Querying Rich Ontologies by Exploiting the Structure of Data

Labinot Bajraktari¹

Received: 22 December 2019 / Accepted: 28 May 2020 / Published online: 6 July 2020
© The Author(s) 2020

Abstract

Ontology-based data access (OBDA) has emerged as a paradigm for accessing heterogeneous and incomplete data sources. A fundamental reasoning service in OBDA, the ontology mediated query (OMQ) answering has received much attention from the research community. However, there exists a disparity in research carried for OMQ algorithms for lightweight DLs which have found their way into practical implementations, and algorithms for expressive DLs for which the work has had mainly theoretical oriented goals. In the dissertation, a technique that leverages the structural properties of data to help alleviate the problems that typically arise when answering the queries in expressive settings is developed. In this paper, a brief summary of the technique along with the different algorithms developed for OMQ for expressive DLs is given.

Keywords Knowledge representation · Description logics · Ontologies · Query answering

1 Introduction

Managing and acquiring knowledge from today's data-bases is a complicated endeavor for organizations, in large part due to the limited semantics of the stored data. To remedy the situation one can use *description logics* (DL)—a family of ontological languages with rich modeling features that can be used effectively for representing knowledge. The *ontology-based data access* (OBDA) [1] paradigm has emerged as a way of managing and integrating traditional data sources using ontologies to add knowledge modeling and inferential capabilities on top of databases.

An essential reasoning service in OBDA and the core subject of the doctoral dissertation [2] is the *ontology mediated query* (OMQ) answering. OMQ answering has received much attention in the last decade, and there exist quite some practical algorithms implemented in reasoners for the cases where ontology uses simpler features to express the knowledge. However for *expressive* ontologies, particularly for those that use the disjunctive operator, the work has been mainly foundational and theoretical. In the doctoral dissertation, we study a more feasible approach for OMQ in the setting of expressive ontologies, which utilizes the structure

of data (*ABox* in DL jargon) for ‘guiding’ the reasoning of query answering algorithms. We propose a generic description of the structure of ABoxes, and use it for obtaining scalable algorithms for answering OMQs in expressive setting. More specifically we use the knowledge about the structure of ABoxes of interest to obtain a more goal-oriented reasoning procedure for an expressive DL (*ALCHIQ*) for a broad range of queries. We also extend our approach to hybrid languages which combine ontologies and rules. Moreover, we show the potential of adopting our approach to optimize existing algorithms, by successfully optimizing a well-known query answering algorithm for Horn-*SHIQ*. Importantly, we show that the representation of the structure of ABoxes of interest can be obtained from OBDA specifications where mappings are expressed in R2RML [3]. This makes our approach deployable in practical settings.

2 Leveraging the Structure of ABoxes

Querying a database with an ontology in between is desirable. However, it introduces significant computational challenges. Getting the answers to a query corresponds to logical inference, i.e., we have to check if the query is true in all possible scenarios (interpretations) in which the ontology and the database are interpreted consistently (such interpretations are called models). One of the most well-known approaches to OMQ answering is query rewriting, where an

✉ Labinot Bajraktari
bajraktari@kr.tuwien.ac.at

¹ Institute of Logic and Computation, Favoritenstraße 11,
1040 Wien, Austria

OMQ $(\mathcal{T}, q)$ comprising of a DL ontology $\mathcal{T}$ (a.k.a. TBox) and a query q in some standard query language (e.g., conjunctive queries (CQs)) is written into a new query q' in a target query language, such that the answers to the given OMQ $(\mathcal{T}, q)$ coincide with the answers of the q' evaluated against the plain database using existing engines for the target language. The standard approach is to rewrite the query in a data-independent way, i.e., for any possible database the answers to the rewritten query coincide with the answers of the original query. Numerous OMQ answering algorithms are based on query rewriting and practical implementations for the so-called *lightweight* DLs exist. In the case of expressive DLs, and in particular for non-Horn DLs (DLs which provide features for expressing disjunctive knowledge) most of the research on OMQ answering has had theory-oriented goals, like understanding decidability and worst-case complexity [4]. Rewritings have been proposed (e.g., [5–7]) but they appear impracticable, and to our knowledge, they have not led to implementation attempts. A rewriting into *datalog* for *SHIQ* was implemented a decade ago in the KAON2 reasoner, but only for instance queries (IQs).

