Succinctness of Query Rewriting in OWL 2 QL: The Case of Tree-like Queries

Meghyn Bienvenu¹, Stanislav Kikot², and Vladimir Podolskii³

¹ LRI – CNRS & Université Paris Sud, Orsay, France
 ² Institute for Information Transmission Problems & MIPT, Moscow, Russia
 ³ Steklov Mathematical Institute, Moscow, Russia

Abstract. This paper further investigates the succinctness landscape of query rewriting in OWL 2 QL. We clarify the worst-case size of positive existential (PE), non-recursive Datalog (NDL), and first-order (FO) rewritings for various classes of tree-like conjunctive queries, ranging from linear queries up to bounded treewidth queries. More specifically, we establish a superpolynomial lower bound on the size of PE-rewritings that holds already for linear queries and TBoxes of depth 2. For NDL-rewritings, we show that polynomial-size rewritings always exist for tree-shaped queries with a bounded number of leaves (and arbitrary TBoxes), and for bounded treewidth queries and bounded depth TBoxes. Finally, we show that the succinctness problems concerning FO-rewritings are equivalent to well-known problems in Boolean circuit complexity. Along with known results, this yields a complete picture of the succinctness landscape for the considered classes of queries and TBoxes.

1 Introduction

For several years now, conjunctive query (CQ) answering has been a major focus of description logic (DL) research (cf. survey [19]), due to the growing interest in using description logic ontologies to query data. Formally, the problem is to compute the *certain answers* to a CQ q(x) over a knowledge base $(\mathcal{T}, \mathcal{A})$, that is, the tuples of individuals a that satisfy $\mathcal{T}, \mathcal{A} \models \mathbf{q}(a)$. Much of the work on CQ answering focuses on lightweight DLs of the DL-Lite family [5], and the corresponding OWL 2 QL profile [18]. The popularity of these languages is due to fact that they enjoy first-order (FO) *rewritability*, which means that for every CQ $\mathbf{q}(x)$ and every TBox \mathcal{T} , there exists a computable FO-query q'(x) (called a *rewriting*) such that the certain answers to q(x)over $(\mathcal{T}, \mathcal{A})$ coincide with the answers of the FO-query $\mathbf{q}'(\mathbf{x})$ over the ABox \mathcal{A} (viewed as a database). First-order rewritability provides a means of reducing CQ answering to the evaluation of FO (\sim SQL) queries in relational databases. A great many different query rewriting algorithms have been proposed for OWL 2 QL and its extensions, cf. [5, 20, 25, 6, 10, 24, 21, 7, 17, 23]. Most of these algorithms produce rewritings expressed as unions of conjunctive queries (UCQs), and the size of such rewritings can be huge, making it difficult, or even impossible, to evaluate them using standard relational database management systems.

It is not difficult to see that exponential-size rewritings are unavoidable if rewritings are given as UCQs (consider for instance the CQ $\mathbf{q}(x) = B_1(x) \wedge ... \wedge B_n(x)$ and TBox $\{A_i \sqsubseteq B_i \mid 1 \le i \le n\}$). A natural (and non-trivial) question is whether an exponential

blowup can be avoided by moving to other standard query languages, like positive existential (PE) queries, non-recursive datalog (NDL) queries, or first-order (FO-) queries⁴. More generally, under what conditions can we ensure polynomial-size rewritings? A first (negative) answer was given in [14], which proved exponential lower bounds for the worst-case size of PE- and NDL-rewritings, as well as a superpolynomial lower bound for FO-rewritings (under the widely-held assumption that NP $\not\subseteq$ P/poly). Interestingly, all three results hold already for tree-shaped CQs, which are a well-studied class of CQs that often enjoy better computational properties, cf. [28,4]. While the queries used in the proofs had a simple structure, the TBoxes induced full binary trees of depth n. This raised the question of whether better results could be obtained by considering restricted classes of TBoxes. A recent study [15] explored this question for TBoxes of depth 1 and 2, that is, TBoxes that generate canonical models whose elements are at most 1 or 2 'steps away' from the ABox (see Section 2 for a formal definition). It was shown that for depth 1 TBoxes, polysize PE-rewritings do not exist, polysize NDL-rewritings do exist, and polysize FO-rewritings exist iff $NL/poly \subseteq NC^1$. For depth 2 TBoxes, neither polysize PE- nor NDL-rewritings exist, and polysize FO-rewritings do not exist unless NP $\not\subseteq$ P/poly. These results used simpler TBoxes, but the considered CQs were no longer tree-shaped. For depth 1 TBoxes, this distinction is crucial, as it was further shown in [15] that polysize PE-rewritings do exist for tree-shaped COs.

While existing results go a fair ways towards understanding the succinctness landscape of query rewriting in OWL 2 QL, a number of questions remain open:

- What happens if we consider tree-shaped queries and bounded depth TBoxes?
- What happens if we consider generalizations or restrictions of tree-shaped CQs?

In this paper, we address these questions by providing a complete picture of the succinctness of rewritings for tree-shaped queries, their restriction to *linear and bounded* branching queries (i.e. tree-shaped CQs with a bounded number of leaves), and their generalization to *bounded treewidth queries*. More specifically, we establish a superpolynomial lower bound on the size of PE-rewritings that holds already for linear queries and TBoxes of depth 2. For NDL-rewritings, we show that polynomial-size rewritings always exist for bounded branching queries (and arbitrary TBoxes), and for bounded treewidth queries and bounded depth TBoxes. Finally, we show that the succinctness problems concerning FO-rewritings are equivalent to well-known problems in Boolean circuit complexity: $NL/poly \subseteq NC^1$ in the case of linear and bounded branching queries, and $SAC^1 \subseteq NC^1$ in the case of tree-shaped and bounded treewidth queries and bounded depth TBoxes. Along with known results, this yields a complete picture of the succinctness landscape for the considered classes of queries and TBoxes. To prove our results, we establish tight connections between Boolean functions induced by queries and TBoxes and the non-uniform complexity classes NL/poly and SAC¹, reusing and further extending the machinery developed in [14, 15].

Many proofs have been omitted for lack of space. We invite the interested reader to consult the long version [3] for full proofs and additional material.

⁴ We focus on so-called *pure* FO-rewritings, cf. for [8, 11] discussion and related results.

2 Preliminaries

OWL 2 QL In this paper, we use the simplified DL syntax of the *OWL 2 QL* profile [18]. As usual, we assume countably infinite, mutually disjoint sets N_C , N_R , and N_I of *concept names, role names,* and *individual names. Roles R* and *basic concepts B* are defined by the grammar:

 $R \quad ::= \quad r \quad | \quad r^- \qquad B \quad ::= \quad A \quad | \quad \exists R$

where $A \in N_{\mathsf{C}}$ and $r \in N_{\mathsf{R}}$. We use $\mathsf{N}_{\mathsf{R}}^{\pm}$ to refer to the set of all roles.

