heidelberg (2012)
11. Margara, A., Urbani, J., van Harmelen, F., Bal, H.: Streaming the web: Reasoning over dynamic data. Web Semantics: Science, Services and Agents on the World Wide Web 0(0) (0)
12. Marz, N., Warren, J.: Big Data: principles and best practices of scalable realtime data systems. Manning Publications (2013)
13. Rinne, M., Nuutila, E., Seppo, T.: INSTANS: High-Performance Event Processing with Standard RDF and SPARQL. In: Henson, C., Taylor, K., Corcho, O. (eds.) Proceedings of the 5th International Workshop on Semantic Sensor Networks. pp. 81–96. CEUR-WS, Boston, Massachusetts, USA (2012)
14. Sequeda, J.F., Corcho, O.: Linked Stream Data: A Position Paper. In: Proceedings of the 2nd International Workshop on Semantic Sensor Networks, SSN 09. CEUR-WS, Washington, USA (2009)
15. Sheth, A., Henson, C., Sahoo, S.S.: Semantic Sensor Web. IEEE Internet Computing 12(4), 78–83 (2008)