Artificial Intelligence 213 (2014) 42–59

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

The price of query rewriting in ontology-based data access

Georg Gottlob [a], Stanislav Kikot [b], Roman Kontchakov [b], Vladimir Podolskii [c], Thomas Schwentick [d], Michael Zakharyaschev [b,*]

[a] Department of Computer Science, University of Oxford, UK
[b] Department of Computer Science and Information Systems, Birkbeck, University of London, UK
[c] Steklov Mathematical Institute, Moscow, Russia
[d] Fakultät für Informatik, TU Dortmund, Germany

ARTICLE INFO

Article history:
Received 7 July 2013
Received in revised form 13 March 2014
Accepted 26 April 2014
Available online 5 May 2014

Keywords:
Ontology
Datalog
Conjunctive query
Query rewriting
Succinctness
Boolean circuit
Monotone complexity

ABSTRACT

We give a solution to the succinctness problem for the size of first-order rewritings of conjunctive queries in ontology-based data access with ontology languages such as *OWL 2 QL*, linear Datalog$^{\pm}$ and sticky Datalog$^{\pm}$. We show that positive existential and nonrecursive datalog rewritings, which do not use extra non-logical symbols (except for intensional predicates in the case of datalog rewritings), suffer an exponential blowup in the worst case, while first-order rewritings can grow superpolynomially unless NP $\subseteq$ P/poly. We also prove that nonrecursive datalog rewritings are in general exponentially more succinct than positive existential rewritings, while first-order rewritings can be superpolynomially more succinct than positive existential rewritings. On the other hand, we construct polynomial-size positive existential and nonrecursive datalog rewritings under the assumption that any data instance contains two fixed constants.

1. Introduction

Our aim in this article is to give a solution to the succinctness problem for various types of conjunctive query rewriting in ontology-based data access (OBDA) with basic ontology languages such as *OWL 2 QL* and fragments of Datalog$^{\pm}$.

The idea of OBDA has been around since about 2005 [14,19,28,47]. In the OBDA paradigm, an ontology defines a high-level global schema and provides a vocabulary for user queries. An OBDA system rewrites these queries into the vocabulary of the data and then delegates the actual query evaluation to the data sources (which can be relational databases, triple stores, datalog engines, etc.). OBDA is often regarded as an important ingredient of the new generation of information systems because it (*i*) gives a high-level conceptual view of the data, (*ii*) provides the users with a convenient vocabulary for queries, thus isolating them from the details of the structure of data sources, (*iii*) allows the system to enrich incomplete data with background knowledge, and (*iv*) supports queries to multiple and possibly heterogeneous data sources.

A key concept of OBDA is first-order (FO) rewritability. An ontology language $\mathcal{L}$ is said to enjoy *FO-rewritability* if any conjunctive query (CQ) $\boldsymbol{q}$ over any ontology Σ, formulated in $\mathcal{L}$, can be rewritten to an FO-query $\boldsymbol{q}'$ such that, for any data instance D, the answers to the original CQ $\boldsymbol{q}$ over the knowledge base (Σ, D) can be computed by evaluating the rewriting

* Corresponding author.
E-mail addresses: georg.gottlob@cs.ox.ac.uk (G. Gottlob), staskikotx@gmail.com (S. Kikot), roman@dcs.bbk.ac.uk (R. Kontchakov), podolskii@mi.ras.ru (V. Podolskii), thomas.schwentick@udo.edu (T. Schwentick), michael@dcs.bbk.ac.uk (M. Zakharyaschev).

http://dx.doi.org/10.1016/j.artint.2014.04.004
0004-3702/

$\boldsymbol{q}'$ over D. As $\boldsymbol{q}'$ is an FO-query, the answers to $\boldsymbol{q}'$ can be obtained using a standard relational database management system (RDBMS). Ontology languages with this property include the *OWL 2 QL* profile of the Web Ontology Language *OWL 2*, which is based on description logics of the *DL-Lite* family [16,4], and fragments of Datalog$^{\pm}$ such as linear tgds [11] (also known as atomic-body existential rules [6]) or sticky tgds [12,13]. To illustrate, consider an *OWL 2 QL*-ontology Σ consisting of the following tuple-generating dependencies (tgds):

$$\forall x\big(RA(x) \to \exists y\big(worksOn(x, y) \land Project(y)\big)\big), \tag{1}$$

$$\forall x\big(Project(x) \to \exists y\big(isManagedBy(x, y) \land Professor(y)\big)\big), \tag{2}$$

$$\forall x, y\big(worksOn(x, y) \to involves(y, x)\big), \tag{3}$$

$$\forall x, y\big(isManagedBy(x, y) \to involves(x, y)\big), \tag{4}$$

and the CQ $\boldsymbol{q}(x)$ asking to find those who work with professors:

$$\boldsymbol{q}(x) = \exists y, z\big(worksOn(x, y) \land involves(y, z) \land Professor(z)\big). \tag{5}$$

