Long Rewritings, Short Rewritings

S. Kikot[1], R. Kontchakov[1], V. Podolskii[2], and M. Zakharyaschev[1]

[1] Department of Computer Science and Information Systems
Birkbeck, University of London, U.K.
{kikot,roman,michael}@dcs.bbk.ac.uk
[2] Steklov Mathematical Institute, Moscow, Russia
podolskii@mi.ras.ru

1 Introduction

An ontology language $\mathcal{L}$ is said to enjoy *FO-rewritability* if any conjunctive query (CQ) $\boldsymbol{q}$ over any ontology $\mathcal{T}$, given in $\mathcal{L}$, can be transformed into an FO-formula $\boldsymbol{q}'$ such that, for any data $\mathcal{A}$, all answers to $\boldsymbol{q}$ over the knowledge base $(\mathcal{T}, \mathcal{A})$ can be found by querying $\boldsymbol{q}'$ over $\mathcal{A}$ using a standard relational database management system (RDBMS). Ontology languages with this property include the *OWL 2 QL* profile of *OWL 2*, which is based on description logics of the *DL-Lite* family [7, 16, 2], and fragments of Datalog$^{\pm}$ such as linear or sticky TGDs [5, 6]. Various rewriting techniques have been implemented in the systems QuOnto [1], REQUIEM [15], Presto [22], Nyaya [8], IQAROS[3] and Quest.[4]

The idea of using languages with FO-rewritability for ontology-based data access (OBDA) relies on the empirical fact that RDBMSs are usually very efficient in practice. However, the first rewritings of CQs over *OWL 2 QL* ontologies [7, 15] turned out to be too lengthy even for modern RDBMSs. The attempts to employ various optimisation techniques still produced rewritings of exponential size in the worst case: $O((|\mathcal{T}| \cdot |\boldsymbol{q}|)^{|\boldsymbol{q}|})$ [22, 8, 20, 21]. The alternative two-step combined approach [14, 13]—first expand the data by applying the ontology axioms to the data and introducing (some of) the missing individuals, and only then rewrite the query over the expanded data—resulted in a simple polynomial rewriting only for the fragment of *OWL 2 QL* without role inclusions; for the full language, the rewriting remained exponential. Two seemingly contradictory results, presented at DL 2011, added more spice to the quest for short rewritings: [9] showed that one can construct, in polynomial time, a nonrecursive Datalog (NDL) rewriting for some fragments of Datalog$^{\pm}$ containing *OWL 2 QL*, while [11] argued that no FO-rewriting for *OWL 2 QL* can be constructed in polynomial time.

The aim of this paper is twofold. First, we investigate the worst-case size of FO- and NDL-rewritings for CQs over *OWL 2 QL* ontologies. We distinguish between 'pure' rewritings, which can use the signature of the original query and ontology as well as $=$, $\neq$ (cf. [7]), and 'impure' rewritings, where other means such as new constants are allowed. Here is a summary of the obtained results:

[3] http://code.google.com/p/iqaros/
[4] http://obda.inf.unibz.it/protege-plugin/quest/quest.html

(1) An exponential blow-up is unavoidable for pure positive existential (PE) rewritings and pure NDL-rewritings; pure FO-rewritings can blow-up superpolynomially unless NP $\subseteq$ P/poly.
(2) Pure NDL-rewritings are in general exponentially more succinct than pure PE-rewritings.
(3) Pure FO-rewritings can be superpolynomially more succinct than pure PE-rewritings.
(4) Impure PE- and NDL-rewritings can always be made polynomial, and so they are exponentially more succinct than pure PE- and NDL-rewritings, respectively.

(1)–(3) are proved by establishing connections between pure rewritings for CQs over *OWL 2 QL* ontologies and circuits for monotone Boolean functions. In a nutshell, we show that CQs and *OWL 2 QL* ontologies can encode such problems as the existence of a k-clique in a graph with n vertices whose edges are given by a single-element ABox. The polynomial PE-rewriting in (4) is similar to the NDL-rewriting of [9]: using two extra constants, $=$ and polynomially many new existentially quantified variables, one can guess a relevant part of the canonical model of $\mathcal{T}$ in the rewritten query. The difference between the resulting impure PE-rewritings and exponential-size pure PE-rewritings is of the same kind as the difference between deterministic and nondeterministic Boolean circuits.

Our second aim is to analyse the causes behind long rewritings and whether they occur in real-world queries and ontologies. As a result, we suggest some short rewritings that cover most practical cases.

Omitted proofs can be found in [10] and the full version of [12].

2 Queries over *OWL 2 QL* Ontologies

The language of *OWL 2 QL* is defined by the following grammar:[5]

$$\begin{array}{lcl} R & ::= & P_i \mid P_i^-, \\ B & ::= & \bot \mid A_i \mid \exists R, \\ C & ::= & B \mid \exists R.B, \end{array}$$

where the A_i are concept names and the P_i are role names. An *OWL 2 QL TBox*, $\mathcal{T}$, is a finite set of *inclusions* of the form $B \sqsubseteq C$, $R_1 \sqsubseteq R_2$, $B_1 \sqcap B_2 \sqsubseteq \bot$ and $R_1 \sqcap R_2 \sqsubseteq \bot$. Note that concepts of the form $\exists R.B$ can only occur in the right-hand side of concept inclusions. An *ABox*, $\mathcal{A}$, is a finite set of *assertions* of the form $A_k(a_i)$ and $P_k(a_i, a_j)$, where a_i, a_j are individual names. $\mathcal{T}$ and $\mathcal{A}$ together form the *knowledge base* (KB) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. The semantics is defined in the usual way, based on interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with domain $\Delta^{\mathcal{I}}$ and interpretation function $\cdot^{\mathcal{I}}$. The set of individuals in $\mathcal{A}$ is denoted by $\mathsf{ind}(\mathcal{A})$; $\mathcal{I}_{\mathcal{A}}$ is the interpretation with domain $\mathsf{ind}(\mathcal{A})$ such that, for any concept or role E and any $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$, we have $\mathcal{I}_{\mathcal{A}} \models E(\boldsymbol{a})$ iff $E(\boldsymbol{a}) \in \mathcal{A}$. We write $E_1 \sqsubseteq_{\mathcal{T}} E_2$ if $\mathcal{T} \models E_1 \sqsubseteq E_2$; and we set $[E] = \{E' \mid E \sqsubseteq_{\mathcal{T}} E' \text{ and } E' \sqsubseteq_{\mathcal{T}} E\}$.

[5] We do not consider data properties, attributes and role (ir)reflexivity constraints.