A *TBox* (typically denoted T) is a finite set of *inclusions* of the forms

 $B_1 \sqsubseteq B_2$ $B_1 \sqsubseteq \neg B_2$ $R_1 \sqsubseteq R_2$ $R_1 \sqsubseteq \neg R_2$

The signature of a TBox \mathcal{T} , written sig(\mathcal{T}), is the set of concept and role names that appear in \mathcal{T} . An *ABox* (typically denoted \mathcal{A}) is a finite set of assertions the form A(a) or r(a, b), where $A \in N_{\mathsf{C}}$, $r \in N_{\mathsf{R}}$, and $a, b \in \mathsf{N}_{\mathsf{I}}$. The set of individual names in \mathcal{A} is denoted $\mathsf{Inds}(\mathcal{A})$.

A TBox \mathcal{T} and ABox \mathcal{A} together form a *knowledge base* (KB) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. The semantics of KBs is defined in the usual way based on interpretations $\mathcal{I} = (\mathcal{\Delta}^{\mathcal{I}}, \cdot^{\mathcal{I}})$ [2]. We use $\sqsubseteq_{\mathcal{T}}$ to denote the subsumption relation induced by \mathcal{T} and write $P_1 \sqsubseteq_{\mathcal{T}} P_2$ if $\mathcal{T} \models P_1 \sqsubseteq P_2$, where P_1, P_2 are both concepts or roles.

Query answering and rewriting A conjunctive query (CQ) $\mathbf{q}(\mathbf{x})$ is an FO-formula $\exists \mathbf{y} \varphi(\mathbf{x}, \mathbf{y})$, where φ is a conjunction of atoms of the form $A(z_1)$ or $r(z_1, z_2)$ with $z_i \in \mathbf{x} \cup \mathbf{y}$. The free variables \mathbf{x} are called *answer variables*. Note that we assume w.l.o.g. that CQs do not contain individual names, and where convenient, we regard a CQ as the set of its atoms. We use vars(\mathbf{q}) (resp. avars(\mathbf{q})) to denote the set of variables (resp. answer variables) of \mathbf{q} . The *signature* of \mathbf{q} , denoted sig(\mathbf{q}), is the set of concept and role names in \mathbf{q} . To every CQ \mathbf{q} , we associate the undirected graph $G_{\mathbf{q}}$ whose vertices are the variables of \mathbf{q} , and which contains an edge $\{u, v\}$ whenever \mathbf{q} contains an atom r(u, v) or r(v, u). We call a CQ \mathbf{q} tree-shaped if the graph $G_{\mathbf{q}}$ is a tree⁵.

A tuple $\mathbf{a} \subseteq \text{Inds}(\mathcal{A})$ is a *certain answer* to $\mathbf{q}(\mathbf{x})$ over $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if $\mathcal{I} \models \mathbf{q}(\mathbf{a})$ for all $\mathcal{I} \models \mathcal{K}$; in this case we write $\mathcal{K} \models \mathbf{q}(\mathbf{a})$. By first-order semantics, $\mathcal{I} \models \mathbf{q}(\mathbf{a})$ iff there is a mapping $h : \text{vars}(\mathbf{q}) \to \Delta^{\mathcal{I}}$ such that (i) $h(z) \in A^{\mathcal{I}}$ whenever $A(z) \in \mathbf{q}$, (ii) $(h(z), h(z')) \in r^{\mathcal{I}}$ whenever $r(z, z') \in \mathbf{q}$, and (iii) h maps $\text{avars}(\mathbf{q})$ to $\mathbf{a}^{\mathcal{I}}$. If the first two conditions are satisified, then h is a *homomorphism* from \mathbf{q} to \mathcal{I} , and we write $h : \mathbf{q} \to \mathcal{I}$. If (iii) also holds, then we write $h : \mathbf{q}(\mathbf{a}) \to \mathcal{I}$.

To every ABox \mathcal{A} , we associate the interpretation $\mathcal{I}_{\mathcal{A}}$ whose domain is $\operatorname{Inds}(\mathcal{A})$ and whose interpretation function makes true precisely the assertions from \mathcal{A} . We say that an FO-formula $\mathbf{q}'(\mathbf{x})$ with free variables \mathbf{x} and without constants is an *FO-rewriting of* $CQ \mathbf{q}(\mathbf{x})$ and $TBox \mathcal{T}$ if, for any ABox \mathcal{A} and any $\mathbf{a} \subseteq \operatorname{Inds}(\mathcal{A})$, we have $\mathcal{T}, \mathcal{A} \models \mathbf{q}(\mathbf{a})$ iff $\mathcal{I}_{\mathcal{A}} \models \mathbf{q}'(\mathbf{a})$. If \mathbf{q}' is a positive existential formula (i.e. it only uses \exists, \land, \lor), then it is called a *PE-rewriting* of \mathbf{q} and \mathcal{T} . We also consider rewritings in the form of nonrecursive Datalog queries. We remind the reader that a *Datalog program* (typically denoted Π) is a finite set of rules $\forall \mathbf{x} (\gamma_1 \land \cdots \land \gamma_m \to \gamma_0)$, where each γ_i is an atom

⁵ Tree-shaped conjunctive queries also go by the name of *acyclic queries*, cf. [28,4]

of the form $P(x_1, \ldots, x_l)$ with $x_i \in \mathbf{x}$. The atom γ_0 is called the *head* of the rule, and $\gamma_1, \ldots, \gamma_m$ its *body*. All variables in the head must also occur in the body. A predicate P depends on a predicate Q in program Π if Π contains a rule whose head predicate is P and whose body contains Q. The program Π is called *nonrecursive* if there are no cycles in the dependence relation for Π . For a nonrecursive Datalog program Π and an atom goal(\mathbf{x}), we say that (Π , goal) is an *NDL-rewriting of* $\mathbf{q}(\mathbf{x})$ and \mathcal{T} in case $\mathcal{T}, \mathcal{A} \models \mathbf{q}(\mathbf{a})$ iff $\Pi, \mathcal{A} \models \text{goal}(\mathbf{a})$, for any ABox \mathcal{A} and any $\mathbf{a} \subseteq \text{Inds}(\mathcal{A})$.

For $\mathfrak{R} \in \{\text{PE, NDL, FO}\}\)$, we say that queries from \mathfrak{Q} and TBoxes from \mathfrak{T} have *polysize* \mathfrak{R} -*rewritings* if there exists a polynomial p such that every $\mathbf{q} \in \mathfrak{Q}$ and $\mathcal{T} \in \mathfrak{T}$ has a \mathfrak{R} -rewriting \mathbf{q}' with $|\mathbf{q}'| \leq p(|\mathbf{q}| + |\mathcal{T}|)$.