A moment's thought should convince the reader that the (positive existential) query

$$\boldsymbol{q}'(x) = \exists y, z\big[worksOn(x, y) \land \big(worksOn(z, y) \lor isManagedBy(y, z) \lor involves(y, z)\big) \land Professor(z)\big] \lor \exists y\big[worksOn(x, y) \land Project(y)\big] \lor RA(x)$$

is an FO-rewriting of $\boldsymbol{q}(x)$ and Σ in the sense that, for any set D of ground atoms and any constant a in D, we have

$$(\Sigma, D) \models \boldsymbol{q}(a) \quad \text{if and only if} \quad D \models \boldsymbol{q}'(a).$$

(In Section 2, we shall consider this example in more detail.) A number of different rewriting techniques have been proposed and implemented for *OWL 2 QL* (PerfectRef [47], Presto/Prexto [55,54], Rapid [18], the combined approach [37], Ontop [51,33]) and its various extensions (Requiem/Blackout [45,46], Nyaya [25,43], Clipper [20] and [35]). However, all FO-rewritings constructed so far have, in the worst case, been exponential in the size of the query $\boldsymbol{q}$. Thus, despite the fact that, for data complexity, CQ answering over ontologies with FO-rewritability is as complex as standard database query evaluation (both are in AC^0), rewritings can be too large for RDBMSs to cope with. It has become apparent, in both theory and experiments, that for the OBDA paradigm to work in practice, we have to restrict attention to those ontologies and CQs that ensure *polynomial FO-rewritability* (in the very least).

The major open question we are going to attack in this article is whether the standard ontology languages for OBDA (in particular, *OWL 2 QL*) enjoy polynomial FO-rewritability. Naturally, the answer depends on what means we can use in the rewritings. For example, in the rewriting $\boldsymbol{q}'$ of $\boldsymbol{q}$ and Σ above, we did not use any non-logical symbols other than those that occurred in $\boldsymbol{q}$ and Σ. Such rewritings (perhaps also containing equality) may be described as 'pure' as they can be used with all possible databases; cf. [16]. (Note that all known rewritings apart from the one in the combined approach [37] are pure in this sense.) Other important parameters are the available logical means (connectives and quantifiers) in rewritings and the way we represent them. Apart from the class of arbitrary FO-queries, we shall also consider positive existential (PE) queries and nonrecursive datalog (NDL) queries as possible formalisms for rewritings (needless to say that pure NDL-rewritings may contain new intensional predicates).

At first sight, the results we obtain in this article could be divided into negative and positive. The bad news is that there is a sequence of CQs $\boldsymbol{q}_n$ and *OWL 2 QL* ontologies Σ_n, both of size $O(n)$, such that any pure PE- or NDL-rewriting of $\boldsymbol{q}_n$ and Σ_n is of exponential size in n, while any pure FO-rewriting is of superpolynomial size unless NP $\subseteq$ P/poly. We obtain this negative result by first showing that OBDA with *OWL 2 QL* is powerful enough to compute monotone Boolean functions in NP, and that PE-rewritings correspond to monotone Boolean formulas, NDL-rewritings to monotone Boolean circuits, and FO-rewritings to arbitrary Boolean formulas. Then we use the celebrated exponential lower bounds for the size of monotone circuits and formulas computing the (NP-complete) Boolean function $\textsc{Clique}_{n,k}$ 'a graph with n nodes contains a k-clique' [50,49]; a superpolynomial lower bound for the size of arbitrary (not necessarily monotone) Boolean formulas computing $\textsc{Clique}_{n,k}$ is a consequence of the assumption NP $\not\subseteq$ P/poly. We also use known separation results [49,48] for monotone Boolean functions such as 'a bipartite graph with n vertices in each part has a perfect matching' and 'a given vertex is accessible in a path accessibility system with n vertices' to show that pure NDL-rewritings are in general exponentially more succinct than pure PE-rewritings, while pure FO-rewritings can be superpolynomially more succinct than pure PE-rewritings.

On the other hand, we have some good news as well: assuming that every data instance contains *two* fixed distinct individual constants, we construct polynomial-size *impure* PE- and NDL-rewritings of any CQ and any ontology with the polynomial witness property (in particular, any ontology in *OWL 2 QL*, linear Datalog$^{\pm}$ of bounded arity or sticky Datalog$^{\pm}$ of bounded arity). In essence, the rewriting guesses a polynomial number of ground atoms with database individuals and labelled nulls (encoded as tuples over the two fixed constants), and checks whether these atoms satisfy the given CQ and form a sequence of chase steps. We first construct a polynomial-size impure PE-rewriting and then show how its disjunctions can be encoded by a polynomial-size NDL-rewriting with intensional predicates of small arity. As the two