The focus of the thesis was on obtaining practicable algorithms for OMQ answering for expressive DLs beyond simple query languages such as IQs. We believed that in practice, a significant obstacle for tackling the problem of query answering for expressive DLs is the standard data independent approach to query rewriting. The general idea we pursued for achieving better scalability was, to drop the data-agnostic approach to query rewriting, and to instead devise query answering algorithms that exploit structural properties of the ABoxes of interest given by a simple description. Hence our approach is complete only for ABoxes that adhere to the description characterizing the structure of such ABoxes, i.e., it is not data independent. In the next two sections, a summary of key contributions of the thesis along with core ideas behind the approach is given.

3 Optimizing Reasoning in Horn DLs

We optimize a well-known CQ answering algorithm for Horn-*SHIQ* [7] an expressive Horn DL that in addition to supporting the features of the well-known lightweight DLs (DL-Lite and $\mathcal{EL}$), support transitive roles and some number restrictions. To facilitate CQ answering the algorithm developed in [7] employs an inference calculus that saturates the given Horn-*SHIQ* TBox in such a way that for any given ABox it can be

used to facilitate query answering. The fact that the calculus saturates the TBox in a data-independent manner has its shortcomings. Consider the following example from the anti money laundry domain which uses axioms to stipulate different kinds of flags for monitored accounts (Fig. 1).

The inference calculus run over the TBox shown above would additionally derive three more axioms of the form $\text{MonitoredAccount} \sqcap C \sqsubseteq \exists \text{hasFlag}.(T \sqcap D)$, where C is some conjunction of account types occurring in the left-hand-side of universal axioms (b) and (c), while D some conjunction of flag colors occurring in the right-hand-side of such axioms. It does this since it oversees the possibility that in some ABox there exists a chance that a member of *MonitoredAccount* is also a member of any of the combination of the concepts, e.g., the member of *IndividualAccount* and *BusinessAccount*, in which case it should be flagged with the *YellowFlag* and *RedFlag*. Note that these kinds of patterns are common in ontologies, and they even occur for unrelated classes of objects for which the same “general role” is used. For example, if we also use the role *hasFlag* to flag something other than accounts, such as transactions, the calculus would derive axioms for all combinations of types of transactions and types of accounts. In practice, ontologies often omit some “common sense” disjointness assertions (such as saying that transactions are not accounts, or that accounts are not persons) which are apparent and often irrelevant for modeling the knowledge, but could have a significant impact on this kind of ABox-independent saturation. To constrain the derivation of such axioms we introduce the notion of *activators* which help us characterize the structure of ABoxes of interest. They are simply sets of concept names occurring in the ontology. They are used in the optimized version of the calculus [8] to guard against the derivation of axioms which would not fire, by checking if the left-hand-side of the axiom that is to be derived is contained in any of the activators in the set. In order to illustrate the role of activators let's consider that for the given ontology in the example above a set containing the following two activators containing two activators $\alpha_1 = \{\text{MonitoredAccount}, \text{IndividualAccount}\}$, and $\alpha_2 = \{\text{MonitoredAccount}, \text{BusinessAccount}\}$ covers all the ABoxes of interest. Since neither of the given activators contains all three concepts: *MonitoredAccount*, *IndividualAccount* and *BusinessAccount*, the optimized version of calculus would not derive an axiom that conjoins them in left-hand-side since such an axiom would be irrelevant for the ABoxes of interest, i.e., no member of *MonitoredAccount* would also be a member of *IndividualAccount* and *BusinessAccount*. The optimized version of the calculus was implemented in Clipper—the system of the original algorithm. From our tests [8] conducted against a large corpus of ontologies, we observed significant gains with the optimized version versus the original one, in terms of the number of ontologies processed, the sizes of the saturated TBoxes, and the runtime of the calculus.

$\text{MonitoredAccount} \sqsubseteq \exists \text{hasFlag}.T.$	(a)
$\text{IndividualAccount} \sqsubseteq \forall \text{hasFlag}.\text{YellowFlag}.$	(b)
$\text{BusinessAccount} \sqsubseteq \forall \text{hasFlag}.\text{RedFlag}.$	(c)

Fig. 1 An example TBox

Fig. 2 Example ABoxes

Fig. 3 Corresponding profiles for ABoxes in Fig. 2

4 Model Representations for Expressive DLs

In many application domains, there is a need for the disjunctive operator for modeling certain use cases, e.g., $\text{Person} \sqsubseteq \text{Man} \sqcup \text{Woman}$. Unfortunately, disjunctive operator plays a pivotal role in hardness, since it forces reasoning by cases which yields a significantly higher complexity for query answering. It is well-known that even for the basic DL in which one can express disjunction ($\mathcal{ALC}$) the data complexity of the problem of answering simple queries such as IQs is coNP-hard [9], a significant increase compared to PTime-complete data complexity for Horn- $\mathcal{ALC}$. For non-Horn DLs we introduce the notion of profiles as a characterization of an individual's immediate neighborhood, through which families of ABoxes can be characterized. Profiles are combinations of concept names, role domains, and role ranges that an individual may be asserted to participate in. Consider the following ABoxes shown in Fig. 2.