A *conjunctive query* (CQ) $\boldsymbol{q}(\boldsymbol{x})$ is a formula $\exists \boldsymbol{y}\, \varphi(\boldsymbol{x}, \boldsymbol{y})$, where φ is a conjunction of atoms of the form $A_k(t_1)$ and $P_k(t_1, t_2)$, and each t_i is a *term* (an individual or a variable from $\boldsymbol{x}$, $\boldsymbol{y}$). A tuple $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$ is a *certain answer* to $\boldsymbol{q}(\boldsymbol{x})$ over $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if $\mathcal{I} \models \boldsymbol{q}(\boldsymbol{a})$ for all $\mathcal{I} \models \mathcal{K}$; then we write $\mathcal{K} \models \boldsymbol{q}(\boldsymbol{a})$.

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 $\mathcal{C}_\mathcal{K}$ such that, for all CQs $\boldsymbol{q}(\boldsymbol{x})$ and $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$, we have $\mathcal{K} \models \boldsymbol{q}(\boldsymbol{a})$ iff $\mathcal{C}_\mathcal{K} \models \boldsymbol{q}(\boldsymbol{a})$. The interpretation $\mathcal{C}_\mathcal{K}$, called the *canonical model* of $\mathcal{K}$, can be constructed as follows. For each pair $[R], [B]$ with $\exists R.B$ in $\mathcal{T}$ (we assume $\exists R$ is just a shorthand for $\exists R.\top$), we introduce a fresh symbol $w_{[RB]}$ and call it the *witness for* $\exists R.B$. We write $\mathcal{K} \models C(w_{[RB]})$ if $\exists R^- \sqsubseteq_\mathcal{T} C$ or $B \sqsubseteq_\mathcal{T} C$. Define a *generating relation*, $\rightsquigarrow$, on the set of these witnesses together with $\mathsf{ind}(\mathcal{A})$ by taking:

- $a \rightsquigarrow w_{[RB]}$ if $a \in \mathsf{ind}(\mathcal{A})$, $[R]$ and $[B]$ are $\sqsubseteq_\mathcal{T}$-minimal such that $\mathcal{K} \models \exists R.B(a)$ and there is no $b \in \mathsf{ind}(\mathcal{A})$ with $\mathcal{K} \models R(a, b) \land B(b)$;
- $w_{[R'B']} \rightsquigarrow w_{[RB]}$ if $u \rightsquigarrow w_{[R'B']}$, for some u, $[R]$ and $[B]$ are $\sqsubseteq_\mathcal{T}$-minimal with $\mathcal{K} \models \exists R.B(w_{[R'B']})$ and it is not the case that $R' \sqsubseteq_\mathcal{T} R^-$ and $\mathcal{K} \models B(u)$.

If $a \rightsquigarrow w_{[R_1B_1]} \rightsquigarrow \cdots \rightsquigarrow w_{[R_nB_n]}$, $n \geq 0$, then we say that a *generates the path* $aw_{[R_1B_1]} \cdots w_{[R_nB_n]}$. Denote by $\mathsf{path}_\mathcal{K}(a)$ the set of paths generated by a, and by $\mathsf{tail}(\pi)$ the last element in $\pi \in \mathsf{path}_\mathcal{K}(a)$. $\mathcal{C}_\mathcal{K}$ is defined by taking:

$$
\begin{aligned}
\Delta^{\mathcal{C}_\mathcal{K}} &= \bigcup\nolimits_{a \in \mathsf{ind}(\mathcal{A})} \mathsf{path}_\mathcal{K}(a), \qquad a^{\mathcal{C}_\mathcal{K}} \;=\; a, \text{ for } a \in \mathsf{ind}(\mathcal{A}),\\
A^{\mathcal{C}_\mathcal{K}} &= \{\pi \in \Delta^{\mathcal{C}_\mathcal{K}} \mid \mathcal{K} \models A(\mathsf{tail}(\pi))\},\\
P^{\mathcal{C}_\mathcal{K}} &= \{(a, b) \in \mathsf{ind}(\mathcal{A}) \times \mathsf{ind}(\mathcal{A}) \mid \mathcal{K} \models P(a, b)\} \cup\\
&\quad \{(\pi, \pi \cdot w_{[RB]}) \mid \mathsf{tail}(\pi) \rightsquigarrow w_{[RB]},\ R \sqsubseteq_\mathcal{T} P\} \cup\\
&\quad \{(\pi \cdot w_{[RB]}, \pi) \mid \mathsf{tail}(\pi) \rightsquigarrow w_{[RB]},\ R \sqsubseteq_\mathcal{T} P^-\}.
\end{aligned}
$$

Theorem 1 ([7, 13]). *For every OWL 2 QL KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, every CQ $\boldsymbol{q}(\boldsymbol{x})$ and every $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$, $\mathcal{K} \models \boldsymbol{q}(\boldsymbol{a})$ iff $\mathcal{C}_\mathcal{K} \models \boldsymbol{q}(\boldsymbol{a})$.*

Given a CQ $\boldsymbol{q}(\boldsymbol{x})$ and a TBox $\mathcal{T}$, a first-order formula $\boldsymbol{q}'(\boldsymbol{x})$, possibly with $=$ and $\neq$, is called an *FO-rewriting for $\boldsymbol{q}(\boldsymbol{x})$ and $\mathcal{T}$* if, for any ABox $\mathcal{A}$ and any $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$, we have $(\mathcal{T}, \mathcal{A}) \models \boldsymbol{q}(\boldsymbol{a})$ iff $\mathcal{I}_\mathcal{A} \models \boldsymbol{q}'(\boldsymbol{a})$. If $\boldsymbol{q}'$ is an FO-rewriting of the form $\exists \boldsymbol{y}\, \varphi(\boldsymbol{x}, \boldsymbol{y})$, where φ is built from atoms using only $\land$ and $\lor$, then we call $\boldsymbol{q}'(\boldsymbol{x})$ a *positive existential rewriting for $\boldsymbol{q}(\boldsymbol{x})$ and $\mathcal{T}$* (or a *PE-rewriting*, for short). We say that $\boldsymbol{q}'$ is *pure* if it does not contain constants that do not occur in $\boldsymbol{q}$ (such constants are interpreted by fresh individuals added to $\mathcal{I}_\mathcal{A}$). The *size* $|\boldsymbol{q}'|$ of $\boldsymbol{q}'$ is the number of symbols in $\boldsymbol{q}'$.

We also consider rewritings in the form of nonrecursive Datalog queries. We remind the reader that a *Datalog program*, Π, is a finite set of Horn clauses $\forall \boldsymbol{x}\, (A_1 \land \cdots \land A_m \to A_0)$, where each A_i is an atom of the form $P(t_1, \ldots, t_l)$ and each t_j is either a variable from $\boldsymbol{x}$ or a constant. A_0 is called the *head* of the clause, and $A_1, \ldots, A_m$ its *body*. All variables occurring in the head must also occur in the body. A predicate P *depends* on a predicate Q in Π if Π contains a clause whose head is P and whose body contains Q. Π is called *nonrecursive* if

this dependence relation for Π is acyclic. A *nonrecursive Datalog query* consists of a nonrecursive Datalog program Π and a *goal* G, which is just a predicate. A tuple $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$ is a *certain answer* to (Π, G) over $\mathcal{A}$ if $\Pi, \mathcal{A} \models G(\boldsymbol{a})$. The *size* $|\Pi|$ of Π is the number of symbols in it. We distinguish between *pure* and *impure* Datalog queries [3]. In a *pure query* (Π, G), the clauses in Π do not contain constant symbols in their heads. One reason for considering only pure queries in OBDA is that impure ones can add new facts to the database that do not follow from the background ontology. Impure queries are known to be more succinct than pure ones.

