Optimizing Query Answering over OWL Ontologies*

Ilianna Kollia

ECE School, National Technical University of Athens, Greece
ilianna2@mail.ntua.gr

Abstract. Query answering is a key reasoning task for many ontology based applications in the Semantic Web. Unfortunately for OWL, the worst case complexity of query answering is very high. That is why, when the schema of an ontology is written in a highly expressive language like OWL 2 DL, currently used query answering systems do not find all answers to queries posed over the ontology, i.e., they are incomplete. In this paper optimizations are discussed that may make query answering over expressive languages feasible in practice. These optimizations mostly focus on the use of traditional database techniques that will be adapted to be applicable to knowledge bases. Moreover, caching techniques and a form of progressive query answering are also explored.

1 Problem

Query answering is an important task in the Semantic Web since it allows for the extraction of information from data as specified by the user. The answers to queries are based not only on the explicitly stated facts but also on the inferred facts. In order to dervive such implicit facts, we distinguish between the terminological and the assertional part of an ontology [2]. The terminological part, called TBox, describes general information about the modeled domain of the ontology, e.g., the relationships between classes and properties. The assertional part, called ABox, contains concrete instance data, e.g., stating which indviduals belong to a class or are related with a property.

The derivation of the implicit facts of a knowledge base is done by reasoners and is a computational problem of high complexity. For example, OWL 2 DL entailment is known to be N2ExpTime-complete [7]. Hence the expressivity of the TBox, which defines how complex the background knowledge that will be used to derive implicit facts is, constitutes one source of complexity in query answering. The other source is the size of the ABox.

A query answer is a mapping from the variables appearing in the query to terms of the queried knowledge base such that replacing the variables with their mappings yields an entailed consequence of the ontology. The well known conjunctive queries contain variables which can be mapped to individuals and literals appearing in the ABox of the queried knowledge base. A naive algorithm for finding the answers of a conjunctive query w.r.t. a knowledge base would check which of the instantiated queries (formed by substituting the variables of the query with every individual appearing in the ABox) are entailed by the knowledge base. Hence such an algorithm would perform m^n entailment checks, where m is the number of individuals in the ontology and n is the number of

* This work is partially funded by the EC Indicate project.

G. Antoniou et al. (Eds.): ESWC 2011, Part II, LNCS 6644, pp. 513–517.

variables in the query. The cost of an entailment check depends on the used logic, i.e., by using a less expressive formalism than OWL 2 DL one can reduce the complexity.

Because of the high complexity of entailment checking in OWL 2 DL, it is evident that query answering becomes problematic. Our research tackles this problem attempting to achieve as high expressivity as possible as well as efficiency in practice.

2 State of the Art

To achieve practicality, current query answering systems work either with logics of low complexity or are incomplete in the sense that they compute only some of the answers w.r.t. TBoxes of higher complexity. Moreover, in order to achieve scalability they use databases for storing the instance data.

In order to deal with large amounts of data, the reduction of a $\mathcal{SHIQ}$ knowledge base to a disjunctive datalog program that entails the same set of ground facts as the original knowledge base has been explored [6]. In this way optimization methods from deductive databases, such as the join order optimization or the magic set transformation [3] can be used to answer queries. This technique is better suited for knowledge bases with large ABoxes but small TBoxes.

According to rewriting techniques that are currently applicable to OWL QL and OWL EL, the ontology is split into a schema and a data part. The data can be stored into relational databases or triple stores. The schema part can be used to rewrite the user's query into a union of one or more conjunctive queries (for OWL QL which is based on a family of description logics called DL-Lite [4]) or a datalog query (for OWL EL) [10,11] which can then be evaluated over the data part without taking the schema into account. This approach addresses scalability issues well but suffers from the fact that the size of the rewritten queries can be large making the evaluation difficult.

Materialization techniques have been employed by systems like Jena, Sesame, OWLIM. These techniques extend the ABox with implicit facts that are entailed by the TBox. They produce incomplete answers when queries are evaluated over an ontology with an expressive TBox. This happens because in expressive fragments of OWL due to the presence of disjunctive information it is not the case that a unique canonical model can be used to answer queries. Moreover, due to the presence of existential information it cannot be guaranteed that the models of an ontology are finite. Apart from being incomplete for query answering, the use of materialization techniques is problematic in case the data change frequently because in such case the frequent recomputation of the entailed implicit facts is required which is costly.

