Using Off-the-Shelf Reasoners for Reasoning over Distributed ABoxes

Florian Fuchs and Michael Berger

Siemens AG, Corporate Technology, Intelligent Autonomous Systems
Otto-Hahn-Ring 6, 81739 Munich, Germany
{florian.fuchs.ext|m.berger}@siemens.com

1 Introduction

In the recent past, there has been a growing interest in reasoning over large ABoxes. As an example, consider a large-scale infrastructure monitoring system, where types of monitored events are modelled as concepts in the TBox and actual observations are represented as individuals with corresponding assertions in the ABox. ABox reasoning (such as instance retrieval and instance check) is used for analyzing the acquired monitoring data.

In many scenarios, the ABox data is produced distributedly, e.g. by the sensor system which acquired the observation. Today's established reasoners (RacerPro, Pellet, Fact++, KAON2) require the complete ABox to be available locally for query answering. However, centralizing the ABox is frequently not an option: Data is owned by different organizations, who do not want it to be replicated outside of their control. Data volumes may be too large to be handled by a single reasoner and induce high network traffic. High data rates prevent effective caching.

Existing approaches to reasoning over distributed ABoxes [1] cannot directly be adopted as they assume several semantically-mapped TBoxes. As they propose to extend the underlying reasoning algorithm, they require to modify an existing reasoning implementation, which is only possible for open source implementations and requires in-depth knowledge about implementation details.

This work therefore investigates an alternative approach, where reasoner instances are considered black boxes and process a query using subqueries to other relevant reasoner instances. We assume a set of reasoner instances $\{R_i\}$ with a common TBox $\mathcal{T}$ and a distributed ABox $\mathcal{A} = \bigcup_i \mathcal{A}_i$, where $\mathcal{A}_i$ is the ABox subset available at R_i. The goal is to answer (grounded) conjunctive queries $Q = q_1 \wedge \ldots \wedge q_n$ (with q_i (named) concept or role query terms), i.e. to determine all (named) variable bindings for which $\langle \mathcal{T}, \mathcal{A} \rangle \models Q$.

Each reasoner instance consists of a *query processing component* and a *reasoning component* with a local knowledge base. The query processing component processes the incoming query, generates subqueries to other relevant reasoner instances and adds their answers to the local knowledge base. Then the local reasoning component simply answers the original query w.r.t. the updated local knowledge base and returns the result. Off-the-shelf reasoners can be reused as a

reasoning component. The query processing component is separate and supports different query answering strategies. Apart from conjunctive queries, the query language allows retrieval of explicit ABox assertions. This setup provides a flexible framework for query answering over distributed ABoxes. We design query answering strategies for different combinations of DL expressivity and types of ABox distribution and investigate their properties such as answer completeness.

A naive strategy downloads all ABoxes $\mathcal{A}_i$ to the local knowledge base using subqueries that retrieve all assertions. The reasoning component checks the consistency of $\bigcup_i \mathcal{A}_i$ and answers Q. This strategy delivers complete answers for arbitrary ABox distributions and the DL supported by the reasoning component.

A second strategy can be used for partitioned ABoxes, i.e. $H_{\mathcal{A}_i} \cap H_{\mathcal{A}_j} = \emptyset$ for all $i, j, i \neq j$ where $H_{\mathcal{A}_i}$ is the set of individuals that occur in assertions in $\mathcal{A}_i$. In this case, $\mathcal{A}$ is consistent iff $\mathcal{A}_i$ is consistent for every i. A variable binding satisfies a conjunct q_k w.r.t. $\langle \mathcal{T}, \mathcal{A} \rangle$ iff there are i, k so that the binding satisfies q_k w.r.t. $\langle \mathcal{T}, \mathcal{A}_i \rangle$ (see also [2]). So the query Q is answered by retrieving satisfying variable bindings for each conjunct q_k from each R_i (this can be optimized by using appropriate index structures). The resulting bindings are then consolidated w.r.t. the original conjunctive query. This strategy delivers complete answers also for expressive DLs such as $\mathcal{SHIQ}$.

A third strategy allows shared individuals among ABox subsets ($N_{shared} = \bigcup_{i,j,i \neq j} H_{\mathcal{A}_i} \cap H_{\mathcal{A}_j}$), but imposes the following restrictions: For all $a \in N_{shared}$ only concept assertions $C_{shared}(a)$ and role assertions $R(a, b), R(b, a)$ with $R \sqsubseteq R_{shared}$, b arbitrary, are allowed. C_{shared} is disjoint from all other named concepts, $R \sqsubseteq R_{shared}$ does not occur in any concept definition and role axiom except role hierarchy. These restrictions still allow to express the underlying graph structure of a monitored infrastructure, for example. Moreover, consistency checks and query conjuncts can be answered locally, which allows to process conjunctive queries as described for the previous strategy. Answers are complete for expressive DLs such as $\mathcal{SHIQ}$.

We have implemented this framework and use it as a testbed for experiments with different query answering strategies: Reasoner instances use an extended SPARQL query engine as query processing component and Pellet as reasoning component. They communicate using SPARQL and the SPARQL protocol. Future work includes developing new strategies with less restrictions on ABox distribution. We investigate the spectrum between the two extremes of (i) downloading all ABox assertions (naive strategy) for gaining complete answers and (ii) downloading only ABox assertions about selected individuals for gaining a certain guaranteed degree of completeness.

References

[1] Serafini, L., Tamilin, A.: Distributed Instance Retrieval in Heterogeneous Ontologies. In: *Proc. of Workshop Semantic Web Applications and Perspectives*, 2005.

[2] Pothipruk, P., Governatori, G.: A Formal Ontology Reasoning with Individual Optimization: A Realization of the Semantic Web. In: *Proc. of Conf. on Web Information Systems Engineering*, 2005.