Given a CQ $\boldsymbol{q}(\boldsymbol{x})$ and an *OWL 2 QL* TBox $\mathcal{T}$, a pure nonrecursive Datalog query (Π, G) is called a *nonrecursive Datalog rewriting for* $\boldsymbol{q}(\boldsymbol{x})$ *and* $\mathcal{T}$ (or an *NDL-rewriting*, for short) if, for any ABox $\mathcal{A}$ and any $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$, we have $(\mathcal{T}, \mathcal{A}) \models \boldsymbol{q}(\boldsymbol{a})$ iff $\Pi, \mathcal{A} \models G(\boldsymbol{a})$. If Π does not contain constants that do not occur in $\boldsymbol{q}$ then we say that the NDL-rewriting (Π, G) is *pure*.

3 Query Rewritings and Boolean Circuits

To establish results (1)–(3) on the size of rewritings mentioned in the introduction, we show how the problem of constructing circuits that compute monotone Boolean functions can be reduced to the problem of finding pure FO- and NDL-rewritings for CQs over *OWL 2 QL* ontologies.

Our reduction proceeds in three steps. First, we take any family $f^1, f^2, \dots$ of *monotone* Boolean functions in NP, where $f^n \colon \{0,1\}^n \to \{0,1\}$, a polynomial p and a family $\mathbf{C}^1, \mathbf{C}^2, \dots$ of polynomial-size circuits such that $f^n(\boldsymbol{\alpha}) = 1$ iff $\mathbf{C}^n(\boldsymbol{\alpha}, \boldsymbol{\beta}) = 1$, for some $\boldsymbol{\beta} \in \{0,1\}^{p(n)}$. Using the Tseitin transformation [23], we construct a polynomial-size CNF θ_{f^n} that computes f^n in the following sense:

Lemma 1. *If f^n is monotone then $\varphi^{\boldsymbol{\alpha}}_{f^n} = \big(\bigwedge_{\alpha_j = 0} \neg x_j\big) \wedge \theta_{f^n}$ is satisfiable iff $f^n(\boldsymbol{\alpha}) = 1$, for all $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n) \in \{0,1\}^n$.*

Let φ_{f^n} be $\varphi^{\boldsymbol{\alpha}}_{f^n}$ for $\boldsymbol{\alpha} = (0, \dots, 0)$. It should be clear that $\varphi^{\boldsymbol{\alpha}}_{f^n}$ is obtained from φ_{f^n} by removing the clauses $\neg x_j$ for which the jth component of $\boldsymbol{\alpha}$ is 1.

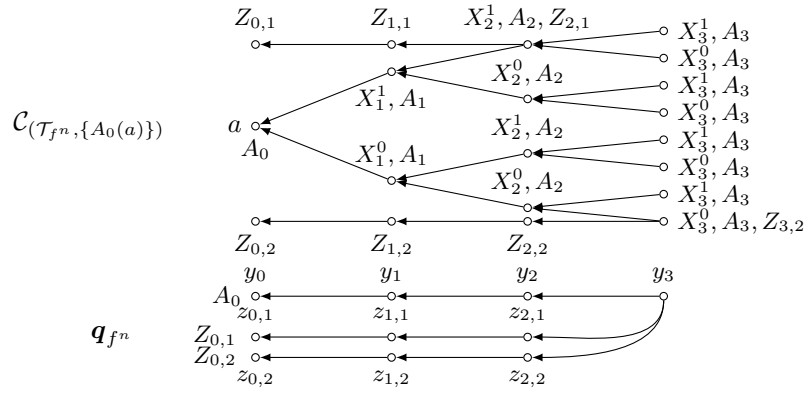
The second step is to encode the φ_{f^n} by means of TBoxes $\mathcal{T}_{f^n}$ and CQs $\boldsymbol{q}_{f^n}$. Let $p_1, \dots, p_N$ be the propositional variables and $C_1, \dots, C_d$ the clauses in φ_{f^n}. Then $\mathcal{T}_{f^n}$ contains the following concept inclusions, for $1 \le i \le N$, $1 \le j \le d$:

$$
\begin{array}{lll}
A_{i-1} \sqsubseteq \exists P^-.X^\ell_i, \quad X^\ell_i \sqsubseteq A_i, \quad \text{for } \ell = 0, 1, & & X^0_i \sqsubseteq Z_{i,j} \quad \text{if} \quad \neg p_i \in C_j, \\
Z_{i,j} \sqsubseteq \exists P.Z_{i-1,j}, & & X^1_i \sqsubseteq Z_{i,j} \quad \text{if} \quad p_i \in C_j, \\
A_0 \sqcap A_i \sqsubseteq \bot, \quad A_0 \sqcap \exists P \sqsubseteq \bot, \quad A_0 \sqcap Z_{i,j} \sqsubseteq \bot, & \text{if } (i,j) \notin \{(0,1), \dots, (0,n)\}, &
\end{array}
$$

and the CQ is defined as follows:

$$
\boldsymbol{q}_{f^n} = \exists \boldsymbol{y}\, \exists \boldsymbol{z} \Big[A_0(y_0) \;\wedge\; \bigwedge_{i=1}^{N} P(y_i, y_{i-1}) \;\wedge\; \bigwedge_{j=1}^{d} \Big(P(y_N, z_{N-1,j}) \;\wedge\; \bigwedge_{i=1}^{N-1} P(z_{i,j}, z_{i-1,j}) \;\wedge\; Z_{0,j}(z_{0,j}) \Big) \Big],
$$

where $\boldsymbol{y} = (y_0, \dots, y_N)$ and $\boldsymbol{z} = (z_{0,1}, \dots, z_{N-1,1}, \dots, z_{0,d}, \dots, z_{N-1,d})$. The size of $\mathcal{T}_{f^n}$ and $\boldsymbol{q}_{f^n}$ is $O(|\mathbf{C}^n|^2)$. Note that $\mathcal{T}_{f^n}$ is acyclic and $\boldsymbol{q}_{f^n}$ is tree-shaped and has no answer variables. The canonical model $\mathcal{C}_{(\mathcal{T}_{f^n},\{A_0(a)\})}$ of $(\mathcal{T}_{f^n}, \{A_0(a)\})$ and the query $\boldsymbol{q}_{f^n}$ are illustrated below.



