4sr - Scalable Decentralized RDFS Backward Chained Reasoning

Manuel Salvadores[1], Gianluca Correndo[1], Steve Harris[2],
Nick Gibbins[1], and Nigel Shadbolt[1]

[1]Electronics and Computer Science,
University of Southampton, Southampton, UK.
{ms8,gc3,nmg,nrs}@ecs.soton.ac.uk
[2]Garlik Ltd, UK. {steve.harris@garlik.com}

http://4sreasoner.ecs.soton.ac.uk/demo.html

Abstract. This poster paper presents the design and implementation of an RDFS reasoner based on a backward chaining approach and implemented on a clustered RDF triplestore. The system presented, called *4sr*, uses 4store as base infrastructure. In order to achieve a highly scalable system we implemented the reasoning at the lowest level of the quad store, the *bind* operation. The *bind* operation in *4sr* traverses the quad store indexes matching or expanding the query variables with awareness of the RDFS semantics.

1 Introduction

The Semantic Web community is promoting RDF stores (or triple stores) as the data storage technology for the Web of Data. RDF stores implement some extra features that make them very attractive for certain type of applications. For instance, data is not bound to a schema and it can be asserted directly from RDF sources (e.g. RDF/XML or Turtle files) due to their native support of Semantic Web data standards. But the most attractive characteristic is the possibility of implementing an entailment regime. Having entailment regimes in a triple store allows us to infer new facts, exploiting the semantics of properties and the information asserted in the knowledge base.

Today, it is still a challenge to query datasets with a few hundred of millions of triples following the RDFS regime for datasets subject to frequent changes. If we want semantic databases to handle big volumes of data transactions then we need to find backward chained approaches that do not add excessive overhead to the query phase.

4sr[1] has been implemented on the *4store* [1] RDF database. *4store* is an efficient, scalable and distributed RDF database which gives us a good platform on which to implement a backward chaining and decentralised approach.

To summarize, the main characteristics of *4sr* are:

[1] 4sr is available from http://4sreasoner.ecs.soton.ac.uk/ under GNU GPL license

- Low level RDFS Backward Chained reasoning implementation.
- Duplicate entailment detection and elimination.
- Named graph management.
- Client applications can disable/enable the RDFS entailment regime via parameters in the HTTP SPARQL endpoint.
- *4sr* doesn't add overhead at the import phase keeping intact *4store*'s import throughput (100kT/s).

2 Background

In the context of distributed techniques, [7] performs Forward Chaining (FC) parallel reasoning to expand the RDFS closure over hundreds of millions of triples. [6] pursues a similar goal and using MapReduce computes the RDFS closure over 865M triples in less than two hours. A continuation of this work has been presented in [5] providing a parallel solution to compute the OWL Horst regime.

[3] presented a novel method based on the fact that Semantic Web data present very skewed distributions among terms. Based on this evidence, the authors present a FC algorithm that works on top of data flows in a p2p-alike infrastructure. This approach reported a materialization of RDFS for 200 million triples in 7.2 minutes on a cluster of 64 nodes.

Obviously, in the last 2-3 years there has been a significant advance on materialization of closure for both RDFS and OWL languages. However very little work has been presented on how to actually query vast amounts of data and how to connect those solutions with SPARQL engines.

3 *4sr* Design and Implementation

The RDFS inferencing in *4sr* is based on two new components that have been incorporated into *4store's* architecture:

- **RDFS Sync:** A new processing node to replicate RDFS statements called *RDFS sync*. This node gathers all RDFS statements from all the storage nodes and keeps a synchronized copy of such information accessible to the *bind* operation in all the segments. After every import, update, or delete, this process extracts the new set of RDFS statements in the KB and sends it to the Storage Nodes. Even for large KBs this synchronization is fast because RDFS statements tend to be a very small proportion of the dataset. This node is also responsible for filtering out entailed duplicates.
- **bind':** The new bind function matches the quads not just taking into account the explicit knowledge but also the extensions from the RDFS semantics.

Figure 1 shows in the architecture how *bind'* and RDFS Sync interact for a hypothetical 2 storage-node deployment.

