On (In)Tractability of OBDA with *OWL 2 QL*

S. Kikot, R. Kontchakov, and M. Zakharyaschev

Department of Computer Science and Information Systems,
Birkbeck College, London. U.K.
{kikot,roman,michael}@dcs.bbk.ac.uk

Abstract. We show that, although conjunctive queries over *OWL 2 QL* ontologies are reducible to database queries, no algorithm can construct such a reduction in polynomial time without changing the data. On the other hand, we give a polynomial reduction for *OWL 2 QL* ontologies without role inclusions.

1 Introduction

Ontology-based data access (OBDA) [9, 13, 21] has recently emerged as a promising application area for description logic (DL) with a potential impact on the new generation of information systems. One of the profiles of the Web Ontology Language *OWL 2*, called *OWL 2 QL*, was tailored specifically aiming at OBDA. In DL terms, OBDA involves the following reasoning problem:

CQA($\mathcal{A}, \mathcal{T}, q$): given an ABox (data) $\mathcal{A}$,[1] a TBox (ontology describing the background knowledge) $\mathcal{T}$, a conjunctive query (CQ) $q(\boldsymbol{x})$, and a tuple $\boldsymbol{a}$ of ABox elements, decide whether $\boldsymbol{a}$ is a certain answer to $q(\boldsymbol{x})$ over $(\mathcal{T}, \mathcal{A})$.

In other words, the task is to check whether $\mathcal{I} \models q(\boldsymbol{a})$ for every model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$. It is to be noted that reasoning problems of this kind are well known in logic and computer science (cf. Prolog or Datalog). A distinctive feature of *OWL 2 QL* is that 'in *OWL 2 QL*, conjunctive query answering can be implemented using conventional relational database systems. Using a suitable reasoning technique, sound and complete conjunctive query answering can be performed in LogSpace with respect to the size of the data' (`www.w3.org/TR/owl2-profiles`).

There exists a number of reductions of OBDA with *OWL 2 QL* to answering queries in relational database management systems, which transform (or *rewrite*) the problem CQA($\mathcal{A}, \mathcal{T}, q$) to the database query evaluation problem QE($\mathcal{A}, q'$), where the first-order (FO) query q' does not depend on $\mathcal{A}$. They have been implemented in the systems QuOnto [1], REQUIEM [20], Presto [23] and Nyaya [11]. In all of these approaches, the size of the query q', posed to the database system, can be $\mathcal{O}((|\mathcal{T}| \cdot |q|)^{|q|})$ in the worst case.

The aim of this paper is to try and understand whether the exponential blow-up in the size of the rewritten query is inevitable and whether polynomial rewritings are possible, at least for fragments of *OWL 2 QL*. In Section 3, we show that

[1] Here we ignore the problem of data representation in database systems; see Section 5.

the problem CQA($\{A(a)\}, \mathcal{T}, q$) for singleton ABoxes and *OWL 2 QL* TBoxes is NP-complete for combined complexity. As the problem QE($\{A(a)\}, q'$) is solved in linear time (LogSpace) in $|q'|$, it follows that no algorithm can construct FO rewritings q' (over $\{A(a)\}$) in polynomial time, unless P = NP. For *OWL 2 QL* without role inclusions and the logic $\mathcal{ELH}$, the problem CQA($\{A(a)\}, \mathcal{T}, q$) is polynomial for combined complexity, while for $\mathcal{ELI}$ it is ExpTime-complete. We observe that the parameterised complexity of the problem CQA($\{A(a)\}, \mathcal{T}, q$), where $|q|$ is regarded as a parameter, is fixed-parameter tractable. In Section 4, we present a polynomial FO rewriting of conjunctive queries over *OWL 2 QL* ontologies without role inclusions. This result improves on the polynomial rewriting from [14], which reduces CQA($\mathcal{A}, \mathcal{T}, q$) to QE($\mathcal{A} + Aux, q'$), where Aux is a set of fresh constants encoding the canonical model of $(\mathcal{T}, \mathcal{A})$. Note also the recent polynomial reduction [12] of CQA($\mathcal{A}, \mathcal{T}, q$) to QE($\mathcal{A} + \{0, 1\}, q''$), which uses two fresh constants 0, 1 and works for the extension Datalog± of *OWL 2 QL* (see Remark 1). We discuss the implications of the obtained results for OBDA in Section 5.

2 *OWL 2 QL* and *DL-Lite*$^{\mathcal{H}}_{core}$

The description logic underlying *OWL 2 QL* was introduced under the name *DL-Lite*$_{\mathcal{R}}$ [6, 7] and called *DL-Lite*$^{\mathcal{H}}_{core}$ in the more general classification of [2] (for simplicity, we disregard some constructs such as reflexivity constraints). The language of *DL-Lite*$^{\mathcal{H}}_{core}$ contains *individual names* a_i, *concept names* A_i, and *role names* P_i, $i \geq 1$. *Roles* R and *concepts* B are defined by:

$$R \quad ::= \quad P_i \quad | \quad P_i^-, \qquad\qquad B \quad ::= \quad \bot \quad | \quad \top \quad | \quad A_i \quad | \quad \exists R.$$

A *DL-Lite*$^{\mathcal{H}}_{core}$ *TBox*, $\mathcal{T}$, is a finite set of *concept* and *role inclusions* of the form $B_1 \sqsubseteq B_2$, $B_1 \sqcap B_2 \sqsubseteq \bot$ and $R_1 \sqsubseteq R_2$, $R_1 \sqcap R_2 \sqsubseteq \bot$, respectively. An *ABox*, $\mathcal{A}$, is a finite set of *assertions* of the form $B(a_i)$ and $R(a_i, a_j)$. $\mathcal{T}$ and $\mathcal{A}$ together constitute the *knowledge base* (KB) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. The semantics of *DL-Lite*$^{\mathcal{H}}_{core}$ is defined as usual in DL [4]. The presented results do not depend on the UNA. *DL-Lite*$_{core}$ is *DL-Lite*$^{\mathcal{H}}_{core}$ without role inclusions of the form $R_1 \sqsubseteq R_2$. Note also that *OWL 2 QL* contains concept inclusions of the form $B' \sqsubseteq \exists R.B$, which here will be regarded as abbreviations for *DL-Lite*$^{\mathcal{H}}_{core}$ inclusions $B' \sqsubseteq \exists R_B$, $\exists R_B^- \sqsubseteq B$ and $R_B \sqsubseteq R$, where R_B is a fresh role name.

