Distributed Reasoning on Semantic Data Streams[*]

Rehab Albeladi

School of Electronics and Computer Science, University of Southampton,
Southampton, SO17 1BJ, United Kingdom
{raab1g09,km,nmg}@ecs.soton.ac.uk

Abstract. Data streams are being continually generated in diverse application domains such as traffic monitoring, smart buildings, and so on. Stream Reasoning is the area that aims to combine reasoning techniques with data streams. In this paper, we present our approach to enable rule-based reasoning on semantic data streams using data flow networks in a distributed manner.

Keywords: Stream reasoning, Rete networks, XMPP.

1 Introduction

Developments in the area of the Internet and Web are constantly evolving. On one hand, the growing usage of sensors and embedded devices gives rise to a vision of the future Internet called "The Internet of Things" which aims to interconnect all these devices in a global network. By providing Internet connectivity to 'smart' embedded devices, computers will be able to automatically identify, monitor, react to, and perform actions on everyday objects. On the other hand, as the current Web is becoming the largest media of information, many researchers are working on "The Semantic Web", a vision of the future Web that aims to enable computers to understand the meanings of Web. Data in the Semantic Web has to be given well-defined meanings to be machine processable. A number of formats have been standardized such as RDF, RDFS, and OWL. The aim of these formats is to structure and give semantics to the Web data, which will then enable automatic reasoning and processing of this data.

Despite the fact that the Internet of Things focuses on the infrastructure issues and the Semantic Web focuses more on knowledge representation - as they basically work in different layers - the two visions aim to interlink the virtual and physical worlds. The Internet of Things identifies real world objects using unique RFID tags, while the Semantic Web uses URIs to uniquely identify real world objects. However, both visions can complement each other.

While event processing engines or Data Stream Management Systems [1][2] can be used to process data streams generated by the IoT devices, the heterogeneity of the data sources and formats makes interoperability a real challenge. Furthermore, stream processing engines cannot perform complex reasoning tasks due to the lack of

[*] Advisors: Kirk Martinez and Nicholas Gibbins.

semantics [3]. Adopting Semantic Web formats provides standardization and enables automatic reasoning over the IoT streaming data.

On the other hand, while semantic reasoners work efficiently on typical static knowledge, the challenge of reasoning upon rapidly changing information and data streams has received far less attention than reasoning upon static data. The combination of reasoning techniques with data streams gives rise to "Stream Reasoning", which is a little explored, but high impact research area [4]. This area aims to provide the abstractions, foundations, methods, and tools required to integrate data streams and reasoning systems.

In this research, we aim to approach the following research questions:

- How to enable rule-based reasoning over RDF data streams as data flow networks efficiently (using minimal resources) and effectively (providing timely results with high precision and recall)?
- How to distribute the reasoning network in order to maintain the scalability and improve the efficiency (e.g. by pushing filters near to data sources)?

We also aim to provide a proof-of-concept implementation, which will be evaluated for reasoning on real-time RDF streams. We expect the system to perform faster than a traditional triple store, as there is no indexing. The tradeoff between completeness of the results and processing time is expected to be controllable by varying window sizes.

2 Related Work

In the Semantic Web, data is represented in RDF, and queries can be performed using SPARQL. However, to express streaming data, RDF needs to be extended to represent time, which is an important concept in data streams. SPARQL also cannot support queries on streaming data as it lacks crucial operators found in the data stream management systems, such as the window operators. Research in the stream reasoning area mainly focuses on extending SPARQL to process RDF streams. The first attempt to extend SPARQL was presented by Bolles et al. [5]. They introduced Streaming SPARQL as a SPARQL extension to cope with window queries over RDF streams. Continuous SPARQL [6] is a SPARQL extension that follows a CQL-like [2] approach. EP-SPARQL [7] is another SPARQL extension proposed as a new language for event processing and stream reasoning.

We focus more on the infrastructure of the reasoning process. Using Rete networks [8], our approach enables reasoning in a continuous manner using low-level operators that work directly on RDF streams. Rete networks are also used in [9] to enable schema-enhanced pattern detection on RDF data streams. However, they present a fixed approach that can only operate over RDF Schema, while we aim to provide generic rule-based reasoning and query answering.

We also aim to provide a scalable distributed reasoning. Hoeksema and Kotoulas [10] present a parallel approach for stream reasoning using Yahoo S4 framework. They introduce a number of RDFS specialized reasoning Processing Elements to distribute triples over multiple streams. Continuous query answering is also supported by a number of components that can be combined to translate a subset of C-SPARQL into a parallel execution plan.

3 Proposed Approach

Continuous Reasoning: To enable the rule-based reasoning process, we use the Rete algorithm [8], in which reasoning is implemented natively over streams as data flow networks. The Rete algorithm – originally designed to solve the many pattern/ many objects problem - can process large data sets efficiently because it avoids iterating over both data elements and production rules. To avoid iteration over data elements, the Rete algorithm stores with each condition (or pattern), a list of the data elements that it matches. These lists are updated when the working memory changes. To avoid iteration over rules, rules are translated into Rete networks of nodes. The nodes represent different operators that can be shared between rules and the data flows between these nodes. The tree-like network divides the matching process into multiple steps that perform different checks, so if a data element does not match the first node, it is simply discarded and does not complete its way through the network.

