

Characterizing Data Complexity for Conjunctive Query Answering in Expressive Description Logics*

Magdalena Ortiz^{1,2}, Diego Calvanese¹, Thomas Eiter²

¹ Faculty of Computer Science
Free University of Bozen-Bolzano
Piazza Domenicani 3, Bolzano, Italy
calvanese@inf.unibz.it,
magdalena.ortiz@stud-inf.unibz.it

² Institute of Information Systems
Vienna University of Technology
Favoritenstraße 9-11, Vienna, Austria
eiter@kr.tuwien.ac.at

Abstract

Description Logics (DLs) are the formal foundations of the standard web ontology languages OWL-DL and OWL-Lite. In the Semantic Web and other domains, ontologies are increasingly seen also as a mechanism to access and query data repositories. This novel context poses an original combination of challenges that has not been addressed before: (i) sufficient expressive power of the DL to capture common data modeling constructs; (ii) well established and flexible query mechanisms such as Conjunctive Queries (CQs); (iii) optimization of inference techniques with respect to data size, which typically dominates the size of ontologies. This calls for investigating data complexity of query answering in expressive DLs. While the complexity of DLs has been studied extensively, data complexity has been characterized only for answering atomic queries, and was still open for answering CQs in expressive DLs. We tackle this issue and prove a tight CONP upper bound for the problem in *SHIQ*, as long as no transitive roles occur in the query. We thus establish that for a whole range of DLs from \mathcal{AL} to *SHIQ*, answering CQs with no transitive roles has CONP-complete data complexity. We obtain our result by a novel tableaux-based algorithm for checking query entailment, inspired by the one in [19], but which manages the technical challenges of simultaneous inverse roles and number restrictions (which leads to a DL lacking the finite model property).

Introduction

Description Logics (DLs) [2] are specifically designed for representing structured knowledge by concepts (i.e., classes of objects) and roles (i.e., binary relationships between classes). They gained increasing attention recently as the formal foundation for the standard Web ontology languages [11]. In fact, OWL-Lite and OWL-DL are syntactic variants of the DLs *SHIF(D)* and *SHOIN(D)*, respectively [12, 21]. In the Semantic Web and domains such as Enterprise Application Integration and Data Integration, ontologies provide a high-level, conceptual view of the relevant information. However, they are increasingly seen also as a mechanism to access and query data repositories.

*This work was partially supported by the Austrian Science Funds (FWF) project P17212 and by the European Commission under project REVERSE (IST-2003-506779) and FET project TONES (FP6-7603).

Copyright © 2006, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

This novel context poses an original combination of challenges unmet before, both in DLs/ontologies and in related areas such as data modeling and querying in databases:

(i) On the one hand, a DL should have sufficient expressive power to capture common constructs used in data modeling [4]. This calls for *expressive DLs* [5, 3], in which a concept may denote the complement or union of others (to capture class disjointness and covering), may involve direct and inverse roles (to account for relationships that are traversed in both directions), and may contain number restrictions (to state existence and functionality dependencies and cardinality constraints on relationships in general). Such concepts are then used in the intentional component (called TBox) of a knowledge base, which contains inclusion assertions between concepts and roles, while the extensional component (called ABox) contains assertions about the membership of individuals to concepts and roles. A notable example of such an expressive DL is *SHIQ*, which in addition allows for asserting the transitivity of certain roles.

(ii) On the other hand, the data underlying an ontology should be accessed using well established and flexible mechanisms such as those provided by database query languages. This goes well beyond the traditional inference tasks involving objects in DL-based systems, like *instance checking* [10]. Indeed, since explicit variables are missing, DL concepts have limited possibility for relating specific data items to each other. *Conjunctive queries (CQs)*, i.e., select-project-join queries, provide a good tradeoff between expressive power and nice computational properties, and thus are adopted as core query language in several contexts, such as data integration [18].