A *conjunctive query* (CQ) $q(\boldsymbol{x})$ is a first-order formula $\exists \boldsymbol{y}\, \varphi(\boldsymbol{x}, \boldsymbol{y})$, where φ is constructed, using only $\wedge$, from atoms of the form $A(t_1)$ and $P(t_1, t_2)$, with A being a concept name, P a role name and t_i a *term* (an individual name or variable from $\boldsymbol{x}$ or $\boldsymbol{y}$). Given an ABox $\mathcal{A}$, we use $\mathsf{Ind}(\mathcal{A})$ to denote the set of individual names in $\mathcal{A}$. A tuple $\boldsymbol{a} \subseteq \mathsf{Ind}(\mathcal{A})$ is a *certain answer* to $q(\boldsymbol{x})$ over $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if $\mathcal{I} \models q[\boldsymbol{a}]$ for all models $\mathcal{I}$ of $\mathcal{K}$; in this case we write $\mathcal{K} \models q[\boldsymbol{a}]$. To simplify notation, we will often identify q with the set of its atoms and use $P^-(t, t') \in q$ as a synonym of $P(t', t) \in q$; $\mathsf{term}(q)$ is the set of terms in q.

Query answering over *OWL 2 QL* KBs is based on the fact that, for any consistent KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, there is an interpretation $\mathcal{U}_{\mathcal{K}}$ such that, for all CQs

$q(\boldsymbol{x})$ and $\boldsymbol{a} \subseteq \mathsf{Ind}(\mathcal{A})$, we have $\mathcal{K} \models q[\boldsymbol{a}]$ iff $\mathcal{U}_\mathcal{K} \models q[\boldsymbol{a}]$. The interpretation $\mathcal{U}_\mathcal{K}$, called the *canonical interpretation* of $\mathcal{K}$, is constructed as follows. Let $\sqsubseteq^*_\mathcal{T}$ be the reflexive and transitive closure of the role inclusion relation given by $\mathcal{T}$, $[R] = \{S \mid R \sqsubseteq^*_\mathcal{T} S \text{ and } S \sqsubseteq^*_\mathcal{T} R\}$, and let $[R] \leq_\mathcal{T} [S]$ iff $R \sqsubseteq^*_\mathcal{T} S$. For each $[R]$, we introduce a fresh symbol $c_{[R]}$, the *witness for* $[R]$, and define a *generating relation* $\rightsquigarrow_\mathcal{K}$ on the set of these witnesses together with $\mathsf{Ind}(\mathcal{A})$ by taking:

- $a \rightsquigarrow_\mathcal{K} c_{[R]}$ if $a \in \mathsf{Ind}(\mathcal{A})$ and $[R]$ is $\leq_\mathcal{T}$-minimal such that $\mathcal{K} \models \exists R(a)$ and $\mathcal{K} \not\models R(a, b)$ for every $b \in \mathsf{Ind}(\mathcal{A})$;
- $c_{[S]} \rightsquigarrow_\mathcal{K} c_{[R]}$ if $[R]$ is $\leq_\mathcal{T}$-minimal with $\mathcal{T} \models \exists S^- \sqsubseteq \exists R$ and $[S^-] \neq [R]$.

A *generating path* for $\mathcal{K}$ is a finite sequence $ac_{[R_1]} \cdots c_{[R_n]}$, $n \geq 0$, such that $a \in \mathsf{Ind}(\mathcal{A})$, $a \rightsquigarrow_\mathcal{K} c_{[R_1]}$ and $c_{[R_i]} \rightsquigarrow_\mathcal{K} c_{[R_{i+1}]}$, for $i < n$. Denote by $\mathsf{path}(\mathcal{K})$ the set of all generating paths for $\mathcal{K}$ and by $\mathsf{tail}(\sigma)$ the last element in $\sigma \in \mathsf{path}(\mathcal{K})$. Now, $\mathcal{U}_\mathcal{K}$ is defined by taking:

$$\begin{aligned}
\Delta^{\mathcal{U}_\mathcal{K}} &= \mathsf{path}(\mathcal{K}), \quad a^{\mathcal{U}_\mathcal{K}} = a, \text{ for all } a \in \mathsf{Ind}(\mathcal{A}),\\
A^{\mathcal{U}_\mathcal{K}} &= \{a \in \mathsf{Ind}(\mathcal{A}) \mid \mathcal{K} \models A(a)\} \cup \{\sigma \cdot c_{[R]} \mid \mathcal{T} \models \exists R^- \sqsubseteq A\},\\
P^{\mathcal{U}_\mathcal{K}} &= \{(a, b) \in \mathsf{Ind}(\mathcal{A}) \times \mathsf{Ind}(\mathcal{A}) \mid \mathcal{K} \models P(a, b)\} \cup\\
&\quad\ \{(\sigma, \sigma \cdot c_{[R]}) \mid \mathsf{tail}(\sigma) \rightsquigarrow_\mathcal{K} c_{[R]},\ [R] \leq_\mathcal{T} [P]\} \cup\\
&\quad\ \{(\sigma \cdot c_{[R]}, \sigma) \mid \mathsf{tail}(\sigma) \rightsquigarrow_\mathcal{K} c_{[R]},\ [R] \leq_\mathcal{T} [P^-]\}.
\end{aligned}$$

We shall also need a compact representation of (in general infinite) $\mathcal{U}_\mathcal{K}$ in the form of the *generating interpretation* $\mathcal{G}_\mathcal{K} = (\Delta^{\mathcal{G}_\mathcal{K}}, \cdot^{\mathcal{G}_\mathcal{K}})$ defined as follows. Its domain $\Delta^{\mathcal{G}_\mathcal{K}}$ consists of $\mathsf{Ind}(\mathcal{A})$ and all $c_{[R_n]}$ for which there are generating paths $ac_{[R_1]} \cdots c_{[R_n]} \in \mathsf{path}(\mathcal{K})$; and we set $a^{\mathcal{G}_\mathcal{K}} = a^{\mathcal{U}_\mathcal{K}}$, $A^{\mathcal{G}_\mathcal{K}} = \{\mathsf{tail}(\sigma) \mid \sigma \in A^{\mathcal{U}_\mathcal{K}}\}$ and $P^{\mathcal{G}_\mathcal{K}} = \{(\mathsf{tail}(\sigma), \mathsf{tail}(\sigma')) \mid (\sigma, \sigma') \in P^{\mathcal{U}_\mathcal{K}}\}$. It is readily seen that $\mathcal{G}_\mathcal{K}$ can be constructed in polynomial time in $\mathcal{K}$.