A prototype RDFS reasoner for RDF data streams has been fully implemented, combining features from both reasoning techniques and stream processing techniques; it performs the inference task as a rule engine using a Rete network, while the implemented Rete network performs some DSMS operations, such as converting streams into relations by using the sliding window technique. The system is fed by RDF streams, which are matched against the RDFS entailment rules, so producing new sets of data in a continuous manner. In our initial evaluation, we have been able to demonstrate the tradeoff between completeness and execution time by varying window sizes.

Distribution: For efficient processing of large volume data, scalability is a major concern. Distributed processing of data streams enables more scalable systems as they can scale in two dimensions: the hardware performance of each computing node and the number of nodes [11]. A distributed stream reasoner should be also more fault-tolerant by avoiding a single point of failure and enabling the migration of operators between the affected nodes.

We propose distributing the Rete network operators across multiple machines and use the eXtensible Messaging and Presence Protocol (XMPP) [12] for nodes' communication. We have chosen XMPP for its push-based distribution style, which satisfies the real-time requirement of streaming applications with minimal latency, and for its ease of integration with Web technologies.

We have built a prototype system that can process RDF data streams using distributed Rete networks, in which nodes are distributed in multiple machines and can communicate with each other using XMPP in a publish/subscribe pattern, and are now working on combining this system with our previous reasoner to perform continuous reasoning on streaming RDF data in a distributed manner.

4 Conclusions and Future Work

We have proposed a stream reasoning framework using distributed Rete networks. Our prototype system supports reasoning over RDFS entailment rules using a pre-designed Rete network. The prototype can be similarly extended to support OWL 2 RL reasoning. However, to enable generic query answering, we need to define an algorithm that can dynamically translate queries into rules and then to Rete networks.

On the distribution side, we only investigated the communication aspect of the challenge. Another issue to be addressed is the load balancing. An efficient method to dynamically adjust the allocation of processing among the available nodes is required.

Finally, a set of experiments to evaluate the system needs to be planned. RDF streams with different arrival rates will be fed into the system, and the results will be evaluated using precision and recall metrics to determine the effectiveness of the system, while measuring processing time and memory consumption.

Acknowledgments. This research is funded by Taibah University, Medina, Saudi Arabia.

References

1. Abadi, D., Carney, D., Çetintemel, U., Cherniack, M., Convey, C., Lee, S., Stonebraker, M., Tatbul, N., Zdonik, S.: Aurora: A New Model and Architecture for Data Stream Management. The VLDB Journal 12(2), 120–139 (2003)
2. Arasu, A., Babcock, B., Babu, S., Datar, M., Ito, K., Motwani, R., Nishizawa, I., Srivastava, U., Thomas, D., Varma, R., Widom, J.: STREAM: The Stanford Stream Data Manager. IEEE Data Engineering Bulletin 26 (2003)
3. 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), 83–89 (2009)
4. Della Valle, E., Ceri, S., Barbieri, D.F., Braga, D., Campi, A.: A First Step Towards Stream Reasoning. In: Domingue, J., Fensel, D., Traverso, P. (eds.) FIS 2008. LNCS, vol. 5468, pp. 72–81. Springer, Heidelberg (2009)
5. Bolles, A., Grawunder, M., Jacobi, J.: Streaming SPARQL - Extending SPARQL to Process Data Streams. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 448–462. Springer, Heidelberg (2008)
6. Barbieri, D., Braga, D., Ceri, S., Grossniklaus, M.: An execution environment for C-SPARQL queries. In: 13th International Conference on Extending Database Technology. ACM, Lausanne (2010)
7. Anicic, D., Fodor, P., Rudolph, S., Stojanovic, N.: EP-SPARQL: A unified language for event processing and stream reasoning. In: Proceedings of the 20th International Conference on World Wide Web. ACM, Hyderabad (2011)
8. Forgy, C.: Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. Artificial Intelligence 19, 17–37 (1982)
9. Komazec, S., Cerri, D.: Towards efficient schema-enhanced pattern matching over RDF data streams. In: Proceedings of the First International Workshop on Ordering and Reasoning (2011)
10. Hoeksema, J., Kotoulas, S.: High-performance distributed stream reasoning using S4. In: Proceedings of the First International Workshop on Ordering and Reasoning (2011)
11. Urbani, J., Kotoulas, S., Oren, E., van Harmelen, F.: Scalable Distributed Reasoning Using MapReduce. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 634–649. Springer, Heidelberg (2009)
12. Saint-Andre, P.: Extensible messaging and presence protocol (xmpp): Core, RFC. The Internet Society (2004)