For each $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n) \in \{0,1\}^n$, we define the following ABox:

$$\mathcal{A}_{\boldsymbol{\alpha}} \quad = \quad \big\{A_0(a)\big\} \quad \cup \quad \big\{Z_{0,j}(a) \mid 1 \leq j \leq n \text{ and } \alpha_j = 1\big\}.$$

Lemma 2. *$(\mathcal{T}_{f^n}, \mathcal{A}_{\boldsymbol{\alpha}}) \models \boldsymbol{q}_{f^n}$ iff $\varphi^{\boldsymbol{\alpha}}_{f^n}$ is satisfiable, for $\boldsymbol{\alpha} \in \{0,1\}^n$.*

To complete our reduction, we show that rewritings for $\boldsymbol{q}_{f^n}$ and $\mathcal{T}_{f^n}$ can be turned into Boolean circuits computing f^n.

Lemma 3. (i) *Suppose $\boldsymbol{q}'_{f^n}$ is a pure FO- (PE-) rewriting for $\mathcal{T}_{f^n}$ and $\boldsymbol{q}_{f^n}$. Then there is a (monotone) Boolean formula ψ_{f^n} computing f^n with $|\psi_{f^n}| \leq |\boldsymbol{q}'_{f^n}|$.*

(ii) *Suppose (Π_{f^n}, G) is a pure NDL-rewriting for $\mathcal{T}_{f^n}$ and $\boldsymbol{q}_{f^n}$. Then there is a monotone Boolean circuit $\mathbf{C}_{f^n}$ computing f^n with $|\mathbf{C}_{f^n}| \leq |\Pi_{f^n}|$.*

The proof proceeds by eliminating quantifiers in the rewriting and replacing its predicates with propositional variables using the fact that, in ABoxes $\mathcal{A}_{\boldsymbol{\alpha}}$, these predicates can only be true on the individual a. Lemmas 1 and 2 ensure that the resulting Boolean formula or circuit computes f^n. The next lemma shows that circuits computing f^n can be turned into pure rewritings for $\boldsymbol{q}_{f^n}$ and $\mathcal{T}_{f^n}$.

Lemma 4. (i) *Suppose $\boldsymbol{q}_n$ is an FO-sentence such that $(\mathcal{T}_{f^n}, \mathcal{A}_{\boldsymbol{\alpha}}) \models \boldsymbol{q}_{f^n}$ iff $\mathcal{I}_{\mathcal{A}_{\boldsymbol{\alpha}}} \models \boldsymbol{q}_n$, for all $\boldsymbol{\alpha} \in \{0,1\}^n$. Then there exists a pure FO-rewriting $\boldsymbol{q}'_n$ for $\boldsymbol{q}_{f_n}$ and $\mathcal{T}_{f_n}$ with $|\boldsymbol{q}'_n| \leq |\boldsymbol{q}_n| + p(n)$, for a polynomial p.*

(ii) *Suppose (Π_n, G) is a pure NDL-query with a propositional goal G such that $(\mathcal{T}_{f^n}, \mathcal{A}_{\boldsymbol{\alpha}}) \models \boldsymbol{q}_{f^n}$ iff $\Pi_n, \mathcal{A}_{\boldsymbol{\alpha}} \models G$, for $\boldsymbol{\alpha} \in \{0,1\}^n$. Then there is a pure NDL-rewriting (Π'_n, G') for $\boldsymbol{q}_{f^n}$ and $\mathcal{T}_{f^n}$ with $|\Pi'_n| \leq |\Pi_n| + p(n)$, p a polynomial.*

Now, results (1)–(3) formulated in the introduction can be obtained by applying Lemmas 1–4 to three concrete Boolean functions. For (1), we use the

function $\textsc{Clique}_{n,k}$ of $n(n-1)/2$ variables e_{ij}, $1 \le i < j \le n$, which returns 1 iff the graph with vertices $\{1,\dots,n\}$ and edges $\{\{i,j\} \mid e_{ij} = 1\}$ contains a k-clique. A series of papers, started by Razborov's [19], gave an exponential lower bound for the size of monotone circuits computing $\textsc{Clique}_{n,k}$: $2^{\Omega(\sqrt{k})}$ for $k \le \frac{1}{4}(n/\log n)^{2/3}$. For monotone formulas, an even better lower bound is known: $2^{\Omega(k)}$ for $k = 2n/3$ [18]. The question whether $\textsc{Clique}_{n,k}$ can be computed by a polynomial-size circuit is equivalent to whether NP $\subseteq$ P/poly.

To show (2), we use the function $\textsc{Gen}_{n^3}$ of n^3 variables x_{ijk}, $1 \le i,j,k \le n$, defined as follows. We say that 1 *generates* $k \le n$ if either $k = 1$ or, for some i and j such that $x_{ijk} = 1$, 1 generates both i and j. $\textsc{Gen}_{n^3}(x_{111},\dots,x_{nnn})$ returns 1 iff 1 generates n. It is clearly a monotone function computable by polynomial-size monotone circuits. On the other hand, any monotone formula computing $\textsc{Gen}_{n^3}$ is of size at least 2^{n^ε}, for some $\varepsilon > 0$ [17].

For (3), we use the function $\textsc{Matching}_{2n}$ of n^2 variables e_{ij}, $1 \le i,j \le n$, which returns 1 iff there is a *perfect matching* in a bipartite graph G with vertices $\{v_1^1,\dots,v_n^1,v_1^2,\dots,v_n^2\}$ and edges $\{\{v_i^1,v_j^2\} \mid e_{ij} = 1\}$, i.e., a subset E of edges in G such that every node in G occurs exactly once in E. An exponential lower bound $2^{\Omega(n)}$ for the size of monotone formulas computing this function was obtained in [18]. However, $\textsc{Matching}_{2n}$ is computable by non-monotone formulas of size $n^{O(\log n)}$ [4].

The exponential bounds for the size of rewritings can be reduced to *polynomial* by using two extra constants, say 0 and 1, which are included in the domain of every $\mathcal{I}_\mathcal{A}$ (see [9] and [10] for polynomial-size NDL- and PE-rewritings, respectively). As these constants may not occur in the original query and intended ABoxes, we call such rewritings *impure* (in [9] and [10], 0, 1, $=$ and polynomially-many fresh existentially quantified variables are used to guess some part of the canonical model). The difference between pure and impure rewritings is similar to the difference between deterministic circuits and nondeterministic circuits with additional existentially quantified input variables. For example, $\textsc{Clique}_{n,k}$ is computed by a polynomial-size monotone nondeterministic circuit, but not by a monotone deterministic circuit of polynomial size [19].