The problem $\mathrm{CQA}(\mathcal{A}, \mathcal{T}, q)$, for *DL-Lite*$^\mathcal{H}_{core}$ TBoxes $\mathcal{T}$, is reducible to the database query evaluation problem $\mathrm{QE}(\mathcal{A}, q')$, with q' being independent of $\mathcal{A}$ [7, 2]. However, in all known reductions, the size of q' is exponential in the size of q: for instance, $|q'| = \mathcal{O}((|\mathcal{T}| \cdot |q|)^{|q|})$ for both QuOnto [1] and REQUIEM [20]; Presto [23] uses sophisticated optimisation techniques and produces a non-recursive Datalog program q', which is still exponential in the worst case. The size of q' is irrelevant for data complexity, but heavily influences the performance of database systems; see Section 5 for a discussion.

In the next section, we show that no algorithm can produce FO rewritings q' of CQs q and *DL-Lite*$^\mathcal{H}_{core}$ TBoxes $\mathcal{T}$ in polynomial time (unless P = NP).

3 Intractability of Query Rewriting for *OWL 2 QL*

To see that query rewriting for *DL-Lite*$^\mathcal{H}_{core}$ is not tractable, we separate the contributions of $\mathcal{A}$ and $\mathcal{T}$ to the complexity of the problem $\mathrm{CQA}(\mathcal{A}, \mathcal{T}, q)$. Indeed, NP-completeness of $\mathrm{CQA}(\mathcal{A}, \mathcal{T}, q)$ for combined complexity does not give any information on the size of rewritings because the lower bound follows from NP-hardness of database query evaluation. To remove the influence of $\mathcal{A}$, one can analyse the combined complexity of CQ answering over *singleton* ABoxes of the form $\mathcal{A} = \{A(a)\}$, i.e., the problem CQA$(\{A(a)\}, \mathcal{T}, q)$. We call this measure the *primitive combined complexity* (PCC). The reason behind this notion is that any FO query q over such an ABox alone can be answered in linear time in $|q|$. Thus, if CQ answering is NP-hard for PCC, then no algorithm can *construct* FO rewritings QE$(\mathcal{A}, q')$ of CQA$(\mathcal{A}, \mathcal{T}, q)$ in polynomial time, unless P = NP.

Theorem 1. *CQ answering in DL-Lite$^{\mathcal{H}}_{core}$ is* NP*-complete for PCC.*

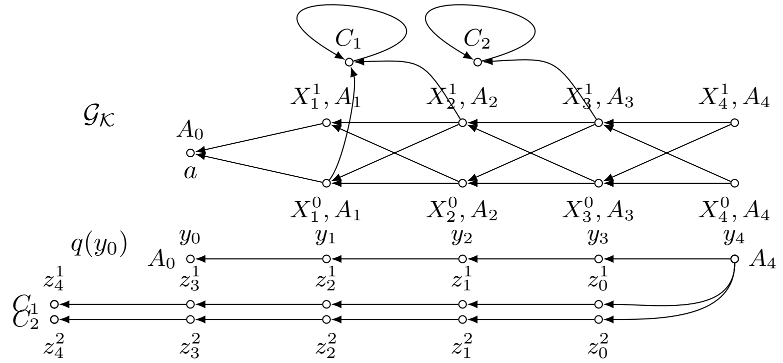
Proof. The lower bound is proved by reduction of Boolean satisfiability. Given a CNF $\varphi = \bigwedge_{j=1}^{m} D_j$ over variables $p_1, \dots, p_n$, where D_j is a clause, we consider the TBox $\mathcal{T}$ containing the following axioms, for $1 \leq i \leq n$, $1 \leq j \leq m$, $k = 0, 1$:

$$A_{i-1} \sqsubseteq \exists P^{-}.X_i^k, \qquad X_i^k \sqsubseteq A_i,$$
$$X_i^0 \sqsubseteq \exists P.C_j \quad \text{if } \neg p_i \in D_j, \qquad X_i^1 \sqsubseteq \exists P.C_j \quad \text{if } p_i \in D_j, \qquad C_j \sqsubseteq \exists P.C_j.$$

The canonical interpretation $\mathcal{U}_{\mathcal{K}}$ of $\mathcal{K} = (\mathcal{T}, \{A_0(a)\})$ is obtained by 'unravelling' the generating interpretation $\mathcal{G}_{\mathcal{K}}$ shown below. Consider the CQ $q(y_0)$:

$$q(y_0) \;=\; \exists \boldsymbol{y}\boldsymbol{z}^1 \dots \boldsymbol{z}^m \big[A_0(y_0) \land \bigwedge_{i=1}^{n} P(y_i, y_{i-1}) \land A_n(y_n) \land \\ \bigwedge_{j=1}^{m}\big(P(y_n, z_0^j) \land \bigwedge_{i=1}^{n} P(z_{i-1}^j, z_i^j) \land C_j(z_n^j)\big)\big].$$

(Note n atoms P connecting y_n to y_0 and $n+1$ atoms P connecting y_n to z_n^j, which means that any match of q in $\mathcal{U}_{\mathcal{K}}$ must map z_n^j onto a point in the infinite chain containing C_j.) One can show that φ is satisfiable iff $\mathcal{K} \models q(a)$.



Theorem 2. *Unless* P = NP*, no polynomial-time algorithm can reduce the problem CQA$(\mathcal{A}, \mathcal{T}, q)$, for DL-Lite$^{\mathcal{H}}_{core}$ TBoxes $\mathcal{T}$ and CQs q, to the problem QE$(\mathcal{A}, q')$, where q' is a first-order query independent of $\mathcal{A}$.*

Note that it is still open whether, for any $\mathcal{A}$, $\mathcal{T}$ and q, there *exists* a polynomial FO query q' giving the same answers over $\mathcal{A}$ as q over $(\mathcal{T}, \mathcal{A})$.

