Streaming OWL

Mike Dean

BBN Technologies, Ann Arbor MI, USA mdean@bbn.com

Abstract. Stream processing can offer significant performance and scalability advantages for many Semantic Web applications. An important OWL profile for stream processing includes single OWL statements that allow inference and/or generation of new rules with single statement bodies. This position paper discusses our experiences and ideas in this area.

1 Introduction

A major next step for the Semantic Web is likely to be support for streaming content, rather than focusing on sedentary web pages and knowledge bases. In our work we've seen 10x+ performance improvements when using streaming vs. materializing and then navigating an in-memory model for suitable applications. This is analogous in the XML world to using SAX vs. DOM. Jeremy Carroll similarly found a threefold time and space improvement over abstract syntax tree approaches in applying stream processing to recognizing OWL dialects [1].

Semantic Web streaming involves processing one RDF statement at a time, while maintaining a minimal amount of state. A useful profile of OWL can be supported by streaming, as discussed in Section 2, particularly when statements are used to generate rules with single statement bodies. In keeping with the 2-character OWL 2 profile [2] naming convention, we might call such a streaming profile OWL SL (which also avoids confusion with OWL-S). Section 2 details OWL SL, while Section 3 describes previous work that led up to these ideas, Section 4 discusses a prototype implementation using DERI Pipes, and Section 5 offers a generalization. Section 6 discusses additional work we plan to pursue, and Section 7 concludes.

2 OWL SL

Table 1 shows the constructs in OWL SL. This is another example of "RDFS plus a little bit of OWL" that many Semantic Web content developers have found useful.

Table 1. OWL SL Constructs

rdf:type
rdfs:domain
rdfs:range
rdfs:subClassOf
rdfs:subPropertyOf
owl:inverseOf
owl:SymmetricProperty

3 Related Work

Early in the DARPA Agent Markup Language (DAML) program I developed `dumpont`[1], a program that provides a view of OWL class and property hierarchies while depicting restrictions using a representation that's basically a combination of Java method signatures and Kleene regular expressions. Compared to ontology browsers that focus on a single class at a time, `dumpont` provides an effective means of "seeing the forest for the trees". We periodically found processes consuming excessive CPU time on the www.daml.org system hosting the `dumpont` web service. This was usually caused by people trying to run `dumpont` on a large ontology such as OpenCyc. Converting the program from internalizing a model to a streaming implementation using Jena's ARP parser alleviated the problem.

Around 2003, I added inference support to our DAML DB triple store[2] (which is now available in open source as Parliament[3]) by adding a simple rule engine limited to single-statement bodies (which avoided any need for unification or query optimization). Triggers were set on non-variable subjects, predicates (other than rdf:type, unless it was the only non-variable) and objects that appeared in rules. Rules were generated on the fly and maintained only in memory. The idea was to generate a large number of very specific rules rather than employ a small number of more general and complex rules [3]. The application that motivated this work had a knowledge base that included a "reference load" data set of about 1 million statements plus regularly incoming triples from natural language extraction of web pages. The reference load happened to include a largely unused OWL version of the United Nations Standard Products and Services Code (UNSPSC), which included about 65,000 rdfs:subClassOf statements, each of which generated 2 in-memory rules with associated triggers. DAML DB still started up in a few seconds on a commodity server, so we never bothered to remove UNSPSC. It turns out that the types of rules and techniques we used here are exactly what's needed for stream processing.

Recently, in performing an analysis of the 2008 and 2009 Billion Triples Challenge corpora, I found a 5-10X increase in performance using stream processing [4].

[1] `http://www.daml.org/2001/03/dumpont/`, `http://www.daml.org/2003/09/dumpont/`, and `http://semwebcentral.org/projects/dumpont/`

[2] `http://www.daml.org/2001/09/damldb/`

[3] `http://parliament.semwebcentral.org`

Other people are also getting interested in streaming of Semantic Web and other content. DERI Pipes [5] provides a research framework and graphical interface for stream processing of Semantic Web and other data.. IBM System S provides a highly scalable but non-semantic streaming infrastructure. Streambase and other Complex Event Processing engines provide stream processing for tuples. Brad Allen proposed using Atom for distributing RDF content [6] at SemTech 2007 while Nova Spivak has recently blogged[4] and twittered about the Stream replacing the Web.

4 Prototype Implementation

We're developing a prototype DERI Pipes[5] operator that embodies these ideas and will report on it at the workshop. The basic approach is to check each incoming statement for each of the OWL SL constructs and execute code that either adds to the internal state (e.g. for rdfs:subClassOf) or that infers additional statements (e.g. rdf:type).