(iii) Finally, data repositories can be very large and are usually much larger than the intentional level. Therefore, the contribution of the extensional level (i.e., the data) to the complexity of inference must be singled out, and one must pay attention to optimizing inference techniques with respect to data size, as opposed to the overall size of the knowledge base. In databases, this is accounted for by *data complexity* of query answering [23], where the relevant parameter is the size of the data, as opposed to *combined complexity*, which additionally considers the size of the query and of the schema.

As for data complexity of DLs, [10] showed that instance checking is CONP-hard already in the rather weak DL $\mathcal{AL}\mathcal{E}$, and [7] that CQ answering is CONP-hard in the yet weaker

DL \mathcal{AL} . For suitably tailored DLs, CQ answering is polynomial (actually LOGSPACE) in data complexity [6, 7]; see [7] for an investigation of the NLOGSPACE, PTIME, and CONP boundaries.

For expressive DLs (with the features above, notably inverse roles), TBox+ABox reasoning has been studied extensively using a variety of techniques ranging from reductions to Propositional Dynamic Logic (PDL) (see, e.g., [8, 5]) over tableaux [3, 14] to automata on infinite trees [5, 22]. For many such DLs, the combined complexity of TBox+ABox reasoning is EXPTIME-complete, including \mathcal{ALCCQI} [5, 22], \mathcal{DLR} [8], and \mathcal{SHIQ} [22]. However, until recently, little attention has been explicitly devoted to data complexity in expressive DLs. An EXPTIME upper bound for data complexity of CQ answering in \mathcal{DLR} follows from the results on CQ containment and view-based query answering in [8, 9]. They are based on a reduction to reasoning in PDL, which however prevents to single out the contribution to the complexity coming from the ABox. In [19] a tight CONP upper bound for CQ answering in \mathcal{ALCN} is shown. However, this DL lacks inverse roles and is thus not suited to capture semantic data models or UML. In [16, 17] a technique based on a reduction to Disjunctive Datalog is used for \mathcal{ALCHIQ} . For instance checking, and (by making use of the notion of tuple-graph [8] or via rolling-up [15]) also for tree-shaped CQs, it provides a (tight) CONP upper bound for data complexity, since it allows to single out the ABox contribution. This is not the case for general CQs, resulting in a non-tight 2EXPTIME upper bound (matching also combined complexity).

Summing up, a precise characterization of data complexity for CQ answering in expressive DLs was still open, with a gap between a CONP lower-bound and an EXPTIME upper bound. We close this gap, thus simultaneously addressing the three challenges identified above. Specifically, we make the following contributions:

- Building on techniques of [19, 14], we devise a novel tableaux-based algorithm for CQ answering over \mathcal{SHIQ} knowledge bases, that works under the assumption that the CQ does not contain transitive roles. Technically, to show soundness and completeness of the algorithm, we have to deal both with a novel blocking condition (inspired by the one in [19], but taking into account inverse and transitive roles), and with the lack of the finite model property.
- This novel algorithm provides us with a characterization of data complexity for CQ answering in expressive DLs. Specifically, we show that data complexity of CQ answering over \mathcal{SHIQ} knowledge bases is in CONP, and thus CONP-complete for all DLs ranging from \mathcal{AL} to \mathcal{SHIQ} .

For lack of space, proofs are omitted here; they can be found in an accompanying technical report [20].

Preliminaries

We only briefly recall \mathcal{SHIQ} and refer to the literature (e.g., [2]) for further details and background. We denote by \mathbf{C} , \mathbf{R} , \mathbf{R}_+ (where $\mathbf{R}_+ \subseteq \mathbf{R}$), and \mathbf{I} the sets of *concept names*, *role names*, *transitive role names*, and *individuals*

respectively. The function Inv is defined on $\mathbf{R} \cup \{P^- \mid P \in \mathbf{R}\}$ by $\text{Inv}(R) = R^-$, and $\text{Inv}(R^-) = R$; $\text{Trans}(R)$ holds true if either $R \in \mathbf{R}_+$ or $\text{Inv}(R) \in \mathbf{R}_+$.