Canonical model We recall that every consistent OWL 2 QL KB $(\mathcal{T}, \mathcal{A})$ possesses a *canonical model* $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ with the property that $\mathcal{T}, \mathcal{A} \models \mathbf{q}(a)$ iff $\mathcal{C}_{\mathcal{T},\mathcal{A}} \models \mathbf{q}(a)$, for every CQ \mathbf{q} and tuple $a \subseteq \mathsf{Inds}(\mathcal{A})$. The domain of $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ consists of all individual names from \mathcal{A} and all sequences $aR_1R_2 \ldots R_n$ $(n \geq 1)$ such that

- $\mathcal{T}, \mathcal{A} \models \exists R_1(a);$ - for every $1 \leq i < n$: $\mathcal{T} \models \exists R_i^- \sqsubseteq \exists R_{i+1} \text{ and } \mathcal{T} \not\models R_i^- \sqsubseteq R_{i+1}.$

Concept and role names are interpreted as follows:

$$\begin{split} A^{\mathcal{C}_{\mathcal{T},\mathcal{A}}} &= \{ a \in \mathsf{Inds}(\mathcal{A}) \mid \mathcal{T}, \mathcal{A} \models A(a) \} \cup \{ wR \in \Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}} \mid \mathcal{T} \models \exists R^{-} \sqsubseteq A \} \\ r^{\mathcal{C}_{\mathcal{T},\mathcal{A}}} &= \{ (a,b) \mid r(a,b) \in \mathcal{A} \} \cup \{ (w,wS) \in \Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}} \times \Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}} \mid \mathcal{T} \models S \sqsubseteq r \} \cup \\ \{ (wS,w) \in \Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}} \times \Delta^{\mathcal{C}_{\mathcal{T},\mathcal{A}}} \mid \mathcal{T} \models S \sqsubseteq r^{-} \} \end{split}$$

Every individual name $a \in \mathsf{Inds}(\mathcal{A})$ is interpreted as itself: $a^{\mathcal{C}_{\mathcal{T},\mathcal{A}}} = a$.

We say that a TBox \mathcal{T} is of depth ω if there is an ABox \mathcal{A} such that $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ has an infinite domain; \mathcal{T} is of depth $d, 0 \leq d < \omega$, if d is the greatest number such that some $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ contains an element of the form $aR_1 \dots R_d$. The depth of \mathcal{T} can be computed in polynomial time, and if \mathcal{T} is of finite depth, then its depth cannot exceed $2|\mathcal{T}|$.

3 Boolean Functions as a Tool for Studying Rewritings

In this section, we introduce different representations of Boolean functions that will play an important role in our results. We assume that the reader is familiar with Boolean circuits [1, 12], built using AND, OR, and NOT gates. The *size of a circuit* C, denoted |C|, is defined as its number of gates. We will be particularly interested in *monotone circuits* (that is, circuits with no NOT gates). (Monotone) *formulas* are (monotone) circuits whose underlying graph is a tree.

Non-deterministic branching programs (NBP) are another well-known model for the representation of Boolean functions [22, 12]. An NBP is defined as a tuple P = (V, E, s, t, l), where (V, E) is an directed graph, $s, t \in V$, and l is a function that labels every edge $e \in E$ with a conjunction of propositional literals. The NBP P induces the function f_P defined as follows: for every valuation α of the propositional variables in $P, f_P(\alpha) = 1$ if and only if there is a path from s to t in the graph (V, E) such that all labels along the path evaluate to 1 under α .

3.1 Hypergraph functions and hypergraph programs

We recall hypergraph functions and programs from [15]. Let H = (V, E) be a hypergraph with vertices $v \in V$ and hyperedges $e \in E$, $E \subseteq 2^V$. A subset $E' \subseteq E$ is independent if $e \cap e' = \emptyset$, for any distinct $e, e' \in E'$. With each vertex $v \in V$ and each hyperedge $e \in E$, we associate propositional variables p_v and p_e , respectively. The hypergraph function f_H for H is given by the Boolean formula

$$f_H = \bigvee_{\text{ind. } E' \subseteq E} \left(\bigwedge_{v \in V \setminus \bigcup E'} p_v \wedge \bigwedge_{e \in E'} p_e \right).$$
(1)

A hypergraph program HGP P consists of a hypergraph $H_P = (V, E)$ and a function I_P that labels every vertex with 0, 1, p_i or $\neg p_i$ (here the p_i are propositional variables, distinct from the p_v, p_e above). An input for P is a valuation of the propositional variables in P's labels. We say that the hypergraph program P computes a Boolean function f in case, for any input α , we have $f(\alpha) = 1$ if and only if there is an independent subset of E that covers all zeros—that is, contains all the vertices in V labelled with 0 under α . A hypergraph program is monotone if there are no negated variables among its vertex labels. The size, |H|, of a hypergraph program that is based upon the hypergraph H computes the Boolean function that is obtained from the hypergraph function f_H by substituting vertex labels for vertex variables and 1 for edge variables. Conversely, it is not hard to construct for a given hypergraph H, a hypergraph program swhose labels are conjunctions of variables and their negations, rather than single literals. It is not hard to see that this does not change the power of such programs.

3.2 Upper bounds via tree witness hypergraph functions

The upper bounds in [15] rely on associating a hypergraph function with every query and TBox. As the hypergraph is defined in terms of tree witnesses, we first recall the definition of tree witnesses. Consider a CQ q and a TBox \mathcal{T} . For every role R, we let $\mathcal{T}_R = \mathcal{T} \cup \{A_R \sqsubseteq \exists R\}$ and $\mathcal{A}_R = \{A_R(a)\}$ (for some fresh concept name A_R). Suppose that $\mathbf{q}' \subseteq \mathbf{q}$ (recall that we view queries as sets of atoms) and there is a homomorphism $h: \mathbf{q}' \to \mathcal{C}_{\mathcal{T}_R,\mathcal{A}_R}$ such that h(x) = a for every $x \in avars(\mathbf{q})$. Let $\mathbf{t}_r = \{x \in vars(\mathbf{q}') \mid h(x) = a\}$, and let \mathbf{t}_i be the remaining set of (quantified) variables in \mathbf{q}' . We call the pair $\mathbf{t} = (\mathbf{t}_r, \mathbf{t}_i)$ a *tree witness for* \mathbf{q} and \mathcal{T} generated by R if $\mathbf{t}_i \neq \emptyset$ and \mathbf{q}' is a *minimal* subset of \mathbf{q} such that, for any $y \in \mathbf{t}_i$, every atom in \mathbf{q} containing ybelongs to \mathbf{q}' . In this case, we denote \mathbf{q}' by \mathbf{q}_t . Note that the same tree witness can be generated by different roles R. We let $\mathcal{O}_{\mathcal{T}}^{\mathbf{q}}$ be the set of all tree witnesses of \mathbf{q} and \mathcal{T} and use $\mathcal{O}_{\mathcal{T}}^{\mathbf{q}}[R]$ to denote those generated by R. We use $x \in \mathbf{t}$ as a shorthand for $x \in \mathbf{t}_r \cup \mathbf{t}_i$.