4 Why are Pure Rewritings so Long?

Let $\boldsymbol{q}(\boldsymbol{x}) = \exists \boldsymbol{y}\, \varphi(\boldsymbol{x},\boldsymbol{y})$ be a connected CQ and $\mathcal{T}$ a TBox. Let $\mathsf{term}\,\boldsymbol{q}$ be the set of terms in $\boldsymbol{q}$. Denote by $\mathcal{C}_\mathcal{T}$ the disjoint union of the canonical models for consistent $(\mathcal{T}, \{R(a_{RB}, b_{RB}), B(b_{RB})\})$, in which the generating relation is extended with $a_{RB} \rightsquigarrow b_{RB}$. The *RB-subtree* of $\mathcal{C}_\mathcal{T}$ has root a_{RB} and consists of the full subtree of $\mathcal{C}_\mathcal{T}$ with root b_{RB} extended with the edge (a_{RB}, b_{RB}). We also need the formulas

$$\mathsf{ext}_C(x) = \bigvee_{A \sqsubseteq_\mathcal{T} C} A(x) \;\vee \bigvee_{\exists R \sqsubseteq_\mathcal{T} C} \exists y\, R(x,y), \qquad \mathsf{ext}_P(x,y) = \bigvee_{R \sqsubseteq_\mathcal{T} P} R(x,y),$$

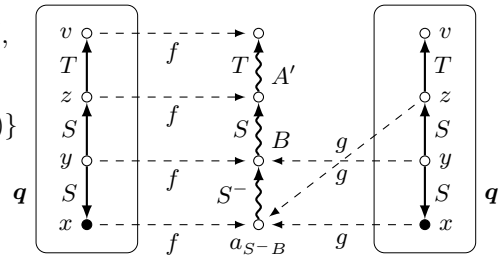
for concepts C and role names P. Suppose $\mathcal{C}_\mathcal{K} \models^\mathfrak{a} \varphi$, $\mathcal{K} = (\mathcal{T},\mathcal{A})$, where $\mathfrak{a}$ is an assignment of elements of $\Delta^{\mathcal{C}_\mathcal{K}}$ to the variables in φ under which $\mathfrak{a}(x) \in \mathsf{ind}(\mathcal{A})$

for all $x \in \boldsymbol{x}$. Consider an atom $P(t,t') \in \boldsymbol{q}$ with bound variables t, t' and assume that $\mathfrak{a}(t_0) \in \mathsf{ind}(\mathcal{A})$, for some $t_0 \in \mathsf{term}\,\boldsymbol{q}$. The assignment $\mathfrak{a}$ can send t and t' to four different locations in $\mathcal{C}_\mathcal{K}$: (A) $\mathfrak{a}(t), \mathfrak{a}(t') \in \mathsf{ind}(\mathcal{A})$; (B) $\mathfrak{a}(t) \in \mathsf{ind}(\mathcal{A})$, $\mathfrak{a}(t') \notin \mathsf{ind}(\mathcal{A})$; (B$^-$) $\mathfrak{a}(t) \notin \mathsf{ind}(\mathcal{A})$, $\mathfrak{a}(t') \in \mathsf{ind}(\mathcal{A})$; (O) $\mathfrak{a}(t), \mathfrak{a}(t') \notin \mathsf{ind}(\mathcal{A})$. Let us see how these alternatives can be reflected in a rewriting. In case (A) we have $\mathcal{C}_\mathcal{K} \models^\mathfrak{a} P(t,t')$ iff $\mathcal{A} \models^\mathfrak{a} \mathsf{ext}_P(t,t')$. Case (B) is possible only if, for some concept $\exists R.B$, we have $\mathfrak{a}(t) \rightsquigarrow w_{[RB]}$, $R \sqsubseteq_\mathcal{T} P$ and the atoms of $\boldsymbol{q}$ 'linked to' t' can be mapped into the RB-subtree of $\mathcal{C}_\mathcal{T}$. To illustrate, consider an example.

Example 1. Let $\mathcal{T}$ and $\boldsymbol{q}$ be as in the picture below. The answer variable x must be mapped by $\mathfrak{a}$ to an ABox element. However, y can be mapped either to an ABox element or to the point $\mathfrak{a}(x) \cdot w_{[S^-B]}$ provided that $\mathfrak{a}(x)$ is an instance of $\exists S^-.B$. In the latter case, we have two ways of mapping z: either to $\mathfrak{a}(x) \cdot w_{[S^-B]}w_{[SA']}$, in which case we must set $\mathfrak{a}(v) = \mathfrak{a}(x) \cdot w_{[S^-B]}w_{[SA']}w_{[T\top]}$, or to $\mathfrak{a}(x)$, provided that $\mathfrak{a}(x)$ is an instance of $\exists T$. Thus we have two (partial) maps f and g from $\boldsymbol{q}$ into the S^-B-subtree of $\mathcal{C}_\mathcal{T}$ shown below.

$$\mathcal{T} = \{\ A \sqsubseteq \exists S^-.B,\quad B \sqsubseteq \exists S.A',\quad A' \sqsubseteq \exists T\ \}$$

$$\boldsymbol{q}(x) = \{S(y,x),\ S(y,z),\ T(z,v)\}$$



These observations motivate our key definition. Given a pair (t,t') of adjacent terms in $\boldsymbol{q}$, a *tree witness*[6] *for* (t,t') is a homomorphism f from the query $\boldsymbol{q}_f = \{E(\boldsymbol{s}) \in \boldsymbol{q} \mid \boldsymbol{s} \subseteq \mathsf{dom}\, f,\ \boldsymbol{s} \not\subseteq [t]_f\}$ to the RB-subtree of $\mathcal{C}_\mathcal{T}$, for some $\exists R.B$, such that $f(t) = a_{RB}$, $\mathsf{dom}\, f$ is the smallest set containing t, t' for which $s' \in \mathsf{dom}\, f$ whenever $S(s,s') \in \boldsymbol{q}$ with $s \in \mathsf{dom}\, f \setminus [t]_f$, and all $s \in \mathsf{dom}\, f \setminus [t]_f$ are bound variables in $\boldsymbol{q}$. Here $\sim_f$ denotes the equivalence relation on $\mathsf{dom}\, f$ defined by taking $s \sim_f s'$ iff $f(s) = f(s')$ and $[t]_f$ the equivalence class of t. In Example 1, f and g are two tree witnesses for (x,y). Returning back to case (B), we can say now that there must exist a tree witness f for (t,t') such that $\mathfrak{a}(t)$ satisfies the following *tree-witness formula* tw_f for f with $f(t) = a_{RB}$:

$$\mathsf{tw}_f \quad = \quad \mathsf{ext}_{\exists R.B}(t) \quad \wedge \bigwedge_{s \in [t]_f} (t = s) \quad \wedge \bigwedge_{E(\boldsymbol{s}) \in \boldsymbol{q},\ \boldsymbol{s} \subseteq [t]_f} \mathsf{ext}_E(\boldsymbol{s}).$$

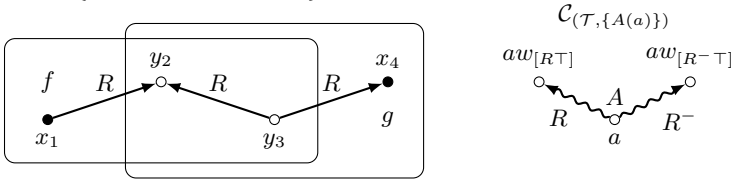
Case (B$^-$) is symmetric, and in case (O) there must exist $S(s,s') \in \boldsymbol{q}$ for which (B) holds, with $P(t,t')$ being 'covered' by the tree witness for (s,s').

