Proc. On the Move to Meaningful Internet Systems 2004 (OTM 2004),
Cyprus, 25.-29. Oct. 2004 Springer, LNCS 3292, 2004, pp.28-29

Efficient Processing of Huge Ontologies in Logic and Relational Databases

Timo Weithöner, Thorsten Liebig, and Günther Specht

University of Ulm, D-89069 Ulm
{weithoener|liebig|specht}@informatik.uni-ulm.de

Abstract. Today ontologies are heavily used in the sematic web. As they grow in size reasoning systems can't work without secondary storage anymore. Thus database technology is required for storing and processing huge ontologies. In this paper we present an efficient technique for representing and reasoning with ontologies in databases. We also present some benchmarking results in comparison with previous approaches.

1 Mapping Ontologies into Logic Programs

Converting ontologies into description logic programs (DLP; DLP covers most of OWL[1] Lite plus a portion of OWL DL, namely general concept inclusions with disjunction and qualified existential qualification on the left hand side) would allow to use deductive database systems as reasoners. A straightforward "Direct Mapping" approach (DiMA) to convert ontologies into a DLP was suggested in [1]. This approach maps every class or property definition into a rule and every class-instance or instance-property-instance relationship into a fact. The following gives a brief example of some frequently used statements:

DL	DLP
$A \sqsubseteq B$	B(X) :- A(X).

DL	DLP
$i : A$	A(i).

DL	DLP
$B \sqcap C \sqsubseteq A$	A(X) :- B(X), C(X).

DL	DLP
$A \sqsubseteq B \sqcap C$	B(X) :- A(X). C(X) :- A(X).

Obviously, this approach has some conceptual drawbacks:

- First, the concept names cannot be used as query results. For that reason it is impossible to get an answer to the queries like "give me all classes the individual I is instance of".
- The mapping typically results in only a few facts per literal. But the number of different rules grows linear with the complexity of the ontology.
- The names of the relations used and the structure of the rules involved vary from ontology to ontology. Consequently precompilation is impossible and query optimization becomes a real issue in this approach.

We overcome these drawbacks with our approach, which we call the Meta Mapping approach.

[1] OWL is the Web Ontology Language, see http://www.w3.org/TR/owl-features/

R. Meersman et al. (Eds.): OTM Workshops 2004, LNCS 3292, pp. 28–29, 2004.

2 The Meta Mapping Approach (MeMA)

The "Meta Mapping" from OWL DL into DLPs has an emphasis on low computational complexity and high representational flexibility. Two things have to be done to overcome the limitations mentioned above: First the rules and facts are pushed to a meta level, where names of concepts and properties become arguments of "meta predicates". Second we construct a constant set of rules valid for all ontologies accompanied by a set of fact predicates with constant names. As a result concept and property names can be used as query results as well as inputs and query formulation and optimization is eased.

Asserting instance `i` to class `C` results in instantiating the binary relation `type("i","C")`. In MeMA subclass relationships or any other kind of constructors are converted into facts which state, that the ontology defines such relationships. The following table shows some more examples:

DL	DLP
$A \sqsubseteq B$	isSub("A", "B").
$B \sqcap C \sqsubseteq A$	isSub("I1", "A"). intersectionOf("I1", {"B", "C"}).

DL	DLP
$i : A$	type("i", "A").
$A \sqsubseteq B \sqcap C$	isSub("A", "I2").. intersectionOf("I2", {"B", "C"}).

In order to reflect the underlying semantic of the introduced meta relations, we have to add adequate rules[2]. E.g. `type(I,X) :- type(I,Y), isSub(Y,X).` defines that if an individual `I` is instance of concept `Y` and `Y` is subclass of concept `X`, `I` is also an individual of class `X`. All rules are completely independent of any concrete ontology vocabulary and can thus be used for every ontology. With the combination of the ontology specific facts and the general rule we can easily perform common A- and TBox queries within the Meta Mapping approach.

3 Comparison

When benchmarking both mappings it turned out that in DiMA even a linear growth in relations results in fatal performance during preprocessing while loading the program into the CORAL deductive database. The same operation takes only seconds in the MeMA. Our experiments also observed a linear growth of processing time for class instance and subclass querying with an increasing number of individuals for both approaches. However, only in case of the MeMA better results are reached by logic databases with secondary storage indexing mechanisms. We thus get logarithmic behavior for the MeMA.

In consideration of the above we propose the Meta Mapping because of its significant conceptual advantages, higher expressivity and better performance for storing and evaluation of large scale real world ontologies in logical databases.

References

1. Grosof, B.N., Horrocks, I., Volz, R., Decker, S.: Description Logic Programms: Combining Logic Programms with Description Logic. In: Proceedings of the 12^{th} International World Wide Web Conference, Budapest, Hungary (2003)

[2] The complete rule set with 23 rules is provided at http://www.informatik.uni-ulm.de/ki/Liebig/MM-ruleset.txt