Remark 1. If we extend the ABox with fresh constants 0 and 1 then $q(y_0)$ in the proof above can be rewritten as $A_0(y_0) \land \exists p_1 \dots \exists p_n (\bigwedge_{i=1}^{n}(p_i \neq y_0) \land \bigwedge_{j=1}^{m} D_j'),$

where D'_j is obtained from D_j by replacing every literal p_i with $p_i = 1$ and every $\neg p_i$ with $p_i = 0$. Moreover, using $\forall p_i$, one can polynomially encode the PSPACE-complete validity problem for QBFs. A polynomial reduction of CQA$(\mathcal{A}, \mathcal{T}, q)$ to QE$(\mathcal{A} + \{0, 1\}, q')$ is given in [12] for the extension Datalog± of *OWL 2 QL*, where $|\mathcal{T}| + |q|$ steps of the chase are simulated using 0 and 1.

Theorem 1 means that there are two sources of non-determinism in OBDA with *OWL 2 QL*: finding a match in the ABox and finding a match in the remaining tree part of the canonical interpretation. It turns out that, from the complexity-theoretic point of view, these two sources have different status. Recall from [19] that query evaluation QE$(\mathcal{A}, q)$ is *not* fixed-parameter tractable if $|q|$ is regarded as a *parameter*.

Our next result shows, on the contrary, that the problem CQA$(\{A(a)\}, \mathcal{T}, q)$ is *fixed-parameter tractable* for *DL-Lite*$^{\mathcal{H}}_{core}$ TBoxes $\mathcal{T}$. This means that there exist a deterministic algorithm $\boldsymbol{A}$, a computable function f and a polynomial p such that, for any TBox $\mathcal{T}$ and CQ q, $\boldsymbol{A}$ can check whether $(\mathcal{T}, \{A(a)\}) \models q$ in time bounded by $f(|q|) \cdot p(|\mathcal{T}|)$. In a nutshell, the idea of the proof is as follows. First, given a CQ q, we construct all *tree-shaped* homomorphic images of q, the number of which is bounded by a function exponential in $|q|$ and independent of $\mathcal{T}$. Then we show that $(\mathcal{T}, \{A(a)\}) \models q$ iff at least one of those tree-shaped homomorphic images can be 'embedded' in $\mathcal{U}_{\mathcal{K}}$, and that the existence of such an embedding can be established by a dynamic programming (elimination) algorithm in time polynomial in $|\mathcal{T}|$ and $|q|$.

Theorem 3. *The problem CQA$(\{A(a)\}, \mathcal{T}, q)$ with $|q|$ a parameter is fixed-parameter tractable for DL-Lite$^{\mathcal{H}}_{core}$ TBoxes $\mathcal{T}$.*

Proof. A CQ q is *tree-shaped* if its primal graph $(\mathsf{term}(q), \{(t, t') \mid R(t, t') \in q\})$ is a tree. By a *tree reduct* of q we mean a pair (q', r), where q' is a set of atoms and $r \in \mathsf{term}(q')$ is such that the following conditions are satisfied (cf. [10]):

(tree) the query q' is tree-shaped and all of its predicate names occur in q;
(root) if $a \in \mathsf{term}(q')$ then $r = a$;
(hom) there exists a surjection $h\colon \mathsf{term}(q) \to \mathsf{term}(q')$ such that $h(a) = a$ for $a \in \mathsf{term}(q)$, $A(h(t)) \in q'$ for $A(t) \in q$, and $P(h(t), h(t')) \in q'$ for $P(t, t') \in q$.

By **(hom)**, for every $\mathcal{I}$ and every tree reduct (q', r) of q, if $\mathcal{I} \models q'$ then $\mathcal{I} \models q$.

Let (q', r) be a tree reduct of q and let $\mathcal{K} = (\mathcal{T}, \{A(a)\})$. An *embedding* of (q', r) in $\mathcal{U}_{\mathcal{K}}$ is an *injective* map $\mathfrak{a}\colon \mathsf{term}(q') \to \Delta^{\mathcal{U}_{\mathcal{K}}}$ such that $\mathcal{U}_{\mathcal{K}} \models^{\mathfrak{a}} q'$ and

(e-root) $\mathfrak{a}(t) = \mathfrak{a}(r) \cdot \sigma$, for all $t \in \mathsf{term}(q')$, i.e., $\mathfrak{a}(r)$ is located in $\mathcal{U}_{\mathcal{K}}$ nearer to its root than any other $\mathfrak{a}(t)$.

Let $\mathcal{U}_{\mathcal{K}} \models q$. Then there is a homomorphism $\mathfrak{a}$ of q in $\mathcal{U}_{\mathcal{K}}$. As $\mathcal{U}_{\mathcal{K}}$ is a tree with root $a^{\mathcal{U}_{\mathcal{K}}}$, we can construct a tree reduct (q', r) of q by taking q' to be the quotient of q under equivalence $\{(t, t') \mid \mathfrak{a}(t) = \mathfrak{a}(t')\}$ and r the equivalence class of t such that $\mathfrak{a}(t)$ is nearest to the root $a^{\mathcal{U}_{\mathcal{K}}}$. It follows that (q', r) is embeddable in $\mathcal{U}_{\mathcal{K}}$. Checking whether a tree reduct (q', r) of q is embeddable in $\mathcal{U}_{\mathcal{K}}$ can be done in time polynomial in $|\mathcal{T}|$ and $|q|$ using the interpretation $\mathcal{G}_{\mathcal{K}}$ (constructed in polynomial time in $|\mathcal{T}|$) and a standard dynamic programming algorithm [8].

Theorem 1 reflects the interaction between role inclusions and inverse roles. The observations below supplement this theorem by giving a somewhat broader picture (we remind the reader that *DL-Lite*$_{horn}$ extends *DL-Lite*$_{core}$ with concept inclusions of the form $B_1 \sqcap \dots \sqcap B_n \sqsubseteq B$, $\mathcal{EL}$ allows qualified existential restrictions and conjunctions in both sides of concept inclusions, $\mathcal{H}$ allows role inclusions and $\mathcal{I}$ inverse roles; for details see [3]):

Theorem 4. *With respect to primitive combined complexity, CQ answering is* (i) P*-complete for DL-Lite*$_{horn}$ *and* $\mathcal{ELH}$*, and* (ii) ExpTime*-complete for* $\mathcal{ELI}$*.*

Proof. The polynomial-time upper bound for *DL-Lite*$_{horn}$ and $\mathcal{ELH}$ can be obtained using the fact that, for each CQ q and each $r \in \mathsf{term}(q)$, one can construct a *unique* tree reduct of q with root r (by eliminating 'forks') [16] and then check whether it is embeddable in the generating interpretation as in the proof of Theorem 3 (see also Section 4). ExpTime-completeness for $\mathcal{ELI}$ follows from [3].