This analysis suggests the following idea for a rewriting. We guess pairs of adjacent terms (t,t') in $\boldsymbol{q}$ that will be mapped to edges of the tree part of $\mathcal{C}_\mathcal{K}$

[6] A different notion of tree witness was used for *DL-Lite*$^\mathcal{N}_{horn}$ [13], where the structure of the canonical models ensured uniqueness of every tree witness.

and the tree witnesses that will cover them, as in cases (B), (B$^-$) and (O). The part of $\boldsymbol{q}$ that is not covered by these tree witnesses will be mapped to $\mathsf{ind}(\mathcal{A})$, as in case (A). The query representing the guesses is then evaluated over $\mathcal{I}_\mathcal{A}$. The following example shows, however, that this idea needs a refinement.

Example 2. Let $\boldsymbol{q}(x_1, x_4) = \{R(x_1, y_2), R(y_3, y_2), R(y_3, x_4)\}$, shown below on the left, and $\mathcal{T} = \{A \sqsubseteq \exists R,\ A \sqsubseteq \exists R^-\}$.



Clearly, there is a tree witness f for (x_1, y_2) with $\mathsf{dom}\, f = \{x_1, y_2, y_3\}$ and $[x_1]_f = \{x_1, y_3\}$, and a tree witness g for (x_4, y_3) with $\mathsf{dom}\, g = \{x_4, y_3, y_2\}$ and $[x_4]_g = \{x_4, y_2\}$. Although these tree witnesses cover the whole query $\boldsymbol{q}$, they are only 'realised,' say, in the canonical model $\mathcal{C}_{(\mathcal{T},\{A(a)\})}$ (shown above on the right) under *conflicting* maps: f sends x_1, y_3 to a and y_2 to $aw_{[R\top]}$, while g sends x_4, y_2 to a and y_3 to $aw_{[R^-\top]}$; in fact, $(\mathcal{T}, \{A(a)\}) \not\models \boldsymbol{q}(a, a)$.

Tree witnesses f and g, for (t, t') and (s, s'), respectively, are *compatible* if $\mathsf{dom}\, f \cap \mathsf{dom}\, g \subseteq [t]_f \cap [s]_g$. If f and g are incompatible and neither $\mathsf{dom}\, f \subseteq \mathsf{dom}\, g$ nor $\mathsf{dom}\, g \subseteq \mathsf{dom}\, f$, then we call f and g *conflicting* (e.g., f and g in Example 2). A set Ξ of tree witnesses is called *consistent* if all pairs of tree witnesses in Ξ are compatible. Let

$$\boldsymbol{q}_e(\boldsymbol{x}) \;=\; \mathsf{detached}_{\boldsymbol{q}} \;\vee \bigvee_{\Xi \text{ consistent}} \exists \boldsymbol{y}\, \Big(\bigwedge_{f \in \Xi} \mathsf{tw}_f \;\wedge \bigwedge_{\substack{E(\boldsymbol{s}) \in \boldsymbol{q} \\ \boldsymbol{s} \not\subseteq \mathsf{dom}\, f, \text{ for all } f \in \Xi}} \mathsf{ext}_E(\boldsymbol{s}) \Big),$$

where $\mathsf{detached}_{\boldsymbol{q}} = \bot$ if $\boldsymbol{x} \neq \emptyset$; otherwise it is a disjunction of the sentences $\exists x\, \mathsf{ext}_{\exists R.B}(x)$ such that there is a homomorphism from $\boldsymbol{q}$ to the RB-subtree of $\mathcal{C}_\mathcal{T}$. The next theorem shows that $\boldsymbol{q}_e$ is a pure PE-rewriting for $\boldsymbol{q}$ and $\mathcal{T}$:

Theorem 2. *For all $\mathcal{A}$ and $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$, we have $(\mathcal{T}, \mathcal{A}) \models \boldsymbol{q}(\boldsymbol{a})$ iff $\mathcal{I}_\mathcal{A} \models \boldsymbol{q}_e(\boldsymbol{a})$.*

Example 3. Let $\boldsymbol{q}(x) = \{R_i(x, y_i) \mid i \leq n\}$ and $\mathcal{T} = \{A_i \sqsubseteq \exists R_i \mid i \leq n\}$. Each pair (x, y_i) gives rise to one tree witness f_i with $\mathsf{tw}_{f_i} = A_i(x) \vee \exists y\, R_i(x, y)$, and $\boldsymbol{q}_e = \bigvee_{N \subseteq [0,n]} \exists \boldsymbol{y}\, (\bigwedge_{i \in N} \mathsf{tw}_{f_i} \wedge \bigwedge_{j \notin N} R_j(x, y_j))$.

The size of $\boldsymbol{q}_e$ is $O((n_{\mathcal{T},\boldsymbol{q}} + 1)^{|\boldsymbol{q}|} \cdot |\mathcal{T}| \cdot |\boldsymbol{q}|^2)$, where $n_{\mathcal{T},\boldsymbol{q}}$ is the number of distinct tree witnesses. Now, we observe that if

(conf) *there are no conflicting tree witnesses for $\boldsymbol{q}$ and $\mathcal{T}$*

then $\boldsymbol{q}_e$ can be transformed to the query

$$\boldsymbol{q}_c(\boldsymbol{x}) \;=\; \mathsf{detached}_{\boldsymbol{q}} \;\vee\; \exists \boldsymbol{y} \bigwedge_{\substack{\{t,t'\} \\ t,t' \text{ adjacent}}} \Big[\bigwedge_{E(\boldsymbol{s}) \in \boldsymbol{q},\ \boldsymbol{s} \subseteq \{t,t'\}} \mathsf{ext}_E(\boldsymbol{s}) \;\vee \bigvee_{\substack{f \text{ is a tree witness} \\ t,t' \in \mathsf{dom}\, f}} \mathsf{tw}_f \Big]$$

(if $\boldsymbol{q}$ has no binary predicates then $\boldsymbol{q}_c = \boldsymbol{q}_e$).

Theorem 3. *If $\mathcal{T}$ and $\boldsymbol{q}$ satisfy* **(conf)** *then, for any ABox $\mathcal{A}$ and $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$, we have $(\mathcal{T}, \mathcal{A}) \models \boldsymbol{q}(\boldsymbol{a})$ iff $\mathcal{I}_\mathcal{A} \models \boldsymbol{q}_c(\boldsymbol{a})$.*

The size of $\boldsymbol{q}_c$ is $O(n_{\mathcal{T},\boldsymbol{q}} \cdot |\mathcal{T}| \cdot |\boldsymbol{q}|^2)$. So, if **(conf)** holds and $n_{\mathcal{T},\boldsymbol{q}}$ is polynomial then $\boldsymbol{q}_c$ is a *polynomial* pure PE-rewriting of $\boldsymbol{q}$ and $\mathcal{T}$. For instance, the exponential $\boldsymbol{q}_e$ of Example 3 reduces to polynomial $\boldsymbol{q}_c = \exists \boldsymbol{y} \bigwedge_{i \leq n} (R_i(x, y_i) \vee \mathsf{tw}_{f_i})$. Note, however, that the CQs and TBoxes used in Section 3 generate *exponentially many* distinct tree witnesses.