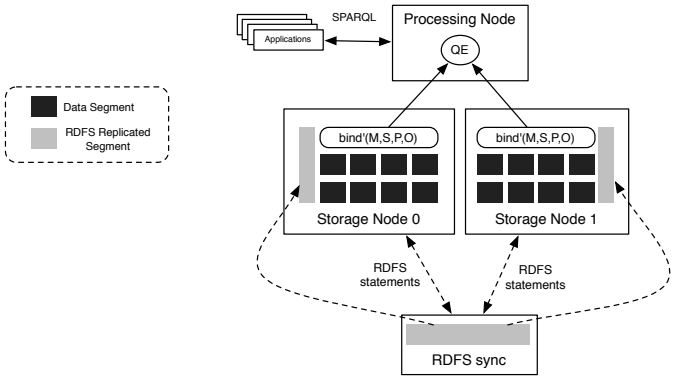

Fig. 1. 4s-reasoner architecture

The current version of *4sr* implements the RDFS rule entailments related to *rdfs:subClassOf*, *rdfs:subPropertyOf*, *rdfs:domain* and *rdfs:range*. These semantics include the rules *rdfs2*, *rdfs3*, *rdfs5*, *rdfs7*, *rdfs9*, *rdfs11*, *ext1*, *ext2*, *ext3* and *ext4* of the RDFS Rule Entailment Regime (see section 7.3 in [2]).

The *bind'* takes a super set of 4 elements $< M, S, P, O >$ to match the segment quads. The first goal in *bind'* is to expand P and O sets so as to broaden the index coverage. After this, the algorithm iterates over the M, S, P, O patterns applying not just an explicit match but a match where the members of the quads are compared taking into account RDFS semantics. Every matched solution with p or/and o unbound will be expanded according to the RDFS entailment regime. As a final step, duplicates are detected and eliminated. This approach has been presented in [4].

This architecture has been implemented in ANSI C 99 using a custom TCP/IP protocol to communicate Storage Nodes and the RDFS Sync Node. Figure 2 gives an overview of how the previous phases are implemented.

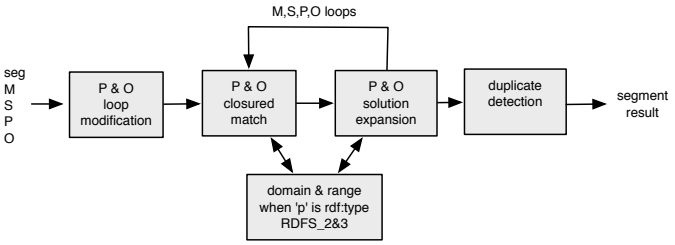

Fig. 2. Bind' processing

4 Evaluation

Due to the lack of space, we refer the committee to [4] where we preliminary tested 4sr using the Berlin SPARQL Benchmark.

5 Conclusions

In this poster paper we present a backward chained decentralized implementation of the RDFS entailment. The novelty of our work is to implement such reasoning in the bind operation of an RDF decentralized database. *4sr* offers a good balance between import throughput and query performance over the RDFS entailment regime. In that sense, *4sr* will support the development of Semantic Web applications where data can change frequently and RDFS inference is required. This poster paper will be accompanied with a demo on site similar to the one available online at:

`http://4sreasoner.ecs.soton.ac.uk/demo.html`

6 Acknowledgements

This work was supported by the EnAKTing project funded by the Engineering and Physical Sciences Research Council under contract EP/G008493/1.

References

1. Harris, S., Lamb, N., Shadbol, N.: 4store: The design and implementation of a clustered rdf store. In: Scalable Semantic Web Knowledge Base Systems - SSWS2009. pp. (p. 94–109) (2009)
2. Hayes, P., McBride, B.: Rdf semantics, w3c recommendation 10 february 2004, `http://www.w3.org/TR/rdf-mt/`
3. Kotoulas, S., Oren, E., van Harmelen, F.: Mind the data skew: Distributed inferencing by speeddating in elastic regions. In: Proceedings of the WWW 2010, Raleigh NC, USA (2010), `http://www.few.vu.nl/~kot/papers/www2010.pdf`
4. Salvadores, M., Correndo, G., Omitola, T., Gibbins, N., Harris, S., Shadbolt, N.: 4s-reasoner: Rdfs backward chained reasoning support in 4store. In: Web-scale Knowledge Representation, Retrieval, and Reasoning (Web-KR3) (September 2010), `http://eprints.ecs.soton.ac.uk/21255/`
5. Urbani, J., Kotoulas, S., Maassen, J., van Harmelen, F., Bal, H.E.: Owl reasoning with webpie: Calculating the closure of 100 billion triples. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC (1). Lecture Notes in Computer Science, vol. 6088, pp. 213–227. Springer (2010)
6. Urbani, J., Kotoulas, S., Oren, E., van Harmelen, F.: Scalable distributed reasoning using mapreduce. In: 8th International Semantic Web Conference (ISWC2009) (October 2009), `http://data.semanticweb.org/conference/iswc/2009/paper/research/374`
7. Weaver, J., Hendler, J.A.: Parallel materialization of the finite rdfs closure for hundreds of millions of triples. In: International Semantic Web Conference. pp. 682–697 (2009)