Although $\mathcal{ALC}$ and $\mathcal{ALCH}$ have no canonical interpretations (they are not Horn), a unique tree reduct for a CQ with a root exists [10], and CQ answering is ExpTime-complete for PCC; in $\mathcal{ALCI}$, we again have to consider multiple tree reducts, which makes CQ answering 2ExpTime-complete for PCC [16].

That CQ answering is in P for PCC does not mean yet that there is a polynomial rewriting q' for any CQ q and ontology $\mathcal{T}$. For instance, as CQ answering for $\mathcal{ELH}$ is P-complete for data complexity, we cannot have any first-order rewriting at all. The reason is that if we put an ABox element a to a concept A, then a TBox axiom of the form $\exists R.A \sqsubseteq B$ requires adding every ABox element b with $R(b, a)$ to B, and so on. In this case, a pre-processing of the ABox, constructing the generating interpretation, is required; see [18].

4 Polynomial Rewriting for *DL-Lite*$_{core}$

The *combined approach* to CQ answering [18, 14] first constructs the generating interpretation $\mathcal{G}_{\mathcal{K}}$ for $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, and then rewrites the given CQ q (independently of $\mathcal{A}$) to an FO query q' to be answered *over* $\mathcal{G}_{\mathcal{K}}$. An important achievement of this approach is that (*i*) $|q'| = \mathcal{O}(|q|^2 + |q| \cdot |\mathcal{T}|)$, even for *DL-Lite*$_{horn}$, and (*ii*) q' is obtained by expanding q by simple conjuncts with $=$ and without any extra variables and quantifiers. The two-step construction of $\mathcal{G}_{\mathcal{K}}$ and q' can be encoded in a polynomial non-recursive Datalog program for *DL-Lite*$_{horn}$, and a polynomial FO query for *DL-Lite*$_{core}$, which require auxiliary constants in the database domain. Here we give a polynomial FO rewriting for *DL-Lite*$_{core}$, which is based on the ideas of [14] but does not involve any constants.

Let $\mathcal{T}$ be a *DL-Lite*$_{core}$ TBox. As we do not have role inclusions, instead of $c_{[R]}$ we write c_R. Let $\mathsf{R}_{\mathcal{T}} = \{c_R \mid R \text{ a role in } \mathcal{T}\}$ and $\mathsf{R}^*_{\mathcal{T}}$ be the set of all finite words over $\mathsf{R}_{\mathcal{T}}$ (including the empty word ε). We use $\mathsf{tail}(\sigma)$ to denote the last element of $\sigma \in \mathsf{R}^*_{\mathcal{T}} \setminus \{\varepsilon\}$; by definition, $\mathsf{tail}(\varepsilon) = \varepsilon$.

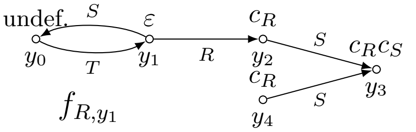
Consider a CQ $q(\boldsymbol{x})$. Without loss of generality we assume that (the primal graph of) q is connected. Let R be a role and t a term in q. A *partial* function f from $\mathsf{term}(q)$ to $(\mathsf{R}_{\mathcal{T}})^*$ is called a *tree witness for* (R, t) in q if

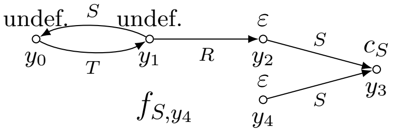
- the domain of f is minimal with respect to set-theoretic inclusion,
- $f(t) = \varepsilon$,
- for all atoms $S(s, s') \in q$ with $f(s)$ defined, we have

$$f(s') = \begin{cases} c_R, & \text{if } f(s) = \varepsilon \text{ and } S = R, \\ \sigma, & \text{if } f(s) = \sigma \cdot c_{S^-}, \\ f(s) \cdot c_S, & \text{if } f(s) \neq \varepsilon \text{ and } \mathsf{tail}(f(s)) \neq c_{S^-}. \end{cases}$$

By definition, if a tree witness for (R, t) exists then it is unique; in this case we denote it by $f_{R,t}$ and use $\mathsf{dom}\, f_{R,t}$ for the domain of $f_{R,t}$. Note that even if q contains no atom of the form $R(t, t')$, the tree witness for (R, t) exists and $f_{R,t}(t) = \varepsilon$. Denote by $q|_{R,t}$ the set of atoms of q whose terms are in $\mathsf{dom}\, f_{R,t}$. When we consider $q|_{R,t}$ as a query, we assume that all of its variables are *free.*

Informally, a tree witness $f_{R,t}$ has *root* t and *direction* R, and describes the situation where t is mapped to an ABox element a of some canonical interpretation without R-successors in the ABox. In this case, the only choice for mapping any t' in $R(t, t') \in q$ is $ac_R = a \cdot f_{R,t}(t')$. Further, any t'' in $S(t', t'') \in q$ has to be mapped to $ac_Rc_S = a \cdot f_{R,t}(t'')$, if $S \neq R^-$; however, if $S = R^-$ then t'' can only be mapped onto a, which reflects the fact that ac_R has a single R^--successor a in the canonical interpretation. To illustrate, consider the CQ $q = \{T(y_0, y_1),\ S(y_1, y_0),\ R(y_1, y_2),\ S(y_2, y_3),\ S(y_4, y_3)\}$. The tree witnesses for (R, y_1) and (S, y_4) in q exist and are as depicted below:





For (S, y_1), (T^-, y_1) and (R^-, y_2), tree witnesses do not exist.