To every CQ q and TBox \mathcal{T} , we can naturally associate the hypergraph whose vertices are the atoms of q and whose hyperedges are the sets q_t , for tree witnesses t for q and \mathcal{T} . We denote this hypergraph by $H_{\mathcal{T}}^{\mathbf{q}}$ and call the corresponding function $f_{H_{\mathcal{T}}^{\mathbf{q}}}$ the tree witness hypergraph function of q and \mathcal{T} . It is known that the circuit complexity of $f_{H_{\mathcal{T}}^{\mathbf{q}}}$ provides an upper bound on the size of rewritings of q and \mathcal{T} .

Theorem 1 (from [15]). If $f_{H^{\mathbf{q}}_{\mathcal{T}}}$ is computed by a (monotone) Boolean formula χ then there is a (PE-) FO-rewriting of \mathbf{q} and \mathcal{T} of size $O(|\chi| \cdot |\mathbf{q}| \cdot |\mathcal{T}|)$.

If $f_{H^{\mathbf{q}}_{\mathcal{T}}}$ is computed by a monotone Boolean circuit C then there is an NDL-rewriting of \mathbf{q} and \mathcal{T} of size $O(|C| \cdot |\mathbf{q}| \cdot |\mathcal{T}|)$.

Observe that $f_{H^{\mathbf{q}}_{\mathcal{T}}}$ contains a variable $p_{\mathbf{t}}$ for every tree witness \mathbf{t} . For this reason, it can only be used to show polynomial upper bounds in cases where $|\Theta^{\mathbf{q}}_{\mathcal{T}}|$ is bounded polynomially in $|\mathbf{q}|$ and $|\mathcal{T}|$. This motivates us to consider a variant of $f_{H^{\mathbf{q}}_{\mathcal{T}}}$:

$$f'_{H^{\mathbf{q}}_{\mathcal{T}}} = \bigvee_{\substack{\Theta \subseteq \Theta^{\mathbf{q}}_{\mathcal{T}} \\ \text{independent}}} \left(\bigwedge_{\varrho \in \mathbf{q} \setminus \mathbf{q}_{\Theta}} p_{\varrho} \wedge \bigwedge_{\mathfrak{t} \in \Theta} \left(\bigwedge_{z, z' \in \mathfrak{t}} p_{z=z'} \wedge \bigvee_{\substack{R \in \mathbb{N}^{\pm}_{\mathsf{R}}, \\ \mathfrak{t} \in \Theta^{\mathbf{q}}_{\mathcal{T}}[R]}} \bigwedge_{z \in \mathfrak{t}} p_{z}^{R} \right) \right)$$
(2)

where $\mathbf{q}_{\Theta} = \bigcup_{\mathbf{t}} \mathbf{q}_{\mathbf{t}}$. Intuitively, $p_{z=z'}$ enforces that variables z and z' are mapped to elements of $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ that begin by the same ABox individual; the variable p_z^R states that z is mapped to an element that begins by an individual a satisfying $\mathcal{T}, \mathcal{A} \models \exists R(a)$.

We observe that the number of variables in $f'_{H^{\mathbf{q}}_{\mathcal{T}}}$ is polynomially bounded in $|\mathbf{q}|$ and $|\mathcal{T}|$, but $f'_{H^{\mathbf{q}}_{\mathcal{T}}}$ retains the same properties as $f_{H^{\mathbf{q}}_{\mathcal{T}}}$ regarding upper bounds.

Theorem 2. Theorem 1 continues to hold if $f_{H^{\mathbf{q}}_{\mathcal{T}}}$ is replaced by $f'_{H^{\mathbf{q}}_{\mathcal{T}}}$.

3.3 Lower bounds via primitive evaluation functions

In order to obtain lower bounds on the size of rewritings, it will prove convenient to associate to each pair $(\mathbf{q}, \mathcal{T})$ a third function $f_{\mathbf{q}, \mathcal{T}}^P$ that describes the result of evaluating \mathbf{q} on single-individual ABoxes. Given Boolean vectors $\boldsymbol{\alpha} : N_{\mathsf{C}} \cap (\operatorname{sig}(\mathcal{T}) \cup \operatorname{sig}(\mathbf{q})) \rightarrow \{0, 1\}$ and $\boldsymbol{\beta} : N_{\mathsf{R}} \cap (\operatorname{sig}(\mathcal{T}) \cup \operatorname{sig}(\mathbf{q})) \rightarrow \{0, 1\}$, we let

$$\mathcal{A}(\boldsymbol{\alpha},\boldsymbol{\beta}) = \{A(a) \mid \boldsymbol{\alpha}(A) = 1\} \cup \{r(a,a) \mid \boldsymbol{\beta}(r) = 1\}$$

and set $f_{\mathbf{q},\mathcal{T}}^{P}(\boldsymbol{\alpha},\boldsymbol{\beta}) = 1$ iff $\mathcal{T}, \mathcal{A}(\boldsymbol{\alpha},\boldsymbol{\beta}) \models \mathbf{q}(\boldsymbol{a})$, where \boldsymbol{a} is a tuple of \boldsymbol{a} 's of the required length. We call $f_{\mathbf{q},\mathcal{T}}^{P}$ the *primitive evaluation function* for \mathbf{q} and \mathcal{T} .

Theorem 3 (implicit in [15]). If \mathbf{q}' is a (PE-) FO-rewriting of \mathbf{q} and \mathcal{T} , then there is a (monotone) Boolean formula χ of size $O(|\mathbf{q}'|)$ which computes $f_{\mathbf{q},\mathcal{T}}^P$.

If (Π, G) is an NDL-rewriting of \mathbf{q} and \mathcal{T} , then $f_{\mathbf{q},\mathcal{T}}^P$ is computed by a monotone Boolean circuit \mathbf{C} of size $O(|\Pi|)$.

4 Bounded Branching Queries

It is known from [14] that tree-shaped CQs do not have polysize PE- or NDL-rewritings (nor polysize FO-rewritings, unless NP \subseteq P/poly). In this section, we investigate the robustness of these results by considering a restricted form of tree-shaped queries. We will say that a tree-shaped CQ q has k leaves if the associated graph G_q (defined in Section 2) contains exactly k vertices of degree 1. We will be interested in *bounded branching queries* (that is, tree-shaped CQs with a bounded number of leaves) and *linear queries* (having exactly 2 leaves).