The corresponding profiles for these ABoxes are shown in Fig. 3. Where profile p_1 represents the structure of individuals a_1, a_4 ; p_2 represents a_2, a_6 ; p_3 represents a_3, a_7 ; and p_4 represents a_5 . We say that a given set of profiles $\mathbb{P}$ covers an ABox if for each of the individuals occurring in the ABox there exists a profile in $\mathbb{P}$ that represents it. Then given a set of profiles $\mathbb{P}$, and a TBox expressed in $\mathcal{ALCHI}$ (DL that extends $\mathcal{ALC}$ with role hierarchies and inverses), we provide an algorithm that computes the model representations of all the KBs $(\mathcal{T}, \mathcal{A})$, where $\mathcal{A}$ is covered by the set of profiles $\mathbb{P}$. The algorithm is based on *types* which are sets of concept names, and intuitively capture the configuration of concept names that an individual may realize in some model of the knowledge base. It computes a structure $\mathbf{T} = (\mathbf{L}, \mathbf{S})$ which we call a type table, where $\mathbf{L}$ records the information which enables us to link between the profiles and their corresponding types computed by the algorithm, whereas the structure $\mathbf{S}$ records the parent-child relation between types that witness

the satisfaction of existential axioms. Due to the non-deterministic nature of the logic, each profile can have multiple types, and in theory there are exponentially many types in the number of concept names occurring in the vocabulary of the ontology. However, the type table computation algorithm implements a goal-oriented procedure deriving only the relevant types needed for building the models that are sufficient for answering a broad range of queries over ABoxes covered by the profiles. More specifically, we show that $\mathbf{T}$ is complete for answering any *monotone* query. We call a query q monotone if for every interpretation $\mathcal{I}$ that can be mapped homomorphically to an interpretation $\mathcal{J}$, if $\mathcal{I}$ models q then $\mathcal{J}$ models q as well. Practically all the families of queries that have been considered in the context of DLs are monotone, and in fact, decidability results for non-monotone OMQs are very limited, e.g., [10].

Assuming a set of profiles that cover all ABoxes of interest the computation of model representations can be done offline. Moreover, the computation procedure is well suited for incremental updates in case the underlying structure of ABoxes changes. Then we devise algorithms that utilize the model representation for rewriting the queries into ASP programs for IQs, reachability queries (RQs), and an interesting query language that combines RQs and CQs with restrictions on the existential quantification of variables. We also provide a direction on the extension of the approach to arbitrary CQs, a very important query language considering that most of the queries over relational databases are essentially CQs. Algorithms for IQ and RQ answering were implemented on a proof of concept and tested over real-world ontologies with feasible computation times.

We also took the idea one step further by showing that our rewriting approach based on profiles can also be exploited in the context of expressive hybrid languages, i.e., languages that combine rules and ontologies. For this purpose, we presented an expressive new hybrid language that we call *Clopen KBs* (CKBs), which generalizes and improves the prominent r-hybrid language [11] and provide a plain ASP translation for it. We compared this translation with a direct translation of CKBs to dl-programs [12], and according to our tests [13] on a real-world ontology and data, the translation that utilizes the profiles significantly outperformed the one in dl-programs. It must also be noted that there were no implementations of the r-hybrid KBs, and since our introduced language generalizes r-hybrid KBs, our implementation can be seen as the first implementation of r-hybrid as well.

Another important result we show, is that via a simple algorithm one can obtain the activators and profiles directly from the mapping layer of an OBDA specification [2, 8]. Since the mapping layer commands how the ABoxes are generated from the underlying data sources, our profiles and activators would characterize any ABox that may result from

the evaluation of the mapping layer over any database that adheres to the specification. Hence, although our approach is not data-independent the fact that we can anticipate the structure of all the relevant ABoxes for a given OBDA specification makes our approach suitable for practical purposes.