We show now that, for a large class of CQs $\boldsymbol{q}$ and TBoxes $\mathcal{T}$, all tree witnesses f for each (t, t') in $\boldsymbol{q}$ (even if there are exponentially many of them) can be represented by a polynomial formula. Observe that each tw_f is determined by a concept $\exists R.B$ (such that the RB-subtree of $\mathcal{C}_\mathcal{T}$ contains the range of f) and the equivalence relation $\sim_f$ on $\mathsf{dom}\, f$. As the number of concepts $\exists R.B$ is linear in $|\mathcal{T}|$, the rewriting $\boldsymbol{q}_c$ may be regarded polynomial if we show that all tree witnesses for each (t, t') have the same equivalence relation. For example, let $\boldsymbol{q}(x) = \{S(x, y), R(y, z_i) \mid 1 \leq i \leq n\}$ and $\mathcal{T} = \{A \sqsubseteq \exists S,\ \exists S^- \sqsubseteq \exists R.B_1,\ \exists S^- \sqsubseteq \exists R.B_2\}$. There are 2^n tree witnesses for (x, y), as each z_i can be mapped either to a B_1- or a B_2-point in $\mathcal{C}_\mathcal{T}$, and yet they all define the same equivalence relation and the same tree-witness formula $\mathsf{ext}_{\exists S}(x)$.

We formalise this intuition in the following definition. For a pair (t, t') of adjacent terms in $\boldsymbol{q}$, we call $\boldsymbol{f} = (\mathsf{dom}\, \boldsymbol{f}, \sim_{\boldsymbol{f}})$ a *universal tree witness for* (t, t') if $\mathsf{dom}\, \boldsymbol{f}$ is a subset of $\mathsf{term}\, \boldsymbol{q}$ with $t, t' \in \mathsf{dom}\, \boldsymbol{f}$ and $\sim_{\boldsymbol{f}}$ is an equivalence relation on $\mathsf{dom}\, \boldsymbol{f}$ such that $\boldsymbol{q}_{\boldsymbol{f}} = \big\{E(\boldsymbol{s}) \in \boldsymbol{q} \mid \boldsymbol{s} \subseteq \mathsf{dom}\, \boldsymbol{f}, \boldsymbol{s} \not\subseteq [t]_{\boldsymbol{f}}\big\}/_{\sim_{\boldsymbol{f}}}$ is a tree-shaped query with root $[t]_{\boldsymbol{f}}$ and, for every tree witness g for (t, t'),

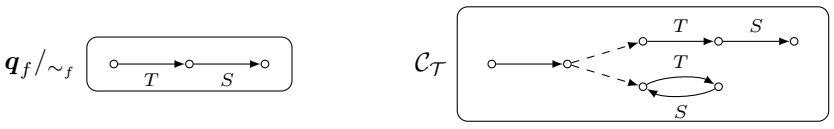
- $\mathsf{dom}\, g = \mathsf{dom}\, \boldsymbol{f}$ and there exists a homomorphism $h \colon \boldsymbol{q}_{\boldsymbol{f}} \to \mathcal{C}_\mathcal{T}$ that preserves the distance from the root and $g(s) = h([s]_{\boldsymbol{f}})$, for every $s \in \mathsf{dom}\, \boldsymbol{f}$.

A universal tree witness $\boldsymbol{f}$ for (t, t') is not a tree witness in the sense of our original definition, but rather a convenient structure representing all tree witnesses for (t, t'): we can merge the tree-witness formulas for (t, t') into one formula

$$\mathsf{tw}_{\boldsymbol{f}} \quad = \quad \Big[\bigvee_{\exists R.B \in \Phi_{\boldsymbol{f}}} \mathsf{ext}_{\exists R.B}(t) \Big] \quad \wedge \bigwedge_{s \in [t]_{\boldsymbol{f}}} (t = s) \quad \wedge \bigwedge_{E(\boldsymbol{s}) \in \boldsymbol{q},\ \boldsymbol{s} \subseteq [t]_{\boldsymbol{f}}} \mathsf{ext}_E(\boldsymbol{s}),$$

where $\Phi_{\boldsymbol{f}}$ is the set of concepts $\exists R.B$ such there is a homomorphism h from $\boldsymbol{q}_{\boldsymbol{f}}$ to the RB-subtree of $\mathcal{C}_\mathcal{T}$ with $h([t]_{\boldsymbol{f}}) = a_{RB}$.

We now identify a class of CQs and TBoxes for which a universal tree witness is unique (if exists) and can be constructed in time polynomial in $|\boldsymbol{q}|$ and $|\mathcal{T}|$. Intuitively, for each tree witness f, we disallow situations in which $\mathcal{C}_\mathcal{T}$ and the quotient $\boldsymbol{q}_f/_{\sim_f}$ of $\boldsymbol{q}_f$ simultaneously contain fragments of the form



We say that a role S is *forward* if $u \rightsquigarrow v$ for all $(u, v) \in S^{\mathcal{C}_\mathcal{T}}$. If neither S nor its inverse S^- is forward then S is said to be a *twisty role*. A tree witness f for (t, t') is called *perfect* if, for all $T(s_1, s_2), S(s_2, s_3) \in \boldsymbol{q}_f/_{\sim_f}$ such that $s_2 \neq [t]_f$ and S is twisty, we have $\mathcal{C}_\mathcal{T} \not\models \mathit{inv}(T, S) \land \mathit{suc}(T, S)$, where

$$\mathit{suc}(T, S) = \exists x, y, z\ (T(x, y) \land S(y, z) \land (x \neq z)),$$
$$\mathit{inv}(T, S) = \exists x, y\ (T(x, y) \land S(y, x)).$$

Lemma 5. *There is a polynomial-time algorithm which, given $\boldsymbol{q}$, $\mathcal{T}$ and a pair (t, t'), checks whether all tree witnesses for (t, t') are perfect, and if this is the case, returns a unique universal tree witness $\boldsymbol{f}_{t,t'}$ for (t, t').*