4.1 Bounded branching queries, interval hypergraphs and NBPs

It is not hard to see that every linear query induces a tree witness hypergraph that is isomorphic to an *interval hypergraph*, i.e. a hypergraph H = (V, E) where $V = \{[i, i+1] \mid 1 \leq i < n\}$ for some finite n and E is a set of intervals of the form $[i, j] = \{[k, k+1] \mid 1 \leq k < j\}$ where $1 \leq i < j \leq n$. Since the hypergraph functions of interval hypergraphs can be computed by polynomial-size NBPs, the same holds for the tree witness hypergraph functions of linear queries. In fact, the following result shows that we can construct polysize NBPs not only for linear queries, but also for bounded branching queries:

Theorem 4. Fix a constant $\ell > 1$. Then there exists a polynomial p such that for every tree-shaped CQ \mathbf{q} with at most ℓ leaves and every OWL 2 QL TBox \mathcal{T} , there is an NBP of size at most $p(|q| + |\mathcal{T}|)$ that computes $f_{H^{\alpha}_{\mathcal{T}}}$.

We next give a polynomial translation from NBPs into interval hypergraph programs, thereby establishing the polynomial equivalence of these formalisms:

Theorem 5. Every function f that is computable by an NBP P is also computable by an interval hypergraph program of size polynomial in |P|.

To complete the chain, we show how to compute hypergraph functions for interval hypergraphs using primitive evaluation functions of linear CQs and TBoxes of depth 2. To every interval hypergraph H, we associate the linear CQ \mathbf{q}_H pictured below:

$$\mathbf{q}_{H} = \exists \mathbf{y} \bigwedge_{[i,i+1] \in V} (r_{i}(y_{i}, y_{i}') \land r_{i}'(y_{i}', y_{i+1})).$$

$$\underbrace{\mathbf{y}_{1}}_{y_{1}} \underbrace{\mathbf{r}_{1}}_{y_{1}'} \underbrace{\mathbf{y}_{2}'}_{y_{2}'} \underbrace{\mathbf{r}_{2}'}_{y_{2}'} \underbrace{\mathbf{y}_{3}'}_{y_{3}'} \underbrace{\mathbf{r}_{3}'}_{y_{3}'} \underbrace{\mathbf{y}_{4}'}_{y_{3}'} \underbrace{\mathbf{r}_{4}'}_{y_{4}'} \underbrace{\mathbf{y}_{4}'}_{y_{4}'} \underbrace{\mathbf{r}_{4}'}_{y_{4}'} \underbrace{\mathbf{y}_{4}'}_{y_{5}'} \underbrace{\mathbf{r}_{2}'}_{y_{5}'} \underbrace{\mathbf{y}_{3}'}_{y_{5}'} \underbrace{\mathbf{r}_{3}'}_{y_{5}'} \underbrace{\mathbf{y}_{3}'}_{y_{5}'} \underbrace{\mathbf{r}_{3}'}_{y_{5}'} \underbrace{\mathbf{y}_{4}'}_{y_{5}'} \underbrace{\mathbf{r}_{4}'}_{y_{5}'} \underbrace{\mathbf{y}_{5}'}_{y_{5}'} \underbrace{\mathbf$$

The TBox \mathcal{T}_H contains the following axioms (on the left) for each edge $[i, j] \in E$:



To the right, we illustrate the canonical model generated by B_{13} . Observe how the axioms ensure that the subquery of q_H lying between y_1 and y_4 can be mapped onto it.

Theorem 6. For every interval hypergraph H = (V, E) and for all $\alpha : V \to \{0, 1\}$ and $\beta : E \to \{0, 1\}$ we have $f_H(\alpha, \beta) = 1$ iff $f^P_{\mathbf{q}_H, \mathcal{T}_H}(\gamma)$ with γ defined as follows: $\gamma(B_{ij}) = \beta([i, j]), \gamma(r_i) = \gamma(r'_i) = \alpha([i, i+1]), and \gamma(s_{ij}) = \gamma(s'_{ij}) = 0.$

4.2 Size of Rewritings of Bounded Branching Queries

We now apply the results from Section 4.1 to derive bounds on rewriting size. It is known that there is a sequence f_n of monotone Boolean functions that are computable

by polynomial-size monotone NBPs, but all monotone Boolean formulas computing f_n are of size $n^{\Omega(\log n)}$ [13]. Using this fact, together with Theorems 1,3, 5, and 6, we obtain a strong negative result for PE-rewritings.

Theorem 7. There is a sequence of linear CQs \mathbf{q}_n and TBoxes \mathcal{T}_n of depth 2, both of polysize in n, such that any PE-rewriting of \mathbf{q}_n and \mathcal{T}_n is of size $n^{\Omega(\log n)}$.

We obtain a positive result for NDL-rewritings using Theorems 1 and 4 and the fact that NBPs are representable as polynomial-size monotone circuits [22].

Theorem 8. Fix a constant $\ell > 1$. Then all tree-shaped CQs with at most ℓ leaves and arbitrary TBoxes have polynomial-size NDL-rewritings.

Finally, we use Theorems 1,3, 5, and 6 to show that the existence of polysize FO-rewritings is equivalent to the open problem of whether $NL/poly \subseteq NC^1$.

Theorem 9. The following are equivalent:

- 1. There exist polysize FO-rewritings for all linear CQs and depth 2 TBoxes;
- 2. There exist polysize FO-rewritings for all tree-shaped CQs with at most ℓ leaves and arbitrary TBoxes (for any fixed ℓ);
- 3. There exists a polynomial function p such that every NBP of size at most s is computable by a formula of size p(s). Equivalently, $NL/poly \subseteq NC^1$.

5 Bounded Treewidth Queries

In Section 4, we gave bounds on the size of rewritings for restricted classes of treeshaped CQs. In the present section, we consider arbitrary tree-shaped queries and their natural generalization to bounded treewidth queries [9]. As the rewriting size of treeshaped queries and arbitrary TBoxes has already been studied [14], we will focus on a class of "well-behaved" TBoxes, that includes TBoxes of bounded depth as a special case. We begin by formally introducing the classes of queries and TBoxes we consider.

Bounded treewidth queries We recall that a tree decomposition of an undirected graph G = (V, E) is a pair (T, λ) such that T is an (undirected) tree and λ assigns a label $\lambda(N) \subseteq V$ to every node N of T such that the following conditions are satisfied:

- 1. For every $v \in V$, there exists a node N with $v \in \lambda(N)$.
- 2. For every edge $e \in E$, there exists a node N such that $e \subseteq \lambda(N)$.
- 3. For every $v \in V$, the nodes $\{N \mid v \in \lambda(N)\}$ induce a connected subtree of T.

The width of a tree decomposition (T, λ) is equal to $\max_N |\lambda(N)| - 1$, and the treewidth of a graph G is the minimum width over all tree decompositions of G. The treewidth of a CQ q is defined as the treewidth of the graph G_q .