A *role expression* R (or simply *role*) is either an atomic role name P or the inverse P^- of a role P . A *concept expression* (or simply *concept*) C is either an atomic concept name A or one of $C \sqcap D$, $C \sqcup D$, $\neg C$, $\forall R.C$, $\exists R.C$, $\geq n S.C$, or $\leq n S.C$, where C and D denote concepts, R a role, S a *simple* role (see later), and $n \geq 0$ an integer. A *knowledge base* is a triple $K = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, where

- \mathcal{T} , the *TBox*, is a set of *concept inclusion axioms* $C_1 \sqsubseteq C_2$;
- \mathcal{R} , the *role hierarchy*, is a set of *role inclusion axioms* $R_1 \sqsubseteq R_2$; and
- \mathcal{A} , the *ABox*, is a set of *assertions* $A(a)$, $P(a, b)$, and $a \neq b$, where A (resp., P) is an atomic concept (resp., role) and a and b are individuals.

Let \sqsubseteq^* denote the reflexive and transitive closure of \sqsubseteq (i.e., the *subrole relation*) over $\mathcal{R} \cup \{\text{Inv}(R_1) \sqsubseteq \text{Inv}(R_2) \mid R_1 \sqsubseteq R_2 \in \mathcal{R}\}$. A role S is *simple*, if for no role $R \in \mathbf{R}_+$ we have that $R \sqsubseteq^* S$. Without loss of expressivity, we assume that all concepts in K are in *negation normal form* (NNF), i.e., negation appears only in front of atomic concepts.

The *closure* of a concept C , $\text{clos}(C)$, is the smallest set of concept expressions containing C that is closed under subconcepts and their negation (expressed in NNF); the *closure of K* is denoted $\text{clos}(K)$ and defined as the union of all $\text{clos}(C)$ for each C occurring in K .

Unless stated otherwise, K will denote a knowledge base $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, \mathbf{R}_K the roles occurring in K and their inverses, and \mathbf{I}_K the individuals occurring in \mathcal{A} .

Example 1 As a running example, we use the knowledge base $K = \langle \{A \sqsubseteq \exists P_1.A, A \sqsubseteq \exists P_2.\neg A\}, \{\}, \{A(a)\} \rangle$.

The semantics of K is defined in terms of first-order *interpretations* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is the domain and $\cdot^{\mathcal{I}}$ the valuation function, as usual (without unique names assumption;¹ see [2]). \mathcal{I} is a *model* of K , denoted $\mathcal{I} \models K$, if it satisfies \mathcal{T} , \mathcal{R} and \mathcal{A} .

Conjunctive Queries. We assume that K has an associated set of *distinguished concept names*, denoted \mathcal{C}_q , which are the concepts that can occur in queries.

Definition 1 (conjunctive query) A conjunctive query (CQ) Q over a knowledge base K is a set of atoms of the form $\{p_1(\bar{Y}_1), \dots, p_n(\bar{Y}_n)\}$ where each p_i is either a role in \mathbf{R}_K or a concept in \mathcal{C}_q , and \bar{Y}_i is a tuple of variables or individuals in \mathbf{I}_K matching its arity.

$\text{VC}(Q)$ denotes the set of variables and individuals in Q .

CQs are interpreted in the standard way. An interpretation \mathcal{I} is a model of Q , denoted $\mathcal{I} \models Q$, if there is a substitution $\sigma : \text{VC}(Q) \rightarrow \Delta^{\mathcal{I}}$ such that $\sigma(a) = a^{\mathcal{I}}$ for each individual $a \in \text{VC}(Q)$ and $\mathcal{I} \models p(\sigma(\bar{Y}))$, for each $p(\bar{Y})$ in Q . A knowledge base K *entails* Q , denoted $K \models Q$, if $\mathcal{I} \models Q$ for each model \mathcal{I} of K .

¹The unique names assumption can be easily emulated using \neq .