Note that even though there may be exponentially many tree witnesses for (t, t'), the algorithm checks whether they all are perfect in polynomial time. We are now in a position to define our *polynomial* PE-rewriting $\boldsymbol{q}_p$ for $\boldsymbol{q}$ and $\mathcal{T}$:

$$\boldsymbol{q}_p(\boldsymbol{x}) \;=\; \mathsf{detached}_{\boldsymbol{q}} \;\lor\; \exists \boldsymbol{y} \bigwedge_{\substack{\{t,t'\} \\ t,t' \text{ adjacent}}} \Big[\bigwedge_{E(\boldsymbol{s}) \in \boldsymbol{q},\ \boldsymbol{s} \subseteq \{t,t'\}} \mathsf{ext}_E(\boldsymbol{s}) \;\lor\; \bigvee_{\substack{s,s' \text{ adjacent} \\ t,t' \in \mathsf{dom}\, \boldsymbol{f}_{s,s'}}} \mathsf{tw}_{\boldsymbol{f}_{s,s'}} \Big].$$

Theorem 4. *If all tree witnesses for $\boldsymbol{q}$ and $\mathcal{T}$ are perfect and condition* **(conf)** *is satisfied then, for any ABox $\mathcal{A}$ and any $\boldsymbol{a} \subseteq \mathsf{ind}(\mathcal{A})$, we have $(\mathcal{T}, \mathcal{A}) \models \boldsymbol{q}(\boldsymbol{a})$ iff $\mathcal{I}_\mathcal{A} \models \boldsymbol{q}_p(\boldsymbol{a})$. Moreover, $\boldsymbol{q}_p$ is constructed in time polynomial in $|\boldsymbol{q}|$ and $|\mathcal{T}|$.*

If $\mathcal{T}$ does not contain any twisty roles, then all tree witnesses in any CQ $\boldsymbol{q}$ over $\mathcal{T}$ are perfect. On the other hand, all examples of conflicting tree witnesses above involve twisty roles. The following theorem shows that this is no accident:

Theorem 5. *Let $\mathcal{T}$ be an OWL 2 QL ontology without twisty roles. Then, for any CQ $\boldsymbol{q}$, there are no conflicting tree witnesses for $\boldsymbol{q}$ and $\mathcal{T}$. Thus, $\boldsymbol{q}_p$ is a pure PE-rewriting for $\boldsymbol{q}$ and $\mathcal{T}$ and it can be constructed in polynomial time.*

Note that *OWL 2 EL* ontologies satisfy this condition, and so a polynomial rewriting similar to $\boldsymbol{q}_p$ can also be used for CQ answering over such ontologies provided that the ABoxes are complete with respect to the ontologies [12].

5 Conclusions

As we saw in Section 3, pure PE- and NDL-rewritings of CQs over *OWL 2 QL* ontologies are of exponential size in the worst case. The analysis in Section 4 showed that the length of a rewriting is related to the number of tree witnesses in the query, which reflect how various parts of the query can be homomorphically mapped to the intensional tree part of the canonical model. Thus, a rewriting can be lengthy if the original query is sufficiently long and the intensional part of the canonical model for the ontology is sufficiently complex. We proved that by restricting the interaction between inverse roles and role inclusions in ontologies and queries, we can guarantee transparent polynomial rewritings. Moreover, as shown by a series of experiments [12], real-world ontologies and CQs contain very few tree witnesses, which are never in conflict, satisfy the above mentioned restrictions, and so enjoy pure, polynomial PE-rewritings.

References

1. Acciarri, A., Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Palmieri, M., Rosati, R.: QuOnto: Querying ontologies. In: Proc. of the 20th Nat. Conf. on AI, AAAI. pp. 1670–1671 (2005)
2. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The DL-Lite family and relations. Journal of Artificial Intelligence Research (JAIR) 36, 1–69 (2009)
3. Benedikt, M., Gottlob, G.: The impact of virtual views on containment. PVLDB 3(1), 297–308 (2010)
4. Borodin, A., von zur Gathen, J., Hopcroft, J.E.: Fast parallel matrix and gcd computations. In: Proc. of FOCS. pp. 65–71 (1982)
5. Calì, A., Gottlob, G., Lukasiewicz, T.: A general Datalog-based framework for tractable query answering over ontologies. In: Proc. of PODS. pp. 77–86 (2009)
6. Calì, A., Gottlob, G., Pieris, A.: Advanced processing for ontological queries. PVLDB 3(1), 554–565 (2010)
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. of Automated Reasoning 39(3), 385–429 (2007)
8. Gottlob, G., Orsi, G., Pieris, A.: Ontological queries: Rewriting and optimization. In: Proc. of the IEEE Int. Conf. on Data Engineering, ICDE (2011)
9. Gottlob, G., Schwentick, T.: Rewriting ontological queries into small nonrecursive Datalog programs. In: Proc. of DL. vol. 745. CEUR-WS.org (2011)
10. Kikot, S., Kontchakov, R., Podolskii, V., Zakharyaschev, M.: Exponential lower bounds and separation for query rewriting. CoRR, arXiv:1202.4193, 2012.
11. Kikot, S., Kontchakov, R., Zakharyaschev, M.: On (in)tractability of OBDA with OWL 2 QL. In: Proc. of DL. vol. 745. CEUR-WS.org (2011)
12. Kikot, S., Kontchakov, R., Zakharyaschev, M.: Conjunctive query answering with OWL 2 QL. In: Proc. of KR. AAAI Press (2012) (see `www.dcs.bbk.ac.uk/~kikot`)
13. Kontchakov, R., Lutz, C., Toman, D., Wolter, F., Zakharyaschev, M.: The combined approach to query answering in DL-Lite. In: Proc. of KR. AAAI Press (2010)
14. Lutz, C., Toman, D., Wolter, F.: Conjunctive query answering in the description logic EL using a relational database system. In: Proceedings of the 21st Int. Joint Conf. on Artificial Intelligence, IJCAI 2009. pp. 2070–2075 (2009)
15. Pérez-Urbina, H., Motik, B., Horrocks, I.: A comparison of query rewriting techniques for DL-Lite. In: Proc. of DL. vol. 477. CEUR-WS.org (2009)
16. 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)
17. Raz, R., McKenzie, P.: Separation of the monotone NC hierarchy. In: Proc. of FOCS. pp. 234–243 (1997)
18. Raz, R., Wigderson, A.: Monotone circuits for matching require linear depth. J. ACM 39(3), 736–744 (1992)
19. Razborov, A.: Lower bounds for the monotone complexity of some Boolean functions. Dokl. Akad. Nauk SSSR 281(4), 798–801 (1985)
20. Rodríguez-Muro, M., Calvanese, D.: Dependencies to optimize ontology based data access. In: Proc. of DL. vol. 745. CEUR-WS.org (2011)
21. Rodríguez-Muro, M., Calvanese, D.: Semantic index: Scalable query answering without forward chaining or exponential rewritings. In: Proc. of ISWC (2011)
22. Rosati, R., Almatelli, A.: Improving query answering over DL-Lite ontologies. In: Proc. of the 12th Int. Conf. KR. AAAI Press (2010)
23. Tseitin., G.: On the complexity of derivation in propositional calculus. In: Automation of Reasoning 2: Classical Papers on Computational Logic 1967–1970 (1983)