Polynomial image property Let \mathcal{T} be an OWL 2 QL TBox, and let \mathbf{q} be a CQ. Then the set $W_{\mathbf{q},\mathcal{T}}$ of relevant words for \mathbf{q} and \mathcal{T} consists of all words w of length at most $|\mathcal{T}|+|q|$ such that there exists an ABox \mathcal{A} that is consistent with \mathcal{T} and a homomorphism $h: \mathbf{q} \to C_{\mathcal{T},\mathcal{A}}$ whose image contains an element of the form aw. The length bound is motivated by the following well-known fact: **Lemma 1.** If \mathcal{A} is consistent with \mathcal{T} and $\mathcal{T}, \mathcal{A} \models \mathbf{q}(a)$, then there is some $h : \mathbf{q}(a) \rightarrow \mathcal{C}_{\mathcal{T},\mathcal{A}}$ whose image is contained in $\{aw \mid a \in \mathsf{Inds}(\mathcal{A}), w \in W_{\mathbf{q},\mathcal{T}}\}$.

We say that a class \mathfrak{T} of TBoxes has the *polynomial image property* if there is a polynomial p such that for every TBox $\mathcal{T} \in \mathfrak{T}$ and every CQ q, $|W_{\mathbf{q},\mathcal{T}}| \leq p(|\mathcal{T}| + |q|)$. Observe that if $d \geq 0$ is fixed, then the class of TBoxes of depth at most d has the polynomial image property. Another relevant class of TBoxes with this property is the class of TBoxes that do not contain role inclusions.

5.1 Bounded treewidth queries and tree hypergraph programs

As in Section 4, our first step will be to relate Boolean functions induced by the query and TBox with hypergraph programs. The main difference is that in lieu of interval hypergraph programs, we will use tree hypergraph programs.

To formally define tree hypergraph programs, we must first introduce some definitions related to trees. Given a tree T with vertices u and v, the *interval* $\langle u, v \rangle$ is the set of edges that appear on the simple path connecting u and v. If v_1, \ldots, v_k are vertices of T, then the *generalized interval* $\langle v_1, \ldots, v_k \rangle$ is defined as the union of intervals $\langle v_i, v_j \rangle$ over all pairs (i, j). A hypergraph $H = (V_H, E_H)$ is a *tree hypergraph* if there is a tree $T = (V_T, E_T)$ such that $V_H = E_T$ and every hyperedge in E_H is a generalized interval of T. A hypergraph program is a *tree hypergraph program* (TreeHGP) if it is based on a tree hypergraph.

From $f'_{H^{\mathbf{q}}_{\mathcal{T}}}$ to TreeHGP. We show how to construct a TreeHGP that computes $f'_{H^{\mathbf{q}}_{\mathcal{T}}}$, given a TBox \mathcal{T} , a CQ \mathbf{q} , and a tree decomposition (T, λ) of $G_{\mathbf{q}}$ of width t. We may suppose w.l.o.g. that T contains at most $(2|q|-1)^2$ nodes, cf. [16]. In order to more easily refer to the variables in $\lambda(N)$, we construct functions $\lambda_1, \ldots, \lambda_t$ such that $\lambda_i(N) \in \lambda(N)$ and $\lambda(N) = \bigcup_i \lambda_i(N)$.

The basic idea underlying the construction is as follows: for each node N in the tree decomposition of \mathbf{q} , we select an abstract description of the way the variables in $\lambda(N)$ are homomorphically mapped into the canonical model, and we check that the selected descriptions respect the subqueries of each node and are consistent with each other. Formally, these abstract descriptions are given by the set $\Gamma_t(\mathbf{q}, \mathcal{T})$ consisting of all *t*-tuples $\boldsymbol{w} = (w_1, \ldots, w_t)$ of words from $W_{\mathbf{q}, \mathcal{T}}$. Intuitively, the words in \boldsymbol{w} specify, for each variable x in $\lambda(N)$, the path of roles that lead from the ABox to the image of x in the canonical model. We say that $\boldsymbol{w} \in \Gamma_t(\mathbf{q}, \mathcal{T})$ is *consistent with a node* N in T if:

- if $A(\lambda_i(N)) \in \mathbf{q}$, then either $\boldsymbol{w}[i] = \varepsilon$ or $\boldsymbol{w}[i] = w'R$ and $\exists R^- \sqsubseteq_{\mathcal{T}} A$
- if $r(\lambda_i(N), \lambda_j(N)) \in \mathbf{q}$, then one of the following holds:
 - $\boldsymbol{w}[i] = \boldsymbol{w}[j] = \varepsilon$
 - $\boldsymbol{w}[j] = \boldsymbol{w}[i] \cdot R$ with $R \sqsubseteq_{\mathcal{T}} r$
 - $\boldsymbol{w}[i] = \boldsymbol{w}[j] \cdot R$ with $R \sqsubseteq_{\mathcal{T}} r^{-1}$

We call a pair of tuples (w_1, w_2) compatible with the pair of nodes (N_1, N_2) if:

- $\lambda_i(N_1) = \lambda_j(N_2)$ implies that $\boldsymbol{w}_1[i] = \boldsymbol{w}_2[j]$

We assume that the elements of $\Gamma_t(\mathbf{q}, \mathcal{T})$ are numbered from 1 to M, and use ξ_i to refer to the *i*-th element. We define a tree T' that replaces each edge $\{N_i, N_j\}$ in T by the following sequence of edges:

$$\{N_i, u_{ij}^1\}, \{u_{ij}^1, v_{ij}^1\}, \{v_{ij}^1, u_{ij}^2\}, \{u_{ij}^2, v_{ij}^2\}, \dots, \{u_{ij}^M, v_{ij}^M\} \{v_{ij}^M, v_{ji}^M\} \\ \{u_{ji}^M, v_{ji}^M\}, \dots, \{u_{ji}^2, v_{ji}^2\}, \{v_{ji}^1, u_{ji}^2\}, \{u_{ji}^1, v_{ji}^1\}, \{N_j, u_{ji}^1\}$$

The desired TreeHGP $(H_{\mathbf{q},\mathcal{T}},\mathfrak{l}_{\mathbf{q},\mathcal{T}})$ is based upon T' and contains the hyperedges:

- $E_i^k = \langle u_{ij_1}^k, \dots, u_{ij_n}^k \rangle$, for every $\xi_k \in \Gamma_t(\mathbf{q}, \mathcal{T})$ that is consistent with N_i , where N_{j_1}, \dots, N_{j_n} are the neighbours of N_i $E_{ij}^{km} = \langle v_{ij}^k, v_{ji}^m \rangle$, for every pair of tuples (ξ_k, ξ_m) that is compatible with (N_i, N_j)

Vertices of the hypergraph (i.e. the edges in T') are labeled by $l_{q,T}$ as follows:

- every edge of the form $\{N_i, u_{ij}^1\}, \{v_{ij}^\ell, u_{ij}^{\ell+1}\}, \text{ or } \{v_{ij}^M, v_{ji}^M\}$ is labelled 0
- every edge of u_{ij}^{ℓ} , v_{ij}^{ℓ} with $\xi_{\ell} = w$ is labelled by the conjunction of: p_{ϱ} , if $vars(\varrho) \subseteq \lambda(N_i)$ and $\lambda_g(N_i) \in vars(\varrho)$ implies $w[g] = \varepsilon$ p_z^R , if $vars(\varrho) = \{z\} \subseteq \lambda(N_i), z = \lambda_g(N_i)$, and w[g] = Rw'• p_z^R , $p_{z'}^R$, and $p_{z=z'}$, if $vars(\varrho) = \{z, z'\} \subseteq \lambda(N_i), z = \lambda_g(N_i), z' = \lambda_{g'}(N_i)$, and either w[g] = Rw' or w[g'] = Rw'

Theorem 10. For every TBox \mathcal{T} and CQ q, the TreeHGP $(H_{q,\mathcal{T}},\mathfrak{l}_{q,\mathcal{T}})$ computes $f'_{H^{q}_{\mathcal{T}}}$. If **q** has treewidth t, then $|H_{\mathbf{q},\mathcal{T}}| \leq 8 \cdot |q|^2 \cdot |W_{\mathbf{q},\mathcal{T}}|^{2t}$.

From TreeHGP to $f_{q,\mathcal{T}}^P$. Consider a tree hypergraph H = (V, E) that is based upon the tree T whose vertices are v_1, \ldots, v_n . Let T^{\downarrow} be the directed tree obtained from T by fixing v_1 as the root and orienting edges away from v_1 .

We wish to construct a tree-shaped query q_H and TBox \mathcal{T}_H of depth 2 whose primitive evaluation function $f_{\mathbf{q}_{H},\mathcal{T}_{H}}^{P}$ can be used to compute f_{H} . The construction generalizes the one from the preceding section for linear queries. The query q_H is obtained by doubling the edges in T^{\downarrow} :

$$\mathbf{q}_{H} = \exists \boldsymbol{y} \bigwedge_{(v_{i},v_{j})\in T^{\downarrow}} (r_{ij}(y_{i},y_{ij}) \wedge r'_{ij}(y_{ij},y_{j})).$$

The TBox \mathcal{T}_H is defined as the union of \mathcal{T}_e over all hyperedges $e \in E$. Consider some hyperedge $e = \langle v_{i_1}, \ldots, v_{i_m} \rangle \in E$, and suppose w.l.o.g. that v_{i_1} is the vertex in e that is highest in T^{\downarrow} . Then \mathcal{T}_e contains $B_e \subseteq \exists s_e, \exists s_e^- \subseteq \exists s'_e$, and the axioms:

$$\begin{split} s_e &\sqsubseteq r_{i_1,k} & \text{ if } \{v_{i_1}, v_k\} \in e \\ s_e^- &\sqsubseteq r'_{j_\ell, i_\ell} & \text{ if } 1 < \ell \le n \text{ and } (v_{j_\ell}, v_{i_\ell}) \in T^{\downarrow} \text{ (so, } \{v_{j_\ell}, v_{i_\ell}\} \in e) \\ s'_e &\sqsubseteq r_{j,k}^- & \text{ if } \{v_j, v_k\} \in e, (v_j, v_k) \in T^{\downarrow}, \text{ and } v_j \ne v_{i_1} \\ s'_e &\sqsubseteq r'_{j,k} & \text{ if } \{v_j, v_k\} \in e, (v_j, v_k) \in T^{\downarrow}, \text{ and } v_k \ne v_{i_\ell} \text{ for any } 1 < \ell \le m \end{split}$$

Observe that both \mathbf{q}_H and \mathcal{T}_H are of polynomial size in |H|.

Theorem 11. For every tree hypergraph H = (V, E) and for all $\alpha : V \to \{0, 1\}$ and $\boldsymbol{\beta}: E \to \{0,1\}, f_H(\boldsymbol{\alpha}, \boldsymbol{\beta}) = 1 \text{ iff } f_{\mathbf{q}_H, \mathcal{T}_H}^P(\boldsymbol{\gamma}) = 1 \text{ where } \boldsymbol{\gamma} \text{ is defined as follows:}$ $\boldsymbol{\gamma}(B_e) = \boldsymbol{\beta}(e), \boldsymbol{\gamma}(r_{ij}) = \boldsymbol{\gamma}(r'_{ij}) = \boldsymbol{\alpha}(\{v_i, v_j\}), \text{ and } \boldsymbol{\gamma}(s_e) = \boldsymbol{\gamma}(s'_e) = 0.$

5.2 Tree hypergraph programs and SAC¹

To characterize the power of tree hypergraph programs, we consider *semi-unbounded fan-in* circuits in which NOT gates are applied only to the inputs, AND gates have fan-in 2, and OR gates have unbounded fan-in. The complexity class SAC^1 [27] is defined by considering circuits of this type having polynomial size and logarithmic depth; SAC^1 is the non-uniform analog of the class LOGCFL of all languages logspace-reducible to context-free languages [26].

We consider semi-unbounded fan-in circuits of size σ and depth log σ , where σ is a parameter, and show that they are polynomially equivalent to TreeHGP by providing reductions in both directions (details can be found in [3]).

Theorem 12. There exist polynomial functions p and p' such that:

- Every semi-unbounded fan-in circuit of size at most s and depth at most $\log \sigma$ is computable by a TreeHGP of size $p(\sigma)$.
- Every TreeHGP of size σ is computable by semi-unbounded fan-in circuit of size at most $p'(\sigma)$ and depth at most $\log p'(\sigma)$.

5.3 Size of rewritings of bounded treewidth queries

Theorems 10 and 12 together show us how to construct a polysize monotone SAC¹ circuit that computes $f'_{H^{\underline{q}}}$. Thus, by applying Theorem 2, we obtain:

Theorem 13. Fix a constant t > 0, and let \mathfrak{T} be a class of OWL 2 QL TBoxes with the polynomial image property. Then all CQs of treewidth at most t and TBoxes in \mathfrak{T} have polynomial-size NDL-rewritings.

In the case of FO-rewritings, we can show that the existence of polysize rewritings corresponds to the open question of whether $SAC^1 \subseteq NC^1$.

Theorem 14. The following are equivalent:

- 1. There exist polysize FO-rewritings for all tree-shaped CQs and depth 2 TBoxes;
- 2. There exist polysize FO-rewritings for all CQs of treewidth at most t and TBoxes from a class \mathfrak{T} with the polynomial image property (for any fixed t);
- 3. There exists a polynomial function p such that every semi-unbounded fan-in circuit of size at most σ and depth at most $\log \sigma$ is computable by a formula of size $p(\sigma)$. Equivalently, $SAC^1 \subseteq NC^1$.

6 Conclusion and Future Work

In this paper, we filled some gaps in the succinctness landscape for query rewriting in OWL 2 QL by providing new bounds on the worst-case rewriting size for various forms of tree-like queries. In doing so, we closed one of the open questions from [15].

In future work, we plan to consider additional dimensions of the succinctness landscape. For example, all existing lower bounds rely on sequences $(\mathbf{q}_n, \mathcal{T}_n)$ in which the number of roles in \mathbf{q}_n and \mathcal{T}_n grows with n. Moreover, \mathcal{T}_n often contains a large number of role inclusions. Thus, an interesting and practically relevant problem is to explore the impact of restricting the number of roles and/or the use of role inclusions.

Acknowledgments. This work was partially supported by ANR grant 12-JS02-007-01, Russian Foundation for Basic Research and the programme "Leading Scientific Schools".

References

- S. Arora and B. Barak. Computational Complexity: A Modern Approach. Cambridge University Press, New York, NY, USA, 1st edition, 2009.
- F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.
- 3. M. Bienvenu, S. Kikot, and V. Podolskii. Succinctness of query rewriting in OWL 2 QL: The case of tree-like queries. *CoRR*, abs/1406.3047, 2014.
- M. Bienvenu, M. Ortiz, M. Simkus, and G. Xiao. Tractable queries for lightweight description logics. In *Proc. of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*. AAAI Press, 2013.
- D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *Journal of Automated Reasoning*, 39(3):385–429, 2007.
- A. Chortaras, D. Trivela, and G. Stamou. Optimized query rewriting for OWL 2 QL. In Proc. of the 23rd Int. Conf. on Automated Deduction (CADE-23), volume 6803 of Lecture Notes in Computer Science, pages 192–206. Springer, 2011.
- T. Eiter, M. Ortiz, M. Šimkus, T.-K. Tran, and G. Xiao. Query rewriting for Horn-SHIQ plus rules. In Proc. of the 26th AAAI Conf. on Artificial Intelligence (AAAI 2012). AAAI Press, 2012.
- G. Gottlob, S. Kikot, R. Kontchakov, V. Podolskii, T. Schwentick, and M. Zakharyaschev. The price of query rewriting in ontology-based data access. *Artificial Intelligence*, to appear, 2014.
- G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. In V. Vianu and C. H. Papadimitriou, editors, *Proc. of the 18th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 1999)*, pages 21–32. ACM Press, 1999.
- G. Gottlob, G. Orsi, and A. Pieris. Ontological queries: Rewriting and optimization. In Proc. of the 27th Int. Conf. on Data Engineering (ICDE 2011), pages 2–13. IEEE Computer Society, 2011.
- G. Gottlob and T. Schwentick. Rewriting ontological queries into small nonrecursive datalog programs. In Proc. of the 13th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2012), pages 254–263. AAAI Press, 2012.
- 12. S. Jukna. Boolean Function Complexity: Advances and Frontiers. Springer, 2012.
- M. Karchmer and A. Wigderson. Monotone circuits for connectivity require superlogarithmic depth. In *Proc. / of the 20th Annual ACM Symposium on Theory of Computing* (STOC 1988), pages 539–550. ACM Press, 1988.
- S. Kikot, R. Kontchakov, V. V. Podolskii, and M. Zakharyaschev. Exponential lower bounds and separation for query rewriting. In *Proc. of the 39th Int. Colloquium on Automata, Languages, and Programming (ICALP 2012), Part II, volume 7392 of Lecture Notes in Computer Science*, pages 263–274. Springer, 2012.
- S. Kikot, R. Kontchakov, V. V. Podolskii, and M. Zakharyaschev. On the succinctness of query rewriting over OWL 2 QL ontologies with shallow chases. In *Proc. of the 29th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2014)*. ACM Press, 2014.
- 16. T. Kloks. Treewidth: Computations and Approximations, volume 842 of Lecture Notes in Computer Science. Springer, 1994.
- M. König, M. Leclère, M.-L. Mugnier, and M. Thomazo. A sound and complete backward chaining algorithm for existential rules. In *Proc. of the 6th Int. Conf. on Web Reasoning* and Rule Systems (RR 2012), volume 7497 of Lecture Notes in Computer Science, pages 122–138. Springer, 2012.

- B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. Lutz. OWL 2 Web Ontology Language profiles. W3C Recommendation, 11 December 2012. Available at http://www.w3.org/TR/owl2-profiles/.
- M. Ortiz and M. Simkus. Reasoning and query answering in description logics. In *Reasoning Web*, volume 7487 of *Lecture Notes in Computer Science*, pages 1–53. Springer, 2012.
- H. Pérez-Urbina, B. Motik, and I. Horrocks. A comparison of query rewriting techniques for DL-Lite. In *Proc. of the 22nd Int. Workshop on Description Logics (DL 2009)*, volume 477. CEUR-WS, 2009.
- H. Pérez-Urbina, E. Rodríguez-Díaz, M. Grove, G. Konstantinidis, and E. Sirin. Evaluation of query rewriting approaches for OWL 2. In *Proc. of SSWS+HPCSW 2012*, volume 943. CEUR-WS, 2012.
- A. Razborov. Lower bounds for deterministic and nondeterministic branching programs. In Proc. of the 8th Int. Symposium on Fundamentals of Computation Theory (FCT'91), volume 529 of Lecture Notes in Computer Science, pages 47–60. Springer, 1991.
- M. Rodríguez-Muro, R. Kontchakov, and M. Zakharyaschev. Ontology-based data access: Ontop of databases. In *Proc. of the 12th Int. Semantic Web Conf. (ISWC 2013)*, volume 8218 of *Lecture Notes in Computer Science*, pages 558–573. Springer, 2013.
- R. Rosati. Prexto: Query rewriting under extensional constraints in DL-Lite. In *Proc. of the* 9th Extended Semantic Web Conf. (EWSC 2012), volume 7295 of Lecture Notes in Computer Science, pages 360–374. Springer, 2012.
- R. Rosati and A. Almatelli. Improving query answering over DL-Lite ontologies. In Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2010), pages 290–300. AAAI Press, 2010.
- H. Venkateswaran. Properties that characterize LOGCFL. J. Computer and System Sciences, 43(2):380–404, 1991.
- 27. H. Vollmer. *Introduction to circuit complexity a uniform approach*. Texts in theoretical computer science. Springer, 1999.
- M. Yannakakis. Algorithms for acyclic database schemes. In Proc. of the 7th Int. Conf. on Very Large Data Bases (VLDB'81), pages 82–94. IEEE Computer Society, 1981.