5 Conclusions

State-of-the-art reasoners for expressive DLs can handle large ontologies (e.g., Pellet [14], Konclude [15]) and they can do fact entailment. However, they use tableau algorithms which are known to be hard to adopt even for the task of instance checking [16]. Extending them to richer query languages seems unfeasible. Our focus was on obtaining an approach that has the potential to scale for richer query languages in expressive DLs. The assumption was that in general data is of regular shape, therefore we wanted to test the viability of an approach that utilizes the structure of data to facilitate query answering. From the tests over a large corpus of ontologies, we confirmed our underlying assumption that in many data-intensive knowledge bases with a large number of individuals, the number of profiles is quite small compared to the data. Moreover, for 80% of the cases, the computation of model representation was feasible. For all these cases we also obtained encouraging results for algorithms for IQ and RQ. The approach also tested well for the case of CKBs compared to a naive translation into dl-programs. Moreover, the optimization based on activators for the case of Horn-*SHIQ* outperformed the original algorithm.

For future work, it remains to study more refined techniques for obtaining types from profiles, such that unneeded combinations of concepts are avoided. This could help scale the approach to even more cases. Another important direction is to extend the approach to more expressive DLs, and to devise algorithms for richer query languages such as CQs. Furthermore, implementing the approach of obtaining profiles and activators from OBDA specifications in [8] and testing it against real-world instances remains as an outstanding task.

Acknowledgements Open access funding provided by Austrian Science Fund (FWF). This work was supported by the Austrian Science Fund projects W1255, P30360, and P30873.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long

as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Calvanese D, De Giacomo G, Lembo D, Lenzerini M, Poggi A, Rodriguez-Muro M, Rosati R, Ruzzi M, Savo DF (2011) The MASTRO system for ontology-based data access. *Semantic Web* 2(1):43–53
2. Bajraktari L (2020) Querying rich ontologies by exploiting the structure of data. PhD dissertation, Technische Universität Wien
3. Das S, Sundara S, Cyganiak R (2012) R2RML: RDB to RDF mapping language. W3C recommendation, W3C
4. Lutz C (2008) The complexity of conjunctive query answering in expressive description logics. Springer, pp 179–193
5. Ahmetaj S, Ortiz M, Simkus M (2016) Polynomial datalog rewritings for expressive description logics with closed predicates. In: *Proceedings of IJCAI, USA*
6. Bienvenu M, ten Cate B, Lutz C, Wolter F (2014) Ontology-based data access: a study through disjunctive datalog, csp, and MMSNP. *ACM Trans Datab Syst* 39(4):33:1–33:44
7. Eiter T, Ortiz M, Šimkus M (2012) Conjunctive query answering in the description logic SH using knots. *J Comput Syst Sci* 78(1):47–85
8. Bajraktari L, Ortiz M, Xiao G (2019) Optimizing horn-*SHIQ* reasoning for OBDA. In: *Proceedings of the 18th ISWC*, pp. 75–92
9. Schaerf A (1993) On the complexity of the instance checking problem in concept languages with existential quantification. *J Intell Inf Syst* 2(3):265–278
10. Gutiérrez-Basulto V, Ibáñez-García YA, Kontchakov R, Kostylev EV (2015) Queries with negation and inequalities over lightweight ontologies. *J Web Sem* 35:184–202
11. Rosati R (2005) On the decidability and complexity of integrating ontologies and rules. *J Web Sem* 3(1):61–73
12. Eiter T, Ianni G, Lukasiewicz T, Schindlauer R, Tompits H (2008) Combining answer set programming with description logics for the semantic web. *Artif Intell* 172(12–13):1495
13. Bajraktari L, Ortiz M, Simkus M (2018) Combining rules and ontologies into clopen knowledge bases. In: *Proceedings of the 31st AAAI, New Orleans, USA*
14. Sirin E, Parsia B, Grau BC, Kalyanpur A, Katz Y (2007) Pellet: a practical OWL-DL reasoner. *J Web Sem* 5(2):51–53
15. Steigmiller A, Liebig T, Glimm B (2014) Konclude: System description. *J Web Sem* 27:78–85
16. Hustadt U, Motik B (2005) Description logics and disjunctive datalog-the story so far. In: *Proceedings of the 2005 International Workshop on Description Logics (DL2005)*, Edinburgh, Scotland, UK, July 26–28, 2005