5 More General Streaming

In many streaming applications, statements are likely to come in batches (e.g. from updated web pages) rather than just one at a time. In this case, it's likely that certain constructs (e.g. OWL Restrictions) will be grouped together. Making this assumption allows us to also add owl:allValuesFrom and owl:hasValue to an extended version of OWL SL, which might be called OWL SL*.

6 Knowledge Streams

We've been developing a concept we call Knowledge Streams, which is depicted in Figure 1 (from [7]). This shows stream networks for 2 overlapping Communities of Interest (each likely using their own ontologies), with nodes (operators) providing filtering, translation, augmentation (enrichment), aggregation, alerting, inference, and other services. OWL ST could well be used for the inference operator.

Knowledge Streams can also be viewed as a step toward Semantic Complex Event Processing based on triples rather than tuples.

7 Conclusions

We've identified a profile of OWL, which we call OWL SL, that's suitable for stream processing of RDF and OWL content. We hope we've also gotten other people excited about the prospects for stream processing of active Semantic Web content.

[4] `http://www.twine.com/item/128lryv9z-46/is-the-stream-the-next-new-metaphor`
[5] `http://pipes.deri.org`

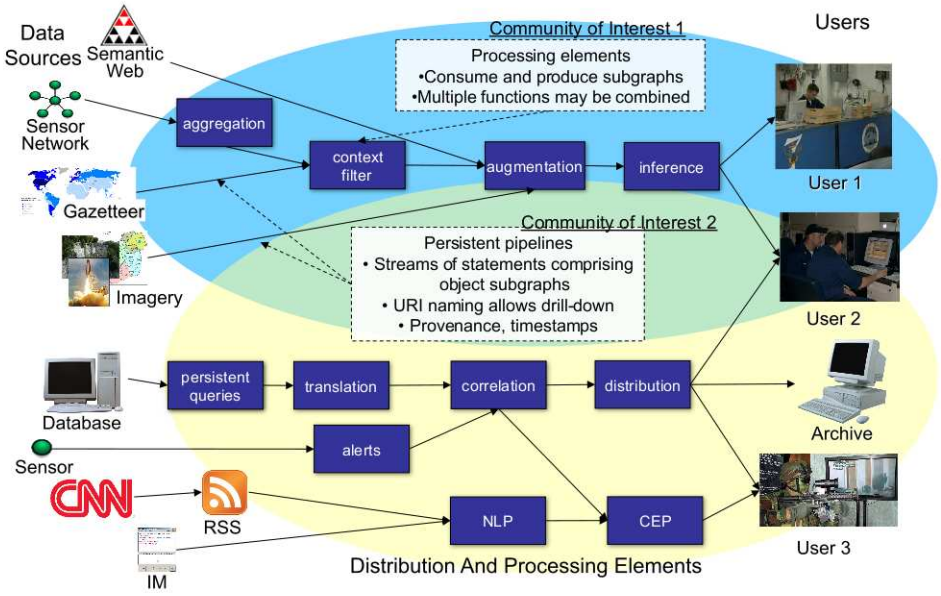

Fig. 1. Knowledge Streams

References

1. Carroll, J: Streaming OWL DL. Proc. First European Semantic Web Symposium (ESWS 2004), Heraklion, Crete, May 2004.
2. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C: OWL 2 Web Ontology Language Profiles. W3C Candidate Recommendation 11 June 2009. `http://www.w3.org/TR/2009/CR-owl2-profiles-20090611/`
3. Dean, M.: Semantic Web Rules: Covering the Use Cases. Proc. 3rd Intl. Workshop on Rules and Rule Markup Languages for the Semantic Web (RuleML 2004), Springer LNCS 3323, Hiroshima, Japan, October 2004.
4. Dean, M.: How is the Semantic Web Being Used?: An Analysis of the Billion Triples Challenge Corpus. 5th Semantic Technology Conference, San Jose, California, May 2009.
5. Le-Phuoc, D., Polleres, A., Morbidoni, C., Hauswirth, M., Tummarello, G.: Rapid Prototyping of Semantic Mash-Ups through Semantic Web Pipes. Proc. 18th World Wide Web Conference (WWW2009), Madrid, Spain, April 2009.
6. Allen, B.: A Semantic Web Without RDF/XML: Building RDF Applications in Atom. 3rd Semantic Technology Conference, San Jose, California, May 2007.
7. Dean, M., Hebeler, J.: Semantic Web @ BBN. 5th Semantic Technology Conference, San Jose, California, May 2009.