Approximations of ontologies written in highly expressive languages have also been used for query answering. In [9] one such technique has been presented which approximates an OWL DL ontology with a DL-Lite ontology producing sound but incomplete answers to queries. It should be stated though that in the case of conjunctive queries that do not contain non-distinguished variables, the described method provides sound as well as complete answers.

A couple of optimizations for conjunctive queries over OWL ontologies have been presented in [12] that take advantage of instance retrieval optimization techniques for tableau reasoners. Most of them are more suitable to be used with tableau and not with resolution based reasoners which is the type of reasoner we will use.

3 Proposed Approach and Methodology

A naive query answering algorithm that checks all possible mappings for query variables needs optimizations in order to deal with expressive languages and behave well in practice. In my PhD, starting from the highly expressive OWL 2 DL, an optimized query answering algorithm will be devised that will use resolution based reasoners for answering conjunctive queries and it will be extended to cover also queries about the schema of an ontology. Through this work, it will be seen whether optimizations can make query answering over OWL 2 DL feasible.

A first set of optimizations will be targeted at transferring techniques from relational and deductive databases [1] to knowledge bases. For example, since it has been observed that the order in which query atoms are evaluated is of critical importance to the running time of a query, techniques from databases, such as cost based query reordering, will be used to find optimal query execution plans. These techniques will be appropriately adapted to take into account the schema of the ontology apart from the data. As an example, let us consider a conjunctive query C(x), R(x,y), D(y), where x,y are individual variables, C,D are classes and R is a property. At the moment it is not clear whether the query is more efficiently evaluated with the query atoms in the order presented above or in a different order such as C(x), D(y), R(x,y) or R(x,y), C(x), D(y).

In many cases there is no need to consider all the elements of the data part as possible mappings for query variables in conjunctive queries and hence avoid checking whether all of them lead to the entailment of the instantiated queries by the queried knowledge base. This can be so, either because the user is not interested in answers belonging to some set or because some sets of mappings are not relevant w.r.t. a query. Such cases will be identified and only an appropriate subset of the possible mappings for query variables will be checked leading hopefully to an important reduction in the running time of queries. Moreover, efficient caching techniques will be used to store parts of the models of the queried ontology since it holds that, queries instantiated by many different mappings and checked afterwards for entailment by the queried ontology, often use the same parts of models. Hence saving these models will avoid the reconstruction of them every time they are needed hopefully reducing the execution time of queries. This is especially useful in the highly expressive OWL 2 DL in which the construction of models requires a substantial amount of time.

The query answering algorithm will be made to work progressively, i.e., to output query answers as soon as they are computed, outputing first the answers that are easily computed and then answers that are more difficult to be found. For example, the answers coming from the explicitly stated facts in the ABox can be found and given to the user relatively quickly. Answers which require reasoning are more difficult to be computed and require more time. What is more, even between mappings that require reasoning to decide whether they constitute answers, the computation time needed differs substantially. This happens because in order to decide whether different mappings consistute answers, the reasoner might have to build models of different size and complexity. For example, the amount of backtracking that the reasoner performs while trying different possibilities that arise from disjunctions defines the complexity of a model and hence

the time that is needed to be constructed. This, in turn, influences the time needed to decide whether a mapping constitutes an answer or not. A more complex setting will then be adopted in which query answers are given to the user in the order of decreased relevance. This includes the definition of appropriate relevance criteria. The profile of the user who types the query can be exploited to define such measures of relevance. Since in highly expressive languages the time to compute all the answers for a query is high, through progressive query answering the user is given some answers to work with as soon as they are derived and more answers as time passes. However, the user should expect incomplete answers since some "hard" answers cannot be computed in reasonable time. The user may, therefore, be interested in the degree of (in)completeness of the used system which can be computed and presented to him.

The above described algorithm will be developed in conjuction with the SPARQL query language which is an RDF based query language that has recently been extended by W3C to find query answers under the OWL entailment relation (the OWL entailment regime of SPARQL [5]). In SPARQL a new class of powerful queries can be written which go beyond conjunctive queries. These queries allow variables in place of classes, object and data properties of OWL axioms apart from individuals and literals and need different optimizations than the ones applicable to conjunctive queries. Such queries have only partly been considered [13].

The steps that will be followed during the research are briefly described below. First, the formalized optimizations and techniques will be implemented in a system that will use SPARQL as a query language. As explained above, we will start with ontologies expressed in OWL 2 DL and see whether the applicable optimizations reduce the running time of queries to such extent that query answering becomes more feasible. In case the time for query answering is not acceptable even with the use of the considered optimizations, we will use techniques like knowledge compilation to approximate OWL DL ontologies with simplified versions of them of lower complexity and see how the use of these simplified ontologies affects the query answering times.

4 Results

A first attempt towards an optimized algorithm has been made. In particular, SPARQL has already been extended to allow the use of OWL inference for computing query answers. A cost based query reordering approach that seems to work well with conjunctive queries has been developed. A couple of optimizations have been made for the new class of expressive queries that can be represented in SPARQL. Such optimizations include the use of query rewriting techniques that transform the initial query to an equivalent one that can be evaluated more efficiently, the use of the class and property hierarchy of the queried ontology to prune the search space of candidate bindings for query variables and the use of more specialized tasks of OWL reasoners than entailment checking to speed query execution. The proposed optimizations can reduce query execution time by up to three orders of magnitude [8]. [1]

[1] This work has been done in collaboration with Dr Birte Glimm and Professor Ian Horrocks in the Oxford University Computing Laboratory.

5 Conclusion

Taking into account the fact that naive query answering techniques are impractical over expressive languages like OWL 2 DL, in my PhD I will try to devise optimized algorithms that will hopefully behave well in practice. In order to achieve this, techniques from relational and deductive databases will be transferred to knowledge bases and an evaluation of their applicability and efficiency will be made. For example, we will analyse whether the magic set technique for finding relevant parts of data w.r.t. queries and rules can be extended in our setting, where we have disjunction and existential quantification in the rule head. The results taken so far are promising. However, more tests need to be performed using a greater range of ontologies and queries.

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley, Reading (1994)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, Cambridge (2007)
3. Beeri, C., Ramakrishnan, R.: On the power of magic. In: PODS. pp. 269–284 (1987)
4. Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The DL-Lite family. J. of Automated Reasoning 39(3), 385–429 (2007)
5. Glimm, B., Krötzsch, M.: SPARQL beyond subgraph matching. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 241–256. Springer, Heidelberg (2010)
6. Hustadt, U., Motik, B., Sattler, U.: Reasoning in description logics by a reduction to disjunctive datalog. J. Autom. Reason. 39, 351–384 (2007)
7. Kazakov, Y.: $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are harder than $\mathcal{SHOIQ}$. In: Brewka, G., Lang, J. (eds.) Proc. 11th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2008), pp. 274–284. AAAI Press, Menlo Park (2008)
8. Kollia, I., Glimm, B., Horrocks, I.: SPARQL Query Answering over OWL Ontologies (2010), accepted for publication, http://www.comlab.ox.ac.uk/files/3681/paper.pdf
9. Pan, J.Z., Thomas, E.: Approximating OWL-DL ontologies. In: Proceedings of the 22nd National Conference on Artificial Intelligence, vol. 2, pp. 1434–1439. AAAI Press, Menlo Park (2007)
10. Pérez-Urbina, H., Motik, B., Horrocks, I.: Tractable query answering and rewriting under description logic constraints. Journal of Applied Logic 8(2), 186–209 (2010)
11. Rosati, R.: On conjunctive query answering in EL. In: Proceedings of the 2007 International Workshop on Description Logic (DL 2007). CEUR Electronic Workshop Proceedings (2007)
12. Sirin, E., Parsia, B.: Optimizations for answering conjunctive abox queries: First results. In: Proc. of the Int. Description Logics Workshop, DL (2006)
13. Sirin, E., Parsia, B.: SPARQL-DL: SPARQL query for OWL-DL. In: Golbreich, C., Kalyanpur, A., Parsia, B. (eds.) Proc. OWLED 2007 Workshop on OWL: Experiences and Directions. CEUR Workshop Proceedings, vol. 258 (2007), CEUR-WS.org