Proposition 1. *Suppose a tree witness for (R, t) exists and $s \in \mathsf{dom}\, f_{R,t}$. If $f_{R,t}(s) \neq \varepsilon$ then a tree witness exists for every (S, s) with $\mathsf{tail}(f_{R,t}(s)) \neq c_{S^-}$. If $f_{R,t}(s) = \varepsilon$ then a tree witness exists for (S, s) with $S = R$. In either case, $\mathsf{dom}\, f_{S,s} \subseteq \mathsf{dom}\, f_{R,t}$ and $f_{R,t}(s') = f_{R,t}(s) \cdot f_{S,s}(s')$, for all $s' \in \mathsf{dom}\, f_{S,s}$.*

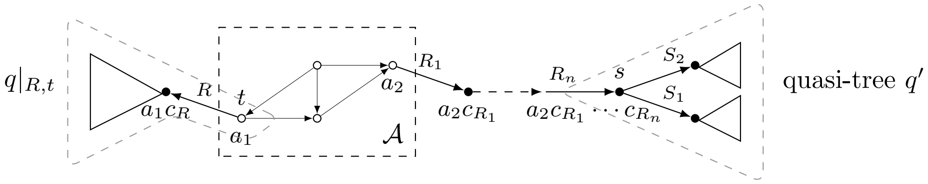
Even if a tree witness for (R, t) exists, $q|_{R,t}$ is not necessarily a tree-shaped query. Define a relation $\equiv_R$ as the set of all pairs (t, s) such that a tree witness for (R, t) exists and $f_{R,t}(s) = \varepsilon$. By Proposition 1, $\equiv_R$ is an equivalence relation (on its domain). By taking the quotient of $q|_{R,t}$ under $\equiv_R$, we obtain a tree reduct of $q|_{R,t}$ (cf. [17]). We call q a *quasi-tree with root* $t \in \mathsf{term}(q)$ if a tree witness for (R, t) exists for *all* directions R and $\bigcup_R \mathsf{dom}\, f_{R,t} = \mathsf{term}(q)$.

Proposition 2. *Suppose q is not a quasi-tree and tree witnesses exist for (R_1, t_1) and (R_2, t_2). If $f_{R_1,t_1}(t_2)$ is defined, $f_{R_1,t_1}(t_2) \neq \varepsilon$ then $\mathsf{dom}\, f_{R_2,t_2} \subsetneq \mathsf{dom}\, f_{R_1,t_1}$ and $f_{R_1,t_1}(s) = f_{R_1,t_1}(t_2) \cdot f_{R_2,t_2}(s)$, for all $s \in \mathsf{dom}\, f_{R_2,t_2}$.*

We are now in a position to introduce the ingredients of our polynomial rewriting. Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ and $q(\boldsymbol{x}) = \exists \boldsymbol{y}\, \varphi(\boldsymbol{x}, \boldsymbol{y})$. Consider an atom $B(t)$ for a concept B. Then

$$\mathsf{ext}_B(x) \quad = \bigvee_{\text{concept } B' \text{ s.t. } \mathcal{T}\models B'\sqsubseteq B} B'(x) \quad \lor \bigvee_{\text{role } R \text{ s.t. } \mathcal{T}\models \exists R\sqsubseteq B} \exists w\, R(x,w)$$

gives the answers to $B(t)$ over ABox $\mathcal{A}$: for every $a \in \mathsf{Ind}(\mathcal{A})$, we have $\mathcal{U}_{\mathcal{K}} \models B(a)$ iff $\mathcal{A} \models \mathsf{ext}_B(a)$. Note that, for all other elements σ in the domain $\Delta^{\mathcal{U}_{\mathcal{K}}}$ of $\mathcal{U}_{\mathcal{K}}$, we have $\mathcal{U}_{\mathcal{K}} \models B(\sigma)$ iff $\mathcal{T} \models \exists T^- \sqsubseteq B$, where $\mathsf{tail}(\sigma) = c_T$.



Consider now an atom $R(t,t') \in q$ and the ways its terms can be mapped in $\mathcal{U}_{\mathcal{K}}$.
1. If both t and t' are mapped to ABox elements a, a' then $\mathcal{U}_{\mathcal{K}} \models R(a,a')$ iff $R(a,a') \in \mathcal{A}$ because $\mathcal{U}_{\mathcal{K}}$ inherits the binary relations from $\mathcal{A}$.

2. If t is mapped to an ABox element a and t' to an 'anonymous' element in $\Delta^{\mathcal{U}_{\mathcal{K}}} \setminus \mathsf{Ind}(\mathcal{A})$, then $R(t,t')$ can only be true if (i) $a \leadsto_{\mathcal{K}} c_R$, (ii) a tree witness for (R,t) exists, and (iii) $q|_{R,t}$ can be embedded into the sub-tree of $\mathcal{U}_{\mathcal{K}}$ beginning with the edge (a, ac_R); see the left-hand side of the picture above. Condition (i) can be defined by the formula

$$\mathsf{wt}_R(x) \quad = \quad \mathsf{ext}_{\exists R}(x) \land \neg\exists w\, R(x,w).$$

For all R and $a \in \mathsf{Ind}(\mathcal{A})$, we have $\mathcal{A} \models \mathsf{wt}_R(a)$ iff $a \leadsto_{\mathcal{K}} c_R$ (i.e., $ac_R \in \Delta^{\mathcal{U}_{\mathcal{K}}}$). For condition (iii), consider the conjunction $\mathsf{treeA}^q_{R,t}(x)$ of the formulas:

($\mathbf{t}_0$) $\mathsf{ext}_A(x)$, for all $A(s) \in q|_{R,t}$ with $f_{R,t}(s) = \varepsilon$;
($\mathbf{t}_1$) $\top$ if $\mathcal{T} \models \exists T^- \sqsubseteq A$ and $\bot$ otherwise, for all $A(s) \in q|_{R,t}$, $\mathsf{tail}(f_{R,t}(s)) = c_T$;
($\mathbf{t}_2$) $\top$ if $\mathcal{T} \models \exists T^- \sqsubseteq \exists S$ and $\bot$ otherwise, for $S(s,s') \in q|_{R,t}$, $\mathsf{tail}(f_{R,t}(s)) = c_T$.

One can show that $\mathcal{A} \models \mathsf{wt}_R(a) \land \mathsf{treeA}^q_{R,t}(a)$ iff $\mathcal{U}_{\mathcal{K}} \models^{\mathfrak{a}} q|_{R,t}$ for an assignment $\mathfrak{a}$ such that $\mathfrak{a}(s) = a \cdot f_{R,t}(s)$, for all $s \in \mathsf{dom}\, f_{R,t}$.

3. If both t, t' are mapped to anonymous elements in $\Delta^{\mathcal{U}_{\mathcal{K}}} \setminus \mathsf{Ind}(\mathcal{A})$, then two more cases need consideration.
3.1. Suppose first that there is a tree witness for some (S,s) such that s is mapped to an ABox element a with $a \leadsto_{\mathcal{K}} c_S$, (iv) both t and t' are in $\mathsf{dom}\, f_{S,s}$, and (v) all the terms $s' \in \mathsf{dom}\, f_{S,s}$ with $f_{S,s}(s') \neq \varepsilon$ are existentially quantified variables in q (only existential variables can be mapped to anonymous elements). In this case, as we observed above, $R(t,t')$ is true in $\mathcal{U}_{\mathcal{K}}$ if the formula

$$\mathsf{wt}_S(s) \land \mathsf{treeA}^q_{S,s}(s) \land \textstyle\bigwedge_{s \equiv_S s'} (s = s')$$

is true in $\mathcal{A}$ under an assignment $\mathfrak{a}$ such that $\mathfrak{a}(s') = a \cdot f_{S,s}(s')$, for $s' \in \mathsf{dom}\, f_{S,s}$. The disjunction of all such formulas for (S,s) satisfying (iv)–(v) depends only on the choice of terms t, t' and will be denoted by $\mathsf{attached\text{-}tree}_{t,t'}(\boldsymbol{x}, \boldsymbol{y})$. (This case is a generalisation of Case 2.)

3.2. Thus, it remains to consider the case (shown in the right-hand side of the picture) where the whole query is mapped to the anonymous part of $\mathcal{U}_\mathcal{K}$. Then q a quasi-tree and all terms in q are existentially quantified variables that are mapped to the sub-tree of $\mathcal{U}_\mathcal{K}$ generated by some ABox element a. More precisely, $a \in \mathsf{Ind}(\mathcal{A})$ generates a sequence of the form $a \leadsto_\mathcal{K} c_{R_1} \leadsto_\mathcal{K} \cdots \leadsto_\mathcal{K} c_{R_n}$, q has a root s (i.e., $\mathsf{term}(q) = \bigcup_S \mathsf{dom}\, f_{S,s}$), s is mapped to $\sigma = ac_{R_1} \cdots c_{R_n}$, while all other terms s' are mapped to $\sigma \cdot f_{S,s}(s')$. The latter condition can be captured by a formula similar to the one in the previous case. The difference is that now we begin with σ, $\mathsf{tail}(\sigma) = c_{R_n}$ (rather than a). To cope with this, consider the union q' of q and $\{R_n(v,s)\}$, for a fresh variable v, and let $\mathsf{treeT}^q_{c_{R_n},s}$ be $\mathsf{treeA}^{q'}_{R_n,v}$, where the tree witnesses are computed in query q'. Note that $\mathsf{treeT}^q_{c_{R_n},s}$ is a sentence because q' has no atoms for item $(\mathbf{t}_0)$. We denote by $\mathsf{detached\text{-}tree}$ the disjunction of sentences of the form

$$\exists w\, \mathsf{wt}_{R_1}(w) \land \mathsf{treeT}^q_{c_{R_n},s}$$

for all roots s of q and all pairs of roles R_1, R_n such that there are $R_2, \dots, R_{n-1}$ with $\mathcal{T} \models \exists R_i^- \sqsubseteq \exists R_{i+1}$ and $R_{i+1} \neq R_i^-$, for $1 \leq i < n$; if q is not a quasi-tree containing only existentially quantified variables, we set $\mathsf{detached\text{-}tree} = \bot$.

Denote by q^* the result of replacing each $A(t)$ and $P(t,t')$ in q with

$$A^*(t) = \mathsf{ext}_A(t) \lor \mathsf{attached\text{-}tree}_{t,t}(\boldsymbol{x},\boldsymbol{y}) \lor \mathsf{detached\text{-}tree},$$
$$P^*(t,t') = P(t,t') \lor \mathsf{attached\text{-}tree}_{t,t'}(\boldsymbol{x},\boldsymbol{y}) \lor \mathsf{detached\text{-}tree},$$

respectively. Note that these formulas depend not only on the predicate name but also on the terms in the atom. The length of q^* is $\mathcal{O}(|q|^2 \cdot |\mathcal{T}|^3)$ and can be made $\mathcal{O}(|q|^2 \cdot |\mathcal{T}|)$ if the sentence $\mathsf{detached\text{-}tree}$ is computed separately (in fact, for the majority of queries, e.g., queries with answer variables, it is simply $\bot$).

Theorem 5. *$\mathcal{U}_\mathcal{K} \models^\mathfrak{a} q(\boldsymbol{x})$ iff $\mathcal{A} \models^\mathfrak{a} q^*(\boldsymbol{x})$, whenever $\mathfrak{a}(x) \in \mathsf{Ind}(\mathcal{A})$ for all $x \in \boldsymbol{x}$.*

The rewriting above can also be adapted to *DL-Lite*$_{horn}$ and even *DL-Lite*$^\mathcal{N}_{horn}$ under the UNA. In this case, however, we need non-recursive Datalog programs to define the predicates $\mathsf{ext}_B(x)$; for details, see [14]. The non-recursive Datalog queries can be transformed to unions of CQs, but at the expense of exponential blowup. The problem whether a polynomial-size FO rewriting (without additional constants as in [12]) exists for *DL-Lite*$_{horn}$ is still open (and equivalent to the complexity problem 'LogSpace = P?').

5 Discussion

FO reducibility (or AC^0 data complexity) does not seem to provide enough information to judge whether a DL is suitable for OBDA. When measuring the complexity of query evaluation in database systems, it is usually assumed that queries are negligibly small compared to data. Thus, it makes sense to consider data complexity [24], which takes account of the data but ignores the query. A more subtle analysis [19] shows, however, that the obvious time $|q| \cdot |\mathcal{A}|^{|q|}$ required to check $\mathcal{A} \models q$ cannot be reduced to $f(|q|) \cdot p(|\mathcal{A}|)$, for any computable function f and polynomial p: $\mathrm{QE}(\mathcal{A}, q)$ is $W[1]$-complete for parameterised complexity, $|q|$ being a parameter. The success of database systems—despite fixed-parameter intractability—seems to imply that optimisation techniques are indispensable, that the 'real-world' queries are small and of 'special' form. In OBDA, the latter does not hold as the rewritten queries can be large and complex. However, data complexity does not differentiate among, e.g., *DL-Lite*$_{core}$, *OWL 2 QL* and the language of sticky sets of TGDs [5], all of which are in AC^0 for data complexity, while the primitive combined complexity, reflecting the size of the rewriting, ranges from P to NP and further to ExpTime. Another explanation of the database efficiency is that we only use queries with a bounded number of variables, in which case query evaluation is P-complete for combined complexity [25]. However, query rewritings may substantially increase the number of variables (for example, a CQ q is rewritten in [12] into a query with $\mathcal{O}(N \cdot \log N)$ auxiliary binary variables, where $N = |T| + |q|$).

The W3C recommendation (`www.w3.org/TR/owl2-profiles`) for OBDA is to reduce it to query evaluation in database systems. Two drawbacks of this recommendation are that it (i) disregards the complexity of possible reductions, and (ii) excludes some useful DLs from consideration. As we saw above, rewritings of CQs in *OWL 2 QL* cannot be done in polynomial time without adding extra constants, variables and quantifiers as in [12]. One might argue that, in the real-world ontologies, role inclusions do not interact with inverse roles in as sophisticated way as in Theorem 1, but then more research is needed to support this argument. A number of 'lightweight' DLs such as $\mathcal{ELH}$ or *DL-Lite*$_{horn}^{(\mathcal{HF})}$ [2] are deemed not suitable for OBDA because they are P-complete for data complexity. Recall that both of these logics are P-complete for primitive combined complexity (vs. NP in the case of *OWL 2 QL*). The combined approach to OBDA [18, 14, 15] resolves this issue by expanding the data at a pre-processing step and then rewriting and answering CQs. The expansion is linear in $|\mathcal{A}|$ and can be done by the database system itself; the size of the rewritten query for $\mathcal{EL}$ and *DL-Lite*$_{horn}^{\mathcal{F}}$ is only quadratic (for *OWL 2 QL*, it is still exponential).

In this paper, we do not touch on the problem of representing ABoxes in database systems, where usually GLAV mappings are used to connect data sources to ontologies. Such mappings introduce some problems as tuples in the same relation can come from different data sources. Also, they provide certain information on the completeness of concepts and roles, which can (and should) be exploited in order to minimise the rewritings [22]. Finally, with so many languages and rewritings for OBDA suggested, it looks like the time is ripe for comprehensive experiments that could clarify the future of OBDA with DLs.

Acknowledgments: supported by the U.K. EPSRC grant EP/H05099X/1.

References

1. Acciarri, A., Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Palmieri, M., Rosati, R.: QuOnto: Querying ontologies. In: Proc. AAAI, 1670–1671 (2005)

2. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The DL-Lite family and relations. J. Artificial Intelligence Research 36, 1–69 (2009)
3. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope further. In: Clark, K., Patel-Schneider, P.F. (eds.) In: Proc. OWLED DC (2008)
4. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook. Cambridge University Press (2003)
5. Calì, A., Gottlob, G., Pieris, A.: Advanced processing for ontological queries. In: Proc. VLDB 3(1), 554–565 (2010)
6. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Data complexity of query answering in description logics. In: Proc. KR. pp. 260–270 (2006)
7. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. J. Automated Reasoning 39(3), 385–429 (2007)
8. Dasgupta, S., Papadimitriou, C., Vazirani, U.V.: Algorithms. McGraw-Hill (2008)
9. Dolby, J., Fokoue, A., Kalyanpur, A., Ma, L., Schonberg, E., Srinivas, K., Sun, X.: Scalable grounded conjunctive query evaluation over large and expressive knowledge bases. In: Proc. ISWC. LNCS, vol. 5318, pp. 403–418. Springer (2008)
10. Eiter, T., Lutz, C., Ortiz, M., Šimkus, M.: Query answering in description logics: The knots approach. In: Proc. WoLLIC. LNCS, vol. 5514. Springer (2009)
11. Gottlob, G., Orsi, G., Pieris, A.: Ontological queries: Rewriting and optimization. In: Proc. ICDE. (2011)
12. Gottlob, G., Schwentick, T.: Rewriting ontological queries into small nonrecursive Datalog programs. In: Proc. DL. (2011)
13. Heymans, S., Ma, L., *et al.*: Ontology reasoning with large data repositories. In: Ontology Management, pp. 89–128. Springer (2008)
14. Kontchakov, R., Lutz, C., Toman, D., Wolter, F., Zakharyaschev, M.: The combined approach to query answering in DL-Lite. In: Proc. KR (2010)
15. Kontchakov, R., Lutz, C., Toman, D., Wolter, F., Zakharyaschev, M.: The combined approach to ontology-based data access. In: Proc. IJCAI (2011)
16. Lutz, C.: Inverse roles make conjunctive queries hard. In: Proc. DL. CEUR Workshop Proceedings, vol. 250. (2007)
17. Lutz, C.: The complexity of conjunctive query answering in expressive description logics. In: Proc. IJCAR. pp. 179–193. LNAI, vol. 5195. Springer (2008)
18. Lutz, C., Toman, D., Wolter, F.: Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In: Proc. IJCAI. pp. 2070–2075 (2009)
19. Papadimitriou, C.H., Yannakakis, M.: On the complexity of database queries. J. Comput. Syst. Sci. 58(3), 407–427 (1999)
20. Pérez-Urbina, H., Motik, B., Horrocks, I.: A comparison of query rewriting techniques for *DL-Lite*. In: Proc. DL. CEUR Workshop Proceedings, vol. 477. (2009)
21. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking data to ontologies. J. on Data Semantics X, 133–173 (2008)
22. Rodríguez-Muro M., Calvanese, D.: Dependencies to optimize ontology-based data access. In: Proc. DL. (2011)
23. Rosati, R., Almatelli, A.: Improving query answering over *DL-Lite* ontologies. In: Proc. KR (2010)
24. Vardi, M.: The complexity of relational query languages (extended abstract). In: Proc. STOC. pp. 137–146 (1982)
25. Vardi, M.: On the complexity of bounded-variable queries (extended abstract). In: Proc. PODS. pp